Research Note 85-145

MOPADS Final Report
With Appendices

Joseph Polito

Pritsker & Associates, Inc.

and

K.R. Laughery

Calspan Corporation

for

Contracting Officer's Representative

Charles C. Jorgensen

ARI Field Unit at Fort Bliss, Texas

Michael H. Strub

SYSTEMS RESEARCH LABORATORY

Franklin L. Moses, Acting Director

U. S. Army

Research Institute for the Behavioral and Social Sciences

November 1984

U. S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the
Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

L. NEALE COSBY
Colonel, IN
Commander

U. S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency under the Jurisdiction of the

Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON
Technical Director

L. NEALE COSBY
Colonel, IN
Commander

3

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER Research Note 84-145	2. GOVT ACCESSION NO. AD-A162 880	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) MOPADS Final Report With Appendices		5. TYPE OF REPORT & PERIOD COVERED Final Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) J. Polito - Pritsker & Associates, Inc. K.R. Laughery - Calspan Corporation		8. CONTRACT OR GRANT NUMBER(s) MDA-903-81-C-AA06
9. PERFORMING ORGANIZATION NAME AND ADDRESS Pritsker & Associates, Inc. P.O. Box 2413 West Lafayette, IN 47906		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 2Q263739A793
11. CONTROLLING OFFICE NAME AND ADDRESS U.S. Army Research Institute for the Behavioral and Social Sciences, 5001 Eisenhower Avenue, Alexandria, VA 22333-5600		12. REPORT DATE November 1984 13. NUMBER OF PAGES 2,245
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) - -		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE - -

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

- -

18. SUPPLEMENTARY NOTES

Charles C. Jorgensen, contracting officer's representative.
Of the total 2,245 pages in the report, only 94 are in the report proper. All of the rest are in the appendices. The report itself is published without the appendices as RN 85-144, MOPADS Final Report.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Human Factors
Simulation
Air Defense
SAINT
MOPADS

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes the MOPADS (Models of Operator Performance in Air Defense Systems) simulation system. The material is presented in four major sections. The first describes the Human Factors methodology that is being used to represent operator task performance and goal seeking behavior. Times for skill-based behaviors will be "moderated" by literature-based functions that reflect the influences of factors endogenous and exogenous to the operator. Operators sequence tasks by selecting activities that improve their goals. The

(over)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

(continued)
second section describes the models for the AN/TSQ-73 and IHAWK systems that comprise MOPADS. The operator goals and tasks that are represented are discussed. The third section discusses the computer implementation of the models. A data base is used to maintain problem parameters and to manage the communications among operators and AD units. The last section describes the documentation for MOPADS. Appendixes A - C contain reports that provide user documentation. They are mandatory reading for individuals who will design, perform, and analyze simulations using MOPADS. These documents provide sufficient information for a MOPADS user to exercise the models that exist in MOPADS. Appendixes D - J are a collection of documents for the MOPADS modeler who will design and develop MOPADS models of new air defense systems and integrate them with the rest of the MOPADS system. Appendixes K - AA are a collection of reference documents that describe the methodology and software modules of MOPADS. They are intended as reference reports of primary interest to the MOPADS modeler, although MOPADS users may find some of them interesting. Keywords: Human factors engineering, Simulation.

MOPADS FINAL REPORT

CONTENTS

LIST OF APPENDIXES
(Appendixes each published under separate cover)

S. MOPADS FREE-FORMAT SYNTAX PROCESSOR (MOPADS/FFSP)

T. MOPADS USER INTERFACE (MOPADS/UI)

U. THE MOPADS DATA BASE CONTROL SYSTEM (MOPADS/DBCS)

V. MOPADS FREE-FORMAT INPUT PROGRAM (MOPADS/FFIN2)

W. DOCUMENTATION MANUAL FOR THE AN/TSQ-73 SYSTEM MODULE

X. DOCUMENTATION MANUAL FOR THE IHAWK SYSTEM MODULE

Y. THE MOPADS DATA BASE

Z. THE MOPADS DATA BASE APPLICATION PROGRAMS (MOPADS/DBAP)

AA. DOCUMENTATION MANUAL FOR THE MOPADS CONTROL MODULE (MOPADS/CNTRL) AND THE MOPADS COMMON SYSTEM MODULE PROGRAMS (MOPADS/CSMP)

LIST OF FIGURES

LIST OF FIGURES

LIST OF TABLES

I. OVERVIEW OF MOPADS

1-0 INTRODUCTION TO THE MOPADS PROJECT

The MOPADS (Models of Operator Peformance in Air Defense Systems) Project has developed modeling tools to represent the performance of human beings in complex man-machine air-defense systems. The primary goals of MOPADS were to create an analysis vehicle that is:

1. flexible - in other words, able to model a wide variety of system configurations, human factors conditions, and air defense scenarios,

2. expandable - i.e., amenable to inclusion of additional elements of air defense systems and additional human factors considerations without disrupting previously existing features and with a reasonable effort, and

3. user-oriented - in other words, sufficiently easy to use so that a professional behavioral scientist can conduct meaningful experiments without performing programming or explicit computer modeling activities.

Success in achieving these goals results in a modeling tool that permits low cost analysis of human factors considerations in complex air defense situations. The analyses will be low cost to the behavioral scientist because the main expenditure of his/her time will be in creating the experimental design and in selecting the various input parameters for MOPADS. Expensive single purpose models will not need to be developed for each new application. Also, if the MOPADS system is expanded to provide new capabilities, the analyst will not have to learn a new modeling framework or new data requirements, formats, etc. Performing the experiments will be low cost also, since only a few minutes of computer time will be required for most MOPADS simulations.

2-0 HUMAN FACTORS MODELING IN MOPADS

Since operators are the main focus of MOPADS, the way in which human factors are represented in the models is central to the methodology. Human factors affect the simulation outcomes in two ways:

1. by affecting activity performance times, and

2. by affecting operator task sequencing.

Three types of operator activities are represented in MOPADS. The first is skill based behavior. This type of behavior involves actions requiring one or more skills such as tracking, detection, and fine manipulation. Skill based behaviors are simple control actions. Examples in the Air Defense setting are pressing appropriate buttons, entering alphanumeric values on a keyboard, and locating a symbol on a display. In MOPADS terminology, these actions are called "task elements" because they are the components of operator tasks. MOPADS task elements correspond to the lowest level of instruction information found in Army documentation. For example, when an AN/TSQ-73 operator performs a number hook, one of the actions required is to enter a track identifier on a keyboard. This activity obviously requires hand and arm motions, finger motions, eye motions, and head motion. MOPADS does not explicitly represent these components of the activity. Rather, the skills required for this action are specified, and the human factors modules compute the time required. The MOPADS model contains only a single modeling symbol that represents the keyboard entry. This means that there is a nearly one-to-one match between MOPADS model symbols and Army system documentation. Also, data collection is restricted to values which can be directly observed from operator actions.

The second type of behavior represented in MOPADS is rule based behavior. This type of behavior is typified by the performance of check lists. Much of the activity of air defense operators can be classified as rule based behavior. Operator tasks (sometimes called "critical tasks") are specified in Army documentation, and they are, in effect, check lists which the operators memorize. Operator tasks (or simply "tasks") involve skill based actions (task elements), and simple decisions. Examples include hooking a track, clearing alerts, and manually identifying a track.

Operator tasks are also obtained directly from Army documentation, and there is a nearly one-to-one correspondence between official documents and MOPADS representations of tasks. In the same way that a single MOPADS modeling symbol represents a single task element, a collection of such MOPADS symbols represents an operator task.

The third type behavior represented in MOPADS is knowledge based behavior. This type of behavior is strategic in nature. Air defense operators perform this type of behavior in selecting which tasks to perform in order to accomplish their mission. In particular, the operators must decide which operator tasks to perform and in what order. The MOPADS term for this activity is "task sequencing." MOPADS operators are represented as goal seekers. They evaluate the potential impact of available operator tasks on their goals when selecting the next task to perform.

It is more difficult to obtain Army documentation references for operator goals because they result from common practice, standard operating procedures and individual operator motivations. The approach taken in MOPADS has been to consult subject matter experts (in addition to Army documentation) to determine a sufficient set of operator goals.

It is clear that the way human factors modeling is performed in MOPADS will be the central concern of behavioral scientists and that no such methodology is sufficiently well established so as to elicit no controversy. Therefore, the human factors modules developed for MOPADS are stand-alone software modules. This means that other researchers will be able to experiment with the methodology in a context removed from the MOPADS project. Furthermore, the modules are sufficiently well documented so that other researchers can test alternate parameter values and even substitute human performance equations.

This approach has benefits to MOPADS in that alternate human factors representations can be readily tested within the MOPADS system, and it will hopefully provide a useful tool for behavioral scientists to evaluate theories in a unified framework.

3-0 SIMULATION METHODOLOGY IN MOPADS

The simulation methodology selected for MOPADS is discrete event simulation. This means that the computer explicitly represents each action and event (to the level of detail selected by the modeler) that occurs in the system. This is in contrast to algebraic or differential equation models which aggregate and smooth individual events to obtain overall average performance measures.

The advantages, in the MOPADS context, of a discrete event simulation are:

1. An actual time history of events is produced by the simulation. This can be important for interfacing the simulation with real time hardware simulators or field equipment, since the simulator events will more closely approximate the events in the "live" systems.

2. Discrete event modeling provides the potential for a higher degree of fidelity than do more aggregated techniques. The degree of detail can be determined by the modeler, and individual subsystems can be selectively aggregated or disaggregated as required.

3. It allows the introduction of human factors considerations at the level at which they naturally occur. In other words, individual operator actions can be affected rather than some performance measure aggregated over many actions.

The SAINT simulation language has been selected as the host language for the MOPADS operator models. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It was initially developed by Pritsker & Associates, Inc. for the U. S. Air Force to model human performance in man-machine systems and has had numerous applications including modeling of operators of remotely piloted vehicles.

The unique feature of SAINT is that it provides a formal capability to introduce human factors considerations. This is done using "moderator functions" which modify the nominal time to perform tasks. The modification, of course, is based upon the operator's ability to perform the task at that time. Thus, SAINT has features which make it immediately useful for constructing models in MOPADS.

4-0 MOPADS TERMINOLOGY

The remainder of this report discusses the above topics in greater detail, and it will be helpful if the reader familiarizes himself/herself with the "Standard MOPADS Terminology" contained in Table I-1.

Table I-1. Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data BAse Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

II. MOPADS HUMAN FACTORS MODELING

1-0 OVERVIEW OF HUMAN FACTORS IN MOPADS

MOPADS models consist of networks of tasks, and each task is a network of task elements. The way in which the tasks are connected in the network represents potential operator decisions on sequencing of tasks. Human factors will affect this system in two ways:

1. The time to perform task elements is affected ("moderated") by environmental and operator conditions. Thus, the values of independent variables such as lack of sleep, hours of continuous duty, amount of training, and tracking ability as well as environmental variables such as ambient temperature may affect the time required to perform tasks.

2. The order in which the operator performs operator tasks will be determined by the degree which a particular task will help to satisfy the operator's goals. When a task is completed, the operator will evaluate the state of each goal and estimate the impact on these goals from the various options at hand. A task will be selected based on the operator's objective and his estimates of the probable goal improvement.

2-0 TASK ELEMENT MODERATOR FUNCTIONS

2-1. The MOPADS Skills Taxonomy.

As discussed earlier, task elements are generally the lowest level of activity described in Army documentation. Since these actions are directly observable, it is relatively easy to obtain estimates of the nominal time to perform them. Such times, however, will be estimates of the "nominal" time under average conditions and by average operators. In order to systematically estimate the effect on operator performance of environmental and scenario specific conditions, the task element performance times must be known as functions of these variables.

MOPADS uses the concept of a "moderator function" to accomplish this. The moderator functions accept the nominal task element performance time and alter it ("moderate it") based on endogenous and exogenous environmental variables. In this way, an operator's

performance is affected dynamically during the simulation by the changing conditions of the system. A major activity of the MOPADS project has been the development of the moderator functions. Single observations of operators is not a practical method to obtain such functions because of the many variables and conditions that would need to be controlled.

Consider the action of an AN/TSQ-73 operator. There are literally dozens of task elements which this operator performs. It would be a prohibitively expensive chore to develop separate moderator functions for each task element. To do so would require that operators be observed performing each task element under controlled conditions while varying the independent variables of interest. Then the data would need to be reduced to a functional form suitable for use in MOPADS models.

A moderator function approach was needed that would be generic in nature. In other words, one that would be applicable to more than one task element. The approach selected was to consider each task element as an activity that requires one or more operator skills. The human factors literature contains a good deal of data on how skill performance varies with a wide variety of independent variables. The idea was to develop skill moderator functions for the skills required by air defense operators, and then to combine these functions according to the skills required for a particular task element to obtain a combined moderator tailored for the task element. A skills taxonomy similar to the one presented in Finley, Obermeyer, Bertone, Meister, & Muckler(1970) was selected since it includes skills required by air defense operators. It is shown in Table II-1.

Moderator functions for each of these skills have been developed. Each of the operator task elements is considered to require a combination of one or more of the skills shown in Table II-1. A moderator function for a particular task element is evaluated by combining the moderators for the component skills as explained in Section 2-2 below. In this way, moderators for a wide variety of air defense operator actions are available from a relatively small set of skill moderator functions.

The skill moderator functions were developed from the open human factors literature. The following five steps were performed to develop them.

Table II-1. MOPADS Skills Taxonomy.

1	Probability Estimation
2	Time Estimation
3	Long Term Memory-Sensory
4	Long Term Memory-Symbolic
5	Short Term Memory-Sensory
6	Short Term Memory-Symbolic
7	Numeric Manipulation
8	Recognition
9	Unused
10	Unused
11	Timesharing
12	Detection
13	Fine Manipulation
14	Gross Manipulation
15	Unused
16	General Physical Effort
17	Reaction Time
18	Tracking
19	Team Coordination

1. Literature Review
2. Development of a Computer Data Base
3. Selecting Independent Variables
4. Developing Moderator Functions
5. Cross-checking the Moderator Functions with a Structural Model of the Human

Since the moderator functions are to refelct the current state-of-the-art in quantitative human performance modeling, the first step was to review the current literature. This was completed and documented in Laughery (1981a). The next step was to organize this literature in a meaningful way. One part of the organization was already defined by the skill categories. Each human performance model in an article was categorized by the skill it modeled. In some cases the model grouped several skill categories (e.g., detection and identification). In this event, it was assigned to multiple skill categories. Secondly, a model within an article could be characterized by the independent variables which moderated performance of the skill. By compiling these "data" on the literature, it was possible to select all articles involving a particular skill and examine the independent variables included in the models.

To facilitate rapid analysis of the literature data base, a set of computer programs and files for data base management were prepared (Laughery, 1981b). As the literature was reviewed and the data entered into the data base, no screening was performed on the information. In other words, each time a new independent variable was encountered, it was added to the list of variables. The extent of the literature review was constrained by resources and time available and by the skills taxonomy which bounded the set of pertinent literature. The entire data base and the data base programs have been delivered to the government for further developemnt if warranted.

When the literature search was complete, it was necessary to reduce the information collected to a coherent set of moderator functions. Not all of the independent variables represented in the data base would be relevant to MOPADS, and the issue of "sufficiency" needed to be addressed. In other words, was the set of independent variables sufficient to model air defense operators.

Some type of cross-check was required. Relying solely on the literature to identify which independent variables affect skills assumes that all relationships have been studied and reported in the open literature. Since this is surely not the case, a conceptual model of the human was developed from a slightly different perspective (Laughery & Ditzian, 1981). Rather than treating the human as a performer of different types of skills, a model was developed of the human with respect to functional systems (e.g., visual, auditory, memory). Those human systems involved in each skill category from the taxonomy were then intuitively identified and arrayed into a "human systems" by "skills" matrix. As the list of independent variables was developed, those independent variables which were intuitively expected to interact with the human systems were identified and documented in a "human systems" by "independent variables" matrix. Finally, a list of hypothesized independent variables was linked to every skill by crossing the "human systems" by "skills" matrix with the "independent

variables" by "human systems" matrix, resulting in an "independent variables" by "skill categories" matrix. This matrix was compared with the moderator functions for each skill category to see if all theoretically related independent variables were included.

Promising articles in the data base were examined in greater detail and their performance models compared with the intuitive human function model. Two articles that significantly affected the final model were Siegel, Pfeiffer, Kopstein, Wilson, & Ozkaptan (1979) and Pew, Baron, Seehrer, & Miller (1977). The final set of independent variables for MOPADS is shown in Table II-2.

The three categories of independent variables specify the scope of each category of variables. Operator variables constitute the operator's state vector. Each operator has his own set of values for these variables.

The environmental variables form the environmental state vector. These variables affect all operators in the same environment. For example, both operators in an AN/TSQ-73 are affected by the same environmental state vector. Finally, task variables apply to any operator that performs the task, and each task has its own set of values for each of the variables.

Obviously, some of the independent variables do not apply to operators of the AN/TSQ-73 and IHAWK. They have been included, however, because the objective in developing the taxonomy was to characterize the skill requirements of most air defense operators. Thus, MOPADS can be expanded at a later time to include other air defense systems such as Redeye, Vulcan, etc. All of the independent variables shown in Table II-2 have been implemented in the current MOPADS software. Since only the AN/TSQ-73 and IHAWK are modeled, however, not all of the variables are used. This causes no difficulty in the present models, but it will greatly facilitate future expansions.

2-2. Computation of Combined Task Element Moderator Functions.

The current implementation of the MOPADS skill moderator functions affect only the mean task element time. The software has been developed in such a way that, if at some time appropriate data become available, then the standard deviation and distribution function can also be moderated.

Consider the operator task shown in Figure II-1 for the AN/TSQ-73, and the aggregate SAINT task model shown in Figure II-2. The trapezoids labeled (48) in Figure II-1 corresponds to the SAINT "task node" numbered 48 (with LABL: NUMHOOK) in Figure II-2. Task node 48 in Figure II-2 is the SAINT modeling symbol that corresponds to the task elements "ENTER TRACK NUMBER, FIRE UNIT, OR SITE ADDRESS ON A/N KEYBOARD" and "PRESS TASK FUNCTIONS-NUMBER HOOK".

Table II-2. MOPADS Independent Variables.

OPERATOR STATE VARIABLES

1	CORE TEMPERATURE
2	CIO VALUE
3	TIME ON TASK
4	DAYS OF DUTY
5	SEARCH DIMENSIONS
6	NUMBER FIRE UNITS
7	PERCENTAGE RECOVERY
8	PREVIOUS WORK
9	PREVIOUS REST
10	FLASH INTENSITY
11	TARGET SPEED
12	TARGET TYPE
13	TARGET SIZE
14	TARGET COLOR
15	SEARCH AREA
16	BINOCULAR USAGE
17	SLANT RANGE TO TARGET
18	TARGET TRAJECTORY
19	TARGET BACKGROUND COMPLEXITY
20	NUMBER BACKGROUND CHARACTERS
21	MESSAGE BACKLOG
22	SIGNALS PER MINUTE
23	HOURS WORKED PER WEEK
24	DAYS WITHOUT SLEEP
25	DAYS OF NIGHT DUTY
26	SIMULTANEOUS TASKS
27	CONTRAST RATIO
28	AVERAGE HOURS SLEEP
29	OBJECTIVE FUNCTION
30	GOALS CONSIDERED
31	TARGET BRIGHTNESS
32	NIGHTS
33	SKY GROUND RATIO
34	AIRCRAFT ALTITUDE
35	METEOROLOGICAL RANGE
36	THRESHOLD CONTRAST
37	TARGET HEIGHT
38	TARGET WIDTH
39	TARGET DEPTH
40	HORIZONTAL RANGE
41	NUMBER OF RESOLUTION ELEMENTS
42	NUMBER OF SUSPECT AREAS
43	AIRCRAFT SPEED
44	FIELD OF VIEW
45	OBSERVER OFFSET

Table II-2 (continued)

46		DISPLAY TARGET LOCATION
47		TARGET LOCATION
48		DISPLAY RESOLUTION
49		DISPLAY BACKGROUND HEIGHT
50		DISPLAY BACKGROUND WIDTH
51		DISPLAY BACKGROUND DEPTH
52		DISTANCE TO DISPLAY
53		DISPLAY HEIGHT
54		DISPLAY WIDTH
55		TARGET NOISE LEVEL
56		TARGET DURATION
57		EXPERIENCE
58		SIGNAL PROBABILITY
59		REST PERIODS
60		DAYS SINCE PRACTICE
61		SENSE OF DIRECTION
62		SKIN TEMPERATURE
63		TIME IN TEMPERATURE
64		PREVIOUS SKIN TEMPERATURE

ENVIRONMENTAL VARIABLES

1		DRY BULB TEMPERATURE
2		RELATIVE HUMIDITY
3		AIR MOVEMENT RATE
4		NOISE LEVEL
5		WORK AREA ILLUMINATION
6		NUMBER ON DUTY
7		VIBRATION
8		AMBIENT VAPOR PRESSURE
9		NOISE PREDICTABILITY

TASK RELATED VARIABLES

1		KILOCALORIES/MINUTE
2		NUMBER OF BRANCHES OUT
3	*	STIMULUS MODE 1
4	*	STIMULUS MODE 2
5	*	RESPONSE MODE
6	*	OBSERVER TARGET POSITION
7		CONTROL DISTANCE
8		CONTROL WIDTH
9		NUMBER OF DISPLAYS
10		NUMBER OF ALTERNATIVES
11		NUMBER SHORT TERM MEMORY ITEMS

* See Legend

Table II-2 (continued)

Legend

Stimulus Mode 1

A two-digit number
10's digit - 0 - no visual
1 - visual
1's digit 0 - no auditory
1 - auditory

Stimulus Mode 2

A three-digit number as above
100's digit - 0 - no olfactory
1 - olfactory
10's digit 0 - no kinesthetic
1 - kinesthetic
1's digit 0 - no tactile
1 - tactile

Response Mode

A two-digit number
10's digit - 0 - no vocal
1 - vocal
1's digit 0 - no tactile
1 - tactile

Observer Target Position

1 - ground to ground
2 - air to ground
3 - ground to air
4 - air to air
5 - at a display

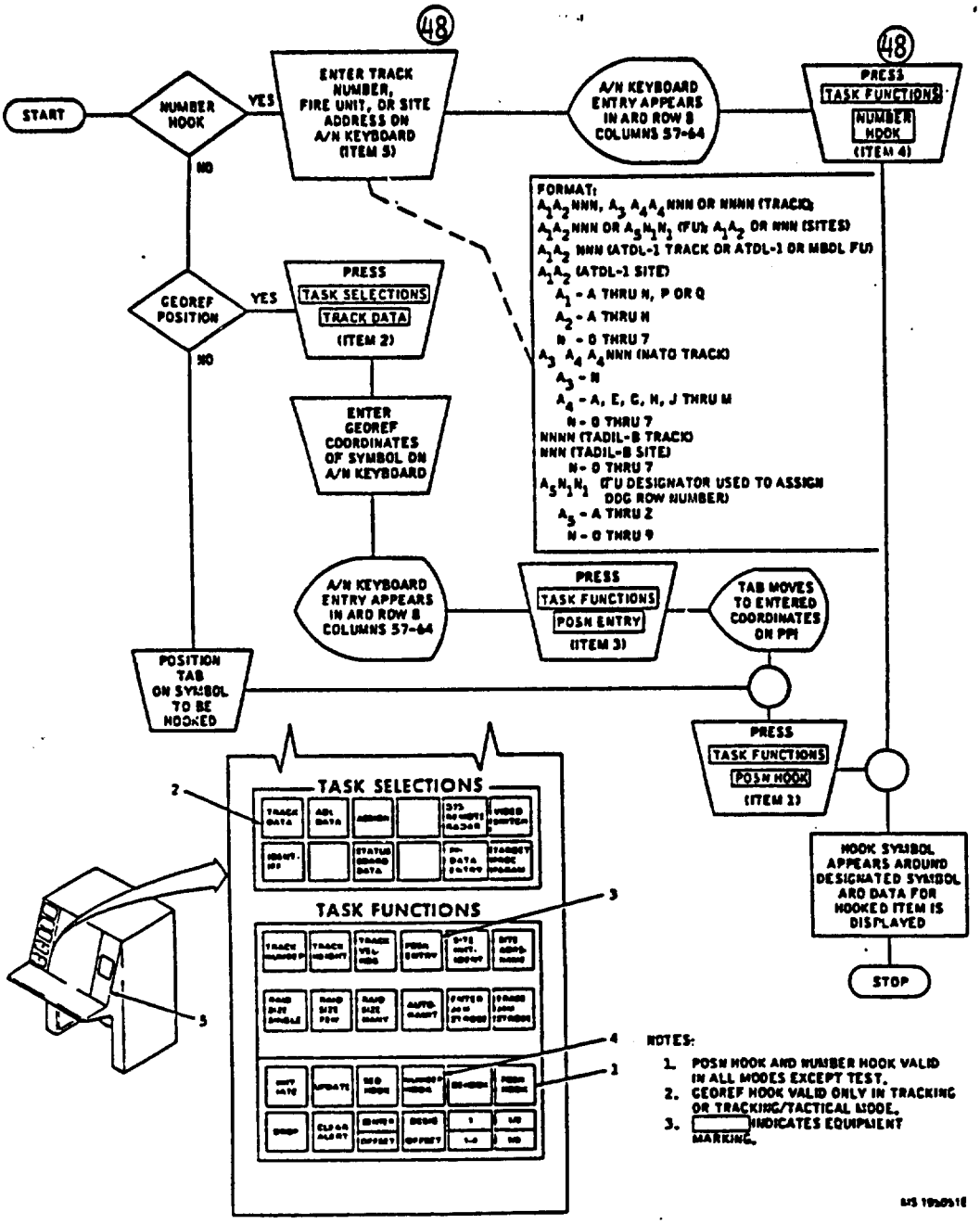


Figure 4-14. Console Hooking Procedures – Number, GEOREF, and Position Hook

Figure II-1. AN/TSQ-73 Console Hooking Procedures.

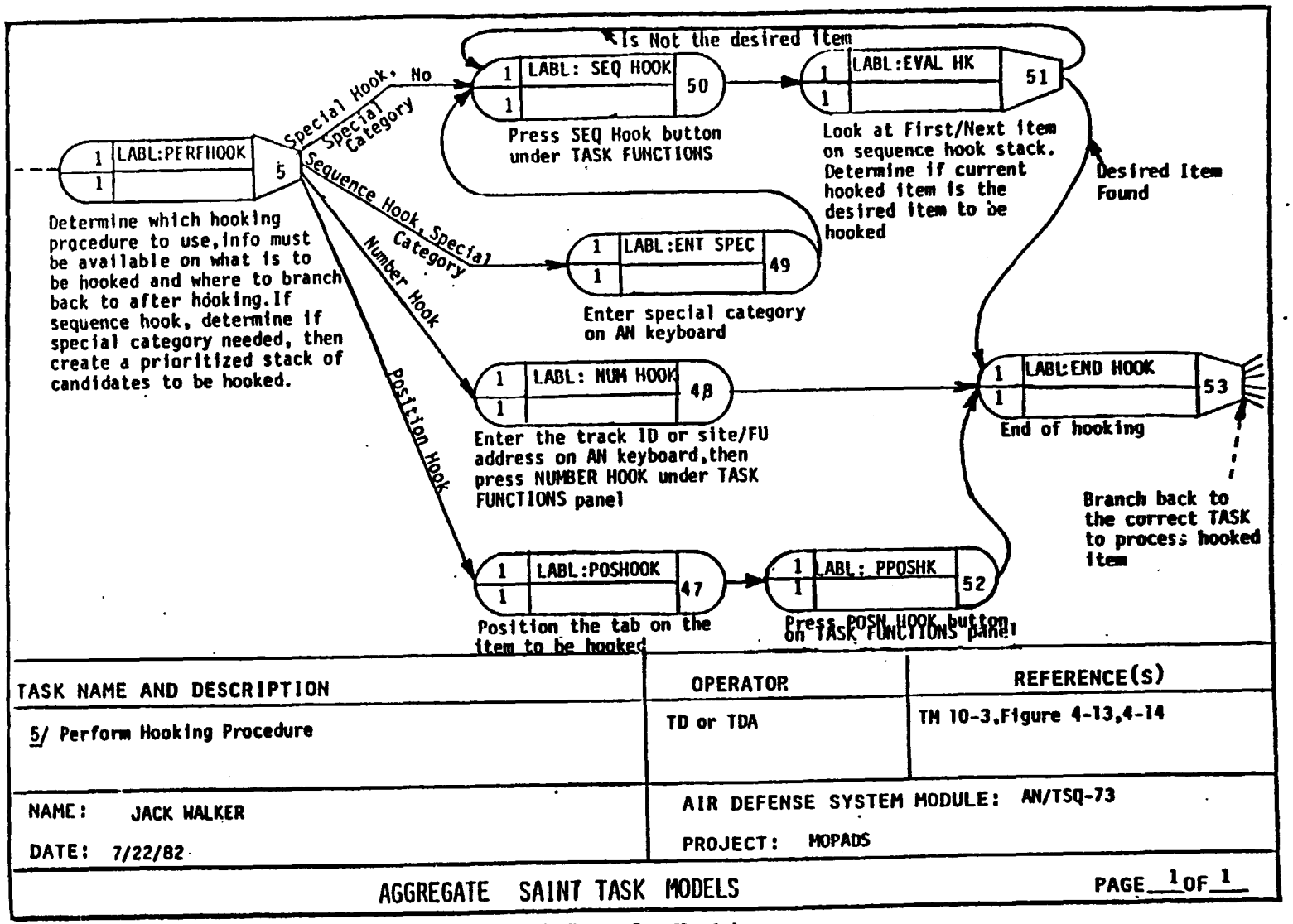


Figure II-2. Aggregate SAINT Task Model of Console Hooking.

Sample data for this task element are shown below:

Nominal Mean Time:	4.5 seconds	
Skills Required:	Fine Manipulation	75%
	Short-Term Memory-Symbolic	15%
	Detection	10%

During a MOPADS simulation, when an operator is to perform this task element, MOPADS will call the moderator functions for the three skills shown above. The moderator functions have access to the information in the operator state vector, the environmental state vector, and the task data. With this information, the moderator functions will each compute a change to the mean of 4.5 seconds.

Let D_1, D_2, and D_3 be the computed changes from the moderator functions for fine manipulation, short-term memory-symbolic, and detection, respectively. See Laughery & Ditzian (1982) & Laughery (1981) for these skill moderator functions. The combined mean is determined from

$$\mu_{moderated} = 4.5 + (.75D_1 + .15D_2 + .10D_3)$$

The obvious interpretation is that approximately 75% of the time performing the task element is in fine manipulation, etc., and minimal interaction between skills occurs. In the absence of data, these are relatively mild assumptions to achieve computable moderators for a wide variety of activities.

The moderated mean may then be used in the simulation to determine the time required for the operator to perform the task element. MOPADS allows several options in this regard:

1. Deterministic, unmoderated - always use the nominal mean time
2. Deterministic, moderated - use $\mu_{moderated}$ for the task element time
3. Stochastic, unmoderated - make a random draw from the specified distribution function whose mean is the nominal mean. Do not use the moderator functions.
4. Stochastic, moderated - make a random draw from the specified distribution function whose mean is $\mu_{moderated}$

MOPADS supports the following distribution functions:

Constant

Normal

Uniform

Erlang -1 (Exponential)

Lognormal

Beta

Gamma

The Gamma distribution is the default distribution, but the user can select any of the above.

2-3. Software Implementation.

As stated earlier, the human factors moderator function module has been developed in a way that allows it to be separated from the rest of the MOPADS software and used independently. This has been accomplished through the following design considerations:

1. Every moderator function subprogram has an identical calling sequence.

2. Moderator functions request data from the operator state vectors, environmental state vectors, and the task data through standard subprogram calls.

The implication of the above are that non-MOPADS software environments can use the moderator functions by calling them in their standard way and by providing utility programs that the moderator functions will call to access operator, environmental, and task data. MOPADS documents Laughery & Ditzian (1982) & Laughery (1981) contain the technical details.

3-0 TASK SEQUENCING METHODOLOGY

3-1. Task Sequencing Considerations.

In order to adequately model the actions of an air defense operator, the simulation must "perform" operator tasks in a way that responds to the air defense scenario. In other words, the simulated operators must perform tasks that tend toward accomplishing the mission of the air defense system. Developing a methodology that will exhibit this behavior is more difficult than simulating the operator tasks, because the simulation must represent the knowledge, experience, and motivations of the operators.

Furthermore, the simulation methodology needs to be accessible to the user. In other words, input parameters for the task sequencing algorithm should be intuitively meaningful and reasonably easy to determine. With these ideas in mind, the objectives in developing a task sequencing procedure were as follows:

The procedure should be:

1. consistent with the literature,

2. intuitively meaningful so that a MOPADS analyst can specify operator oriented parameters rather than abstract parameters that are obtained from some curve fitting procedure, and

3. relatively goal independent. This means that the parameters associated with one operator objective are nearly independent of the parameters associated with all other objectives.

Utility function and goal seeking approaches were considered. The multi-attribute utility function procedure was discarded because it did not satisfy objectives (2) and (3) above. A goal seeking approach, on the other hand, can be implemented in a way that satisfies (2) and (3) above and is consistent with the Newell & Simon (1972) view of humans as goal seekers. The principal advantage of the goal seeking approach is that the utility or "goal priority" of each goal can be computed independently of all other goals (this of course, is an assumption but a relatively mild one).

In particular, goal priority functions can be defined that are ordinal in nature, so that changes in parameters for one goal do not affect the parameters of other goals. The priority, or degree of goal satisfaction, for each goal can then be compared. This is in contrast to a multi-attribute utility approach in which a cardinal utility value is computed as a function of all goal states. The usual approach in utility theory is to develop a utility function of several variables (the goal states) from a curve fitting procedure. This is a cumbersome method for use in MOPADS because changes to individual goals for individual operators cannot be easily incorporated into the model.

3-2. The Task Sequencing Procedure.

The goal seeking behavior of the operators is characterized by the following:

1. The operators have a set of goals which they desire to satisfy simultaneously. They are capable of determining the value or state of each goal. For example, the operator may desire to maximize the distance from a critical asset to any hostile aircraft. The goal state is the minimum distance to any hostile track.

2. The operators can rank the importance of their goal states. In other words, they can assign priorities to their goals based upon their current states. For example, an AN/TSQ-73 operator can determine whether an uncleared alert message or a hostile aircraft within 30 miles of a critical asset should be attended to next.

3. The operators are capable of estimating the changes that will occur in their goal states if a particular task is performed.

4. Operators are limited in their ability to satisfy their goals. They may not be able to consider all of their goals at once.

These concepts are implemented in the following ways. For each operator goal, the goal state (denoted GS) must be explicitly specified in a way that allows GS to be assigned a unique value (e.g., GS = the number of unassigned hostile tracks). Then a goal priority function, GP, is specified for the goal that assigns a non-negative value to each value of GS. Figure II-3 shows an hypothetical example. The goal is satisfied when the goal state, GS, is between m and M. This is signified by GP = 0 when $m \leq GP \leq M$. If the goal state is less than m or greater than M, then the priority of the goal (i.e., its degree of dissatisfaction) increases linearly.

The meaning of the particular goal priority function in Figure II-3 is that the operator is satisfied and indifferent to any value of the goal state between m and M. Downside deviations (i.e., values of GS less than m) are more important than upside deviations (i.e., values of GS greater than M) since the slope for downside deviations is greater than the slope for upside deviations.

Each goal that an operator has may have a different priority function. See Figure II-4 for examples. The values of the goal priorities for each goal are compared in an ordinal fashion during task sequencing to determine the most dissatisfied goal or goals. This scheme allows the parameters for a goal priority function to be specified or changed without affecting the priority functions for other goals. Of course, complete independence is not obtained because the modeler must always be aware of the priority functions of the rest of the goals.

Determination of the goals and the goal priority functions must be accomplished with close coordination with subject matter experts since no open literature is available. Recall that the operators will select tasks in order to improve their goal states. Therefore, the modeler must specify one or more goals whose states

are affected by each task. Goal priority functions may be determined by giving pairwise comparisons to subject matter experts (e.g., given states for two goals, which goal would have the highest priority?). Since the scale of the goal priority is arbitrary, the modeler can ordinally rank the responses to achieve correct rankings of the goals. In the current MOPADS implementation, a goal priority of 10 has been used to imply an extreme emergency, so goal priority values generally fall in the range zero to 10.

The operators seek to achieve one of the following two objectives when selecting the next task.

1. Maximize the expected reduction in the average goal priority of the NG largest goal priorities.

2. Maximize the expected reduction per unit time of the average goal priority of the NG largest goal priorities.

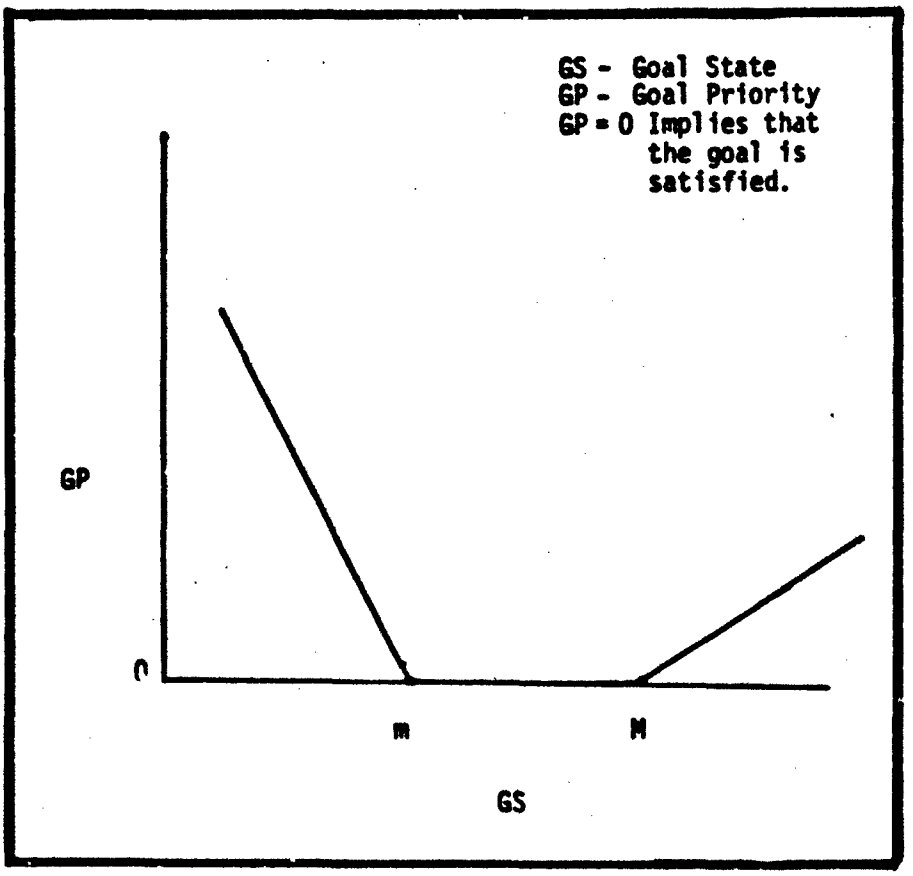


Figure II-3. Example Goal Priority Function.

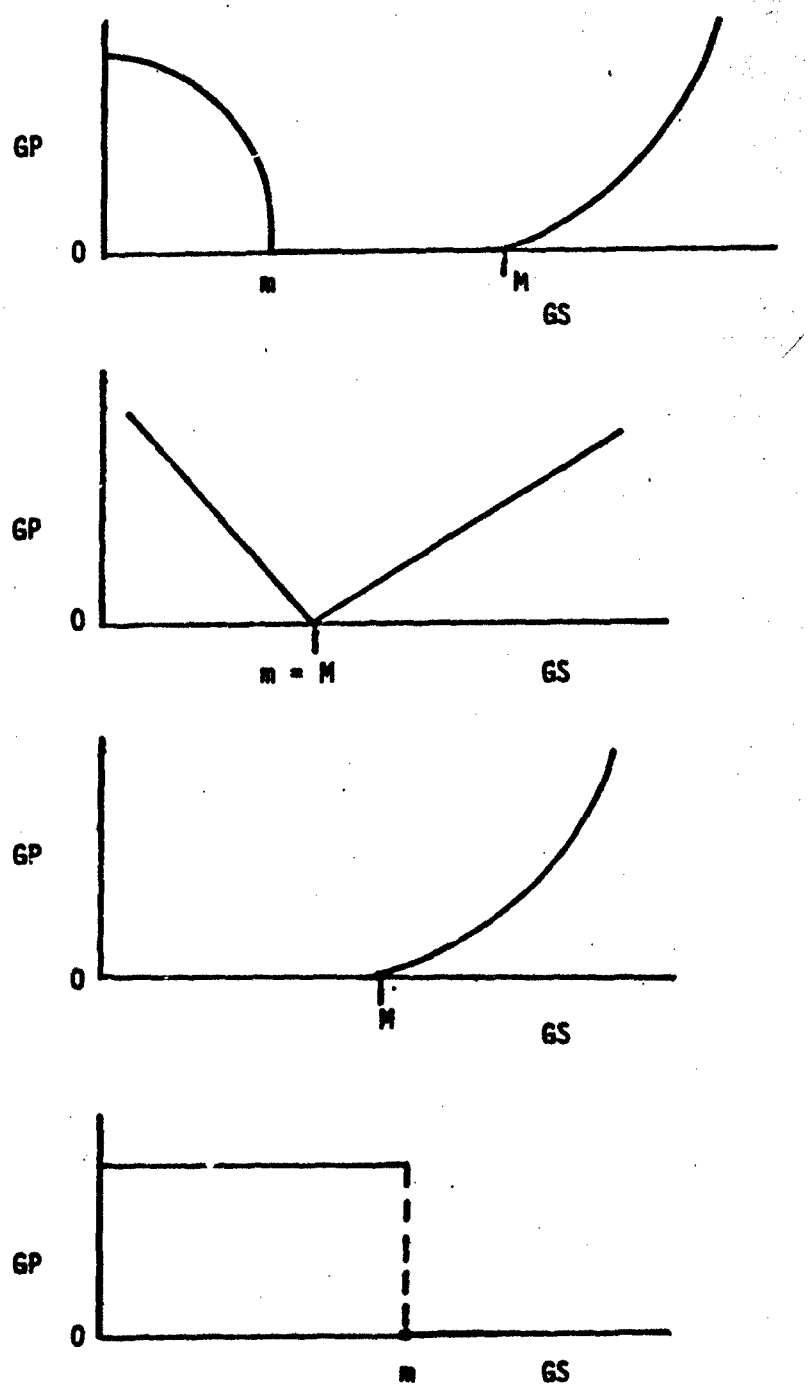


Figure II-4. Example Goal Priority Function Forms.

In the first case, the operator computes the average goal priority of the NG most dissatisfied goals. NG is a parameter of the operator which may be set by the MOPADS user. Note that if NG is one, the operator "puts out the biggest fire first." As NG increases, the operator is modeled as being able to take a more and more global view of his tasks. At the present time, NG remains fixed for each operator during the entire simulation. If relevant data were available, it would be possible to dynamically vary NG in response to operator work load or other conditions during the simulation.

The second objective function is similar to the first, but, in addition, the operator estimates the time it will take to perform each option available. With this information, the operator estimates the change in average goal priority that will occur per unit time and selects the one which gives the most rapid improvement.

Finally, a last-in-first-out task stack is maintained for each operator. Some options available to the operator may involve performing several operator tasks before re-evaluating goals again. For such a case, the tasks which will be performed in sequence are loaded in the operator's task stack. When one task is completed, the operator will immediately perform the next task on his stack. Goal evaluation is performed only when the stack is empty or a high priority alert message has been received. In the case of a high priority message, a complete goal evaluation occurs.

The procedure for goal evaluation and task selection is shown in Figure II-5. The special weighted average of the goal priorities is computed as follows (here assume that the goal priorities have been arranged in decreasing order).

$$AVNG = \sum_{i}^{NG} \left(w_i \; GP_i \right)$$

where

$$w_i = \frac{GP_i}{\sum_{i}^{NG} GP_i}$$

Thus, the maximum weight is given to the most dissatisfied goal (i.e., the one with the largest goal priority). This method prevents certain distortions in task selection that might result from the use of a simple average. For example, suppose the two

1. If the task stack is not empty and no higher priority message is pending, then select the next task from the task stack and exit.

2. Otherwise,

   a) evaluate each goal state,

   b) evaluate each goal priority function

   c) compute the weighted average goal priority

   d) For each alternative task selection -

      i) compute the expected goal states, if the task is performed

      ii) compute the expected goal priorities

      iii) compute the expected average goal priority

   e) select the task that best improves the operator's objective function

   f) load the task stack if necessary, and

   g) exit

Figure II-5. Goal Evaluation Procedure.

most dissatisfied goals (with NG = 2) are $(GP_1, GP_2) = (10,5)$. The simple average goal priority, $\bar{X}$, is 7.5 while $AVNG^2 = 8.33$. ($w_1 = .67$, $w_2 = .33$). Suppose there are two options whose expected results are shown below (the w_i are not recomputed).

	GP_1	GP_2	$\bar{X}$	AVNG
Option 1	10	1	5.5	7.0
Option 2	6	5	5.5	5.67

The simple average of the goal priorities is indifferent to the two options since they both result in a reduction of four in the sum of the goal priorities. Option 1, however, makes no improvement in the most dissatisfied goal. Selection of option 1 would represent an operator who attempts to improve the second most dissatisfied goal when there is an available option that improves the most dissatisfied goal. Using the special weighted average AVNG ensures that the operator will always pay most attention to the most dissatisfied goals even though he is attempting to consider more than one goal simultaneously.

Finally, note that the goal seeking procedure explained above is a simplified special case of utility theory. The utility function is simply a composition of the univariate goal priority functions which are combined through the AVNG function to a single value for each set of goal states. It would be a simple matter to recast the mathematical expression of the procedure in a utility theoretic framework. The goal seeking method simply assumes that a utility surrogate(the goal priority) can be expressed as a function of a subset of the set of the goal states (i.e., the priority of a goal is a function only of its own goal state).

3-3. Operator Goals for the AN/TSQ-73 and IHAWK.

The goals identified for the operators of the AN/TSQ-73 and the IHAWK arise from the basic goals of self preservation and a desire to accomplish their mission. The translation of these basic goals to operative goal statements results in goals that lead the operators to attack aircraft that present threats to themselves and the sites they are assigned to protect and to follow standard procedures to accomplish their missions. The statements of the goals for the AN/TSQ-73 and IHAWK operators are shown in Tables II-3 and II-4, respectively.

3-4. Software Implementation of Task Sequencing.

As is the case in all of the MOPADS software designs, the structure is intended to allow for future expansions (see Figure II-6). The common programs include those parts of the task sequencing

Table II-3. Operator Goal for the AN/TSQ-73.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
1	Self Defense- maximize the minimum time to arrive of any threatening track.	The minimum time for any unassigned, not-receding, hostile or unknown track to arrive if it immediately turned inbound.
2	Protect Critical Assets - maximize the minimum time to arrive at a critical asset of any threatening track.	The minimum time for any unassigned, not-receding, hostile or unknown track to arrive at any protected site if it immediately turned inbound to the site.
3	Attack Threatening Tracks	The number of unassigned, visible, hostile or unknown, not-receding tracks and the number of covered tracks.
4	Minimize the number of unidentified tracks	The number of unidentified tracks visible or visible to owned fire units.
5	Minimize the number of uninitiated video contacts	The number of visible video contacts
6	Conserve ammunition	The number of tracks being engaged by more than one fire unit

NAME: Joseph Polito
DATE: 8/25/83
MAIN DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 1 of 2

Table II-3 (continued)

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
7	Respond to Communications	The maximum priority of any outstanding message that the operator may receive and that no one else is processing.
8	Protect Friendly Tracks	The number of friendly tracks engaged or assigned.

NAME: Joseph Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS

Page 2 of 2

Table II-4. Operator Goal for the IHAWK.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
1	Engage Assigned Tracks	The minimum time for any not receding, in-range, assigned, but not engaged Track to arrive at the IHAWK or its protected site if it immediately turned inbound.
2	Self Defense	The minimum time for any hostile or unknown, not receding, in-range, unassigned Track to arrive at the IHAWK if it immediately turned inbound.
3	Protect Critical Assets	The minimum time for any hostile or unknown, not receding, in-range, unassigned Track to arrive at any protected site if it immediately turned inbound.
4	Minimize the number of unidentified Tracks.	The number of unidentified tracks.
5	Minimize the maximum priority of outstanding messages	The maximum priority of any outstanding message that the operator may receive and no one else is processing.

NAME: Joseph Polito | AIR DEFENSE SYSTEM MODULE: IHAWK
DATE: 8/25/83 | PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 1 of 2

Table II-4 (continued)

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
6	Maximize the number of missiles available	The total number of hot and cold missiles available.
7	Protect friends	The number of friendly tracks being engaged.
8	Minimize the number of unidentified, low altitude tracks.	The number of unidentified, low altitude tracks.

NAME: Joseph Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS

procedure that are independent of the system module and operator type. This includes statistics collection, branching, and certain utility programs. Each system module (designated SM1, SM2, etc. in Figure II-6) has its own entry point subprogram which will, in turn, call a program to process the goals for each operator type in the system. The task sequencing programs labeled SM1, SM2, etc. and their descenderts are documented as part of the system module documentation. In this way, new system modules can be added by "plugging in" the task sequencing programs without disturbing existing system modules.

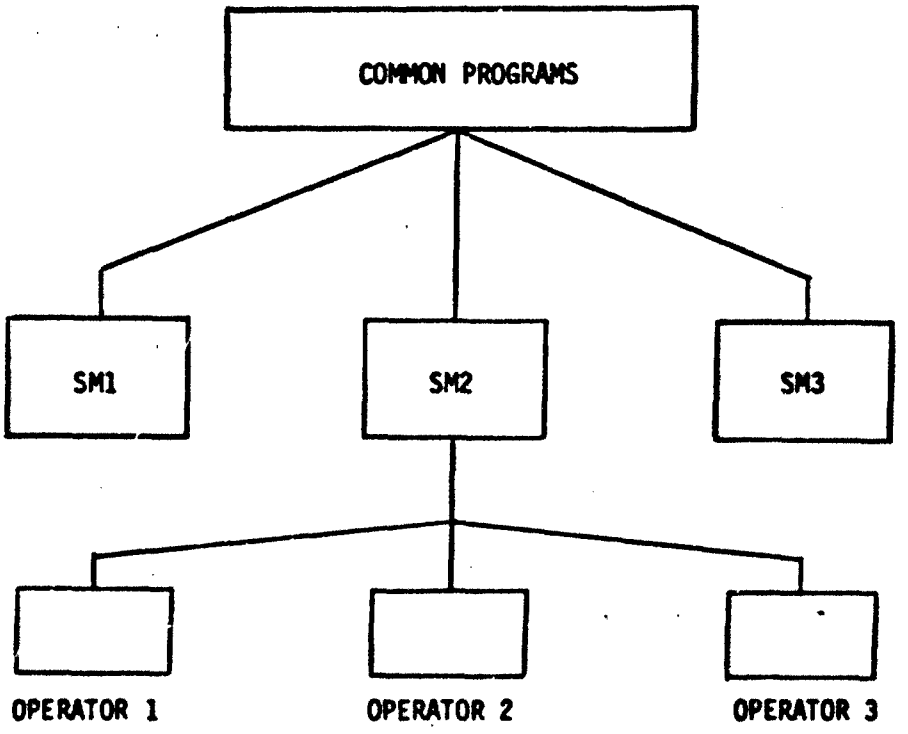


Figure II-6. Schematic Structure of MOPADS Task Sequencing Software.

III. MOPADS AIR DEFENSE MODELS

1-0 DEVELOPMENT METHODOLOGY FOR AIR DEFENSE SYSTEM MODULES

MOPADS is intended to be a long lived software system that will evolve as new system modules are developed. Since the development of system modules is the most complex maintenance activity that will be performed on MOPADS and since it is likely that many individuals will be involved in developing the modules, it is essential that a systematic development and documentation methodology be followed. Procedures for these activities have been created (Walker & Polito, 1982a,b).

The procedure for developing system modules is summarized in Figure III-1. Steps 1, 2, and 3 are data collection functions in which the MOPADS analyst and those who aid him/her collect information and systematically characterize operator and equipment modeling requirements. In steps 4, 5, and 6, task sequencing considerations and constraints are identified and formalized. Human factors are minimally or nominally represented at this stage. The purpose is to ensure that infeasible or unrealistic sequencing does not occur.

At step 11, the MOPADS modeler enters data for the new system module into the MOPADS data base. The MOPADS user interface has facilities to support this activity. At step 12, the MOPADS simulations are performed with a minimal set of existing MOPADS models to test the new system module. When this is completed, the new system model is integrated with the full set of existing MOPADS models and tested. These are steps 13 and 14.

Step 15 is the final documentation effort. A systematic procedure for documentation has been specified to aid in module development and to ensure that adequate documentation of system modules is maintained. This is crucial because it is certain that when a new module is developed, the analyst will have to refer to the documentation of other system modules. This will be necessary or desirable in steps 4, 9, 10, 12, and 13.

Each document describing a system model will be organized in the same way. Figure III-2 is a typical table of contents. Standard forms have been provided to aid in modeling which will be included in Section III of the documents. Examples of these forms are shown in Sections 2-0 and 3-0 below.

Strict adherence to these documentation standards will result in a maintainable analysis tool with a long useful life.

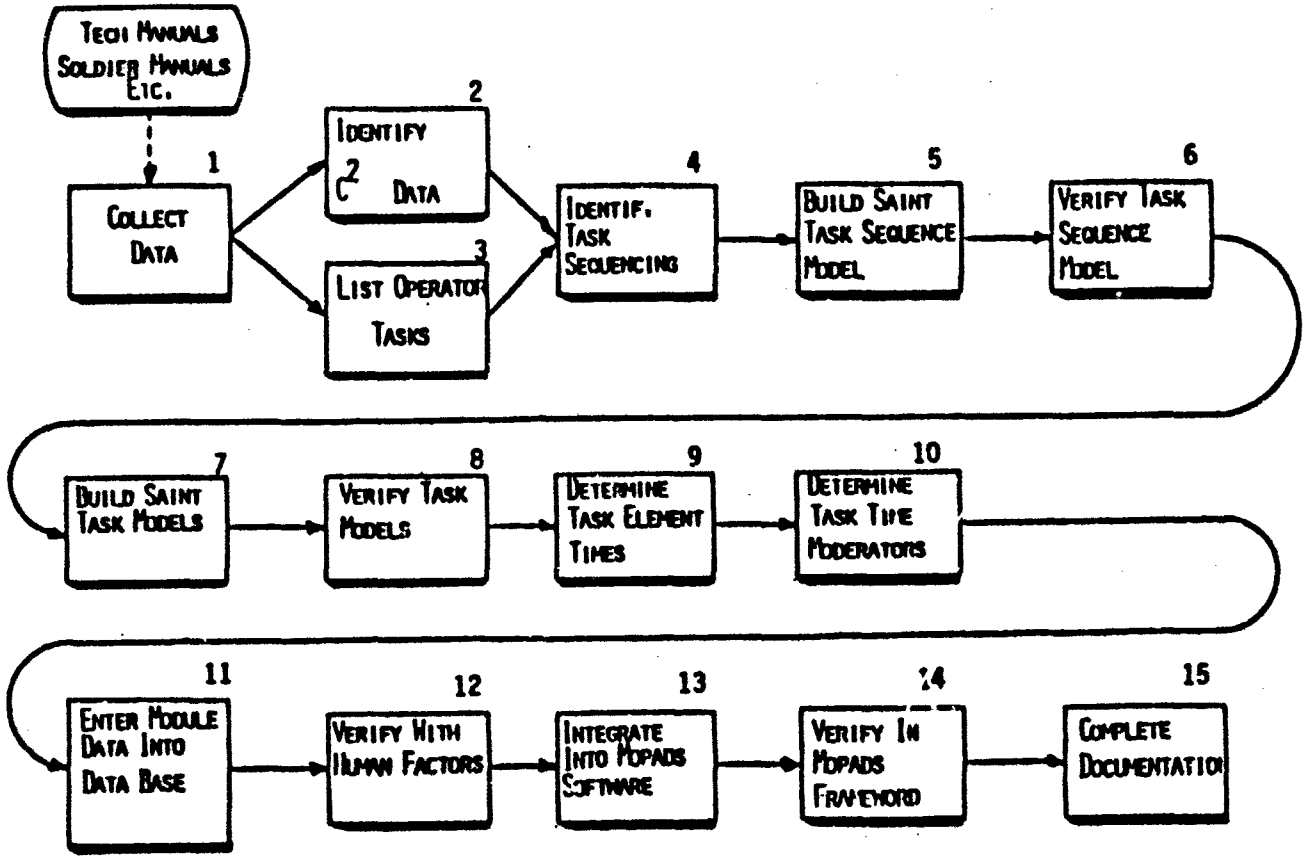


Figure III-1. Development of System Modules.

Figure III-2. Table of Contents for MOPADS System Module Documentation.

2-0 THE AN/TSQ-73 SYSTEM MODULE

2-1. AN/TSQ-73 Operator Tasks.

Table III-1 contains the operator tasks that are represented in the AN/TSQ-73 system module. An important part of developing MOPADS operator models is to determine which operator tasks are required to be represented. Many tasks are irrelevant to the air battle scenario. These include, for example, tasks related to the set-up, take-down, and transportation of the equipment. Furthermore, there may be duplication of task information among the task descriptions in official system documentation. Also, occasionaly it may be necessary to add a task to account for standard or common procedures that are not explicitly discussed in the official documentation.

Development of the task lists is a substantial activity that requires the MOPADS modeler to become familiar with all of the tasks in the official documentation so that informed decisions can be made in selecting those that must be modeled. Official documentation for the AN/TSQ-73 (U. S. Army Technical Manual, TM9-1430-652-10-3), Change 6, 1981) contains flow charts for the operator tasks. An example for hooking is shown in Figure III-3. These or similar representations must be obtained from the pertinent Army documentation and a preliminary task list determined. This activity is step 3 of Figure III-1. Since MOPADS involves models of combat operations, the operator tasks describing set-up and routine maintenance of equipment can usually be excluded at this point from further consideration.

For systems that have more than one operator, it may be necessary to develop task lists for each operator. An example of this case will be seen in Section 3-0 for the IHAWK system module. For the AN/TSQ-73, all tasks can be performed from each console, although it is customary for the operators to perform separate tasks based on their authority. For the AN/TSQ-73, this task split is represented structurally in the models rather than by developing separate task lists for each operator.

An important objective in developing the task list is to select the minimum set necessary to represent the operator interaction with the system to the required detail. Considerable effort is expended in expanding each operator task into an MSAINT representation. Therefore, the task list must be examined with a critical eye to ensure that unnecessary effort is not expended. As an example, consider the AN/TSQ-73. The usual operator configuration is to have a Tactical Director (TD) who has authority to order engagements and a Tactical

Table III-1. AN/TSQ-73 Operator Tasks Represented in MOPADS.

DESCRIPTION
Idle Time (Scan the Displays)
Cancel Secondary Assignment
Send Terminate Commands
Clear Hold Fire, Effective, Status
Perform Hooking Procedure
Enter ID and IFF Data
Interrogate a Target or Sector
Send Command Message
Assign Weapons/Battalions
Receive Commands
Clear Alerts
Receive Miscellaneous Messages

Director Assistant (TDA) who does not have such authority. The TDA will perform tracking and identification tasks, and the TD will order and monitor engagements. It is possible, however, to configurate the system with two operators that each have engagement authority (two "TD's"). In this case, each operator can perform all tasks. The modeler must decide which configuration(s) will be included in the model and specify task lists accordingly.

2-2. AN/TSQ-73 Messages.

Step 2 in Figure III-1 is to "Identify C^2 Data." Essentially, this step involves determining how operators of this equipment communicate with superior, subordinate, and lateral units. The communications occur through voice and data link messages.

MOPADS models of air defense systems (components) mimic this structure by communicating with messages sent through the MOPADS data base. It is necessary for the MOPADS modeler to explicity identify all communications between the air defense system being modeled and all other systems already modeled. For the current implementation, this means that the communication between the group and battalion AN/TSQ-73 and the IHAWK battery must be identified. Once again, a judicious elimination of messages that are not needed in the MOPADS context will greatly reduce later work.

The messages used in the current implementation are shown in Table III-2. For each such message, the sender, receiver, message characteristics, and message contents must be specified. A form has been developed to facilitate specification and documentation of this data. An example is shown in Figure III-4.

2-3. AN/TSQ-73 Operator Goals.

Once all of the messages and tasks for the operator have been determined, initial development of the task sequencing procedures for the operators can begin. The goals for the AN/TSQ-73 operators have been shown already in Table II-3.

The particular parameters used for each goal priority function are specified in Goodin & Polito (1983b).

2-4. AN/TSQ-73 Operator Task Models.

A SAINT task node model analogous to Figure II-2 has been developed for each of the operator tasks shown in Table III-1. This procedure involves defining the "entities" that will flow through the networks. In this case, the entities are operators. Each entity has a list of characteristics called attributes that identify it. Each of the operators is defined on a form as in Figure III-5. The attributes of the operators have been defined. These attributes are used in branching decisions to determine which Task Node will be performed next and in FORTRAN programs written by the MOPADS modeler to represent the operators' actions.

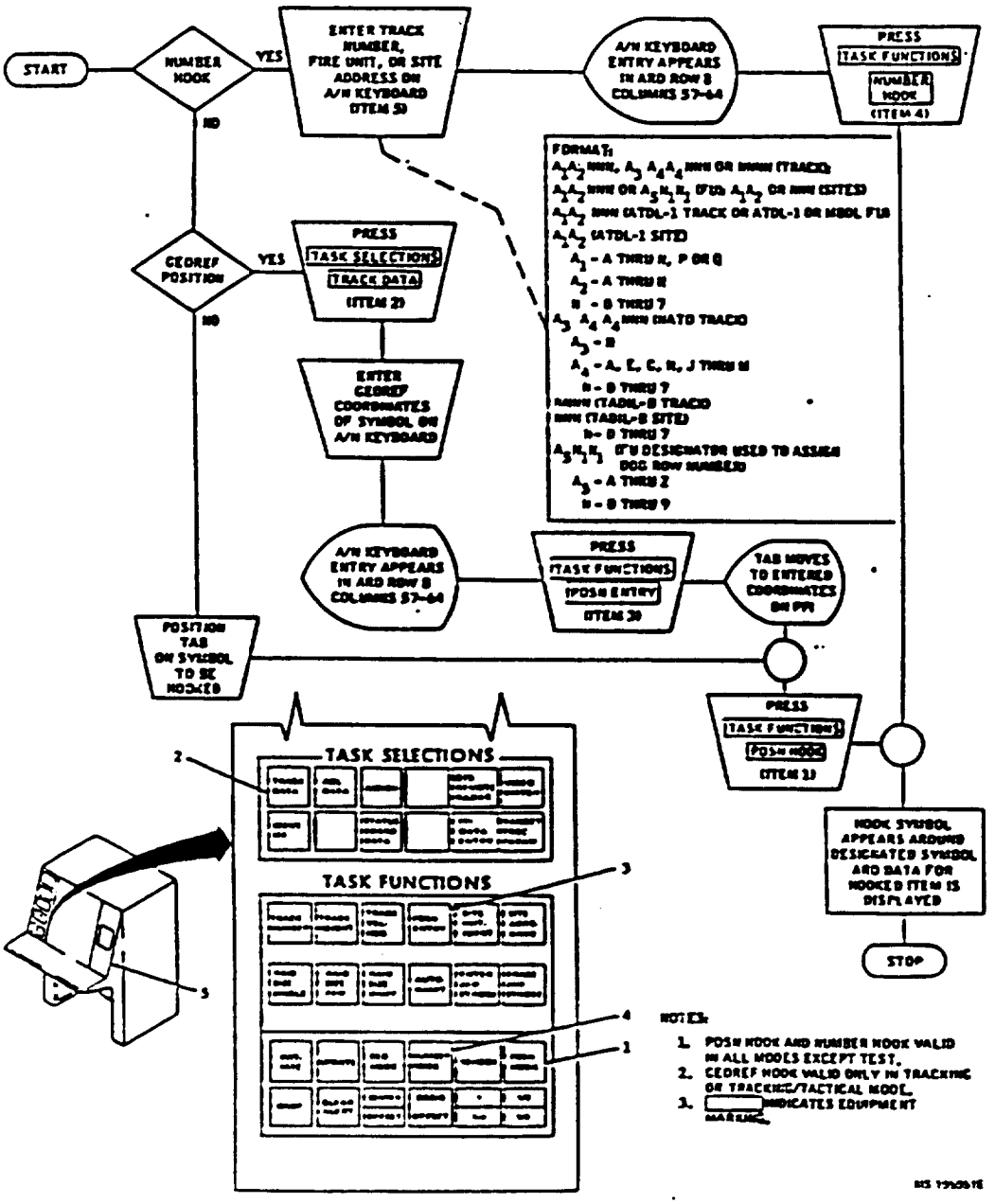


Figure III-3. Example AN/TSQ-73 Operator Task Flow Chart.

Table III-2. Messages for the AN/TSQ-73 and IHAWK.

Command Messages		
1	Hold Fire	
2	Cease Fire	
3	Cease Engagement	
4	Engage	
5	Cover	On track already assigned
6	Engage Ripple	
7	Engage New Track Assignment	
8	Cover New Track Assignment	
9	Engage Ripple Assignment	
10	Investigate/Assign Assignment	
11	Cancel Alert	
12	Track Assignment Status	
13	Change Targets	
14	Method of Fire	
15	Order No Kill	
16	Order Break Lock	
Request for Information Messages		
1	Request Cancel of TCC Alert	
Reporting Status Messages		
1	FU Out of Action	
2	FU Expended Hot Missiles	
3	FU Effective/Not Effective	
4	FU Engaging Pop-up Target	
5	New ASD Target	
6	IHIPIR Lock Status	
7	Raid Size Report	
8	Temporary No Kill	
Acknowledgement Messages		
1	Will Comply	Battalion O-73 to Group
2	Have Complied	
3	Can't Comply	
4	Will Comply	IHAWK to Battalion O-73
5	Can't Comply	
6	Acknowledge ASD Target	
7	Accept ASD Target	

MOPADS MESSAGE DESCRIPTION

MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	—
2	Operator Type	1 or 3
3	Functional Type	1
4	Message Subtype	8
5	Message Priority	—

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	1
3	Unused	—
4	ATDL Code (Unused)	—
5	* Time Message Sent	—
6	* Message Number	—
7	* Sender CRN	—
8	Sender Operator Type	1
9	Sender System Module Type	2 or 3
10	Task Mode Number Sent From	—

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# Words = 2
2	Which Fire Section = 0 Either = 1 A = 2 B
3	Track Column Number

MESSAGE SUBTYPE DESCRIPTION

Cover new target command. Obtain a HIPIR lock on a new target but do not fire.

Name: Jack Walker
Date: 8/3/83

System Module: Q-73
Project: MOPADS

From	To	From	To
GRP(15)	BN(20)	GRP(15)	HAWK(33)
BN(15)	HAWK(33)		

Figure III-4. Example Message Sent From the AN/TSQ-73.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
31(BN) 31 32 (GRP)	TD-Officer for handling all engagement and FU assignment tasks (operator types 1(BN), 8(GRP), and 9(GRP))	1	Operator ID	-	
		2	Copy Row Number	-	
		3	Operator Type (1-TD, 2-TDA)	-	
		4	Current Track Column Pointer (Ø if none)	-	
32(BN)	TDA-Handles tracking and identification tasks (Operator type = 2)	5	Internal Mark Time (For Op.Task Time Statistics)	0.0	
		6	Self-Clearing Indicator (Ø-No Self Clearing >Ø-Node to Clear to)	0	
	NOTE: Both TD's and TDA's have the same IA definitions	7	Last Task Node Branched from (set and used only at release times)	-	
		8	Hooking Indicator (Ø-None,Branch to hooking, >Ø-Task Node to branch back to)	0	
		9	Item to be hooked (<Ø site or FU; =CRN; =Ø No hooking;>Ø Track column pointer)		

Name: Riley Goodin
Date: 11/15/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

SAINT ENTITIES

Page 1 of 2

Figure III-5. Example SAINT Operator Definition Form.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
		10-12	Unused currently	-	
		13-14	Branching (Task Node Specific uses)	-	
		15	Information passed in or out of task sequencing (operator task specific uses)		

Name: Riley Goodin
Date: 11/15/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

SAINT ENTITIES

Page 2 of 2

Figure III-5. (continued)

When operator characteristics are specified, the task networks like Figure II-2 are expanded to describe each Task Node. Figures III-6, 7, 8, and 9 are examples. On these forms are specified the skill and skill weight requirements (see Table II-1), the task related variable data (see Table II-2), and the mean, standard deviation and distribution type of the nominal performance time. Any resource requirements are also specified.

The MOPADS operator task number is also given, so a cross-reference is possible between SAINT task nodes and operator tasks. Given a task node number, it is possible to find the operator task that contains it by looking at the forms in Figures III-6 to 9. Conversely, all of the task nodes that make up an operator task model can be found in the forms shown in Figure II-2.

Specification and verification of all of the data in Figures III-5 through III-9 completes steps 7 to 10 in Figure III-1. This leaves step 11 which involves entering data interactively into the MOPADS data base to support the system module. SAINT task models have been developed for each operator task, and nominal task performance times and skill requirements have been determined for each task node.

3-0 THE IHAWK SYSTEM MODULE

3-1. IHAWK Operator Tasks

The IHAWK operator tasks that are represented in MOPADS are shown in Table III-3. A separate task list has been prepared for each IHAWK operator, because, in contrast to the AN/TSQ-73, the various operators perform substantially different functions. By far, the most complex activities are performed by the Tactical Control Officer (TCO). All communications to other air defense components are represented as passing to and from the TCO.

The other operators perform relatively straightforward tasks. In other words, the task sequencing for these operators is relatively simple. Their activities are primarily rule-based.

The process of developing the IHAWK task lists was analogous to that for the AN/TSQ-73. It was somewhat more difficult to develop the lists and network models, however, because the Army documentation (U. S. Army Technical Manual, TM9-1430-1526-12-1, June 30, 1979) does not contain task flow charts analogous to Figure III-3. The activities are described in text that needed to be examined and put into network form.

TASK NODE: 45

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
45	CLRSTS	4(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: HOPADS

TASK NODE SPECIFIC DATA — Page 1 of 1

Figure III-6. Example SAINT Task Description for AN/TSQ-73.(Node 45)

TASK NODE: 47

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
47	POSHOOK	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

Figure III-7. Example SAINT Task Description for AN/TSQ-73. (Node 47)

TASK NODE: 53

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
53 (BN)	IKROUT3	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

Figure III-8. Example SAINT Task Description for AN/TSQ-73. (Node 53)

TASK NODE: 71

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
71	PROACH	8(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

Figure III-9. Example SAINT Task Node Description Form for AN/TSQ-73.(Node 71)

Table III-3. IHAWK Operator Tasks Represented in MOPADS.

Description
ASO TASKS
ASO Standby, Wait for Action
Detect New Target at CWTBC
Establish Target Priority
Preempt Lower Priority Target
TCC Alert
Mark Target as Accepted by TCC
FCO A FCO B TASKS
FCO Standby, Wait for Action
Track Target
Obtain Lock on Target Manually
Put Fire Section out of Action
Estimate Raid Size
Select Launcher
Fire Missiles
Evaluate Target Intercept
Process Change Targets
Put Fire Section back in Operation
TCA TASKS
TCA Standby, Wait for Action
Accept CWTBC Target From ASO
IFF Challenge
Mark on Reflection Plotter (Friend, Hostile, Unknown)
TCO TASKS
TCO Standby, Wait for Action
Detect IPAR ADP Target
Manually Assign Targets
IHIPIR Tracking
TCO-IHIPIR Does not Acquire Lock
Higher Priority Target to be Assigned to Firing Section
Process a Hostile Target
Assigned Target Determined Friendly
Process Friend
Evaluate Whether More Missiles Are To Be Fired
Give ASO Permission to Cancel Alert

Table III-3. (continued)

TCO MESSAGE TASKS
Accept Q-73 Target (ENGAGE MODE)
Accept Q-73 Target (COVER MODE)
Accept Change Targets Command
Receive "HOLD FIRE" Command
Receive "CEASE FIRE" Command
Receive "CEASE ENGAGE" Command
Issue "PRIORITY, PRIORITY" CALL TO Q-73
Receive "ROGER ENGAGE" Command
Send Cannot Comply Message to Q-73

3-2. IHAWK Messages.

The messages used in the current implementation were shown in Table III-2. This table includes IHAWK messages. MOPADS uses the message concept to represent voice communication within a component as well as data link and voice between units. This mechanism is, then, used in a uniform way to represent all direct communication between individuals in the MOPADS models. Figure III-10 is an example of a voice message specifying the method of fire which is sent between operators of an IHAWK unit. This particular message is sent by the TCO to the Fire Control Officer (FCO) to specify the method of fire for attacking a particular track.

In all other ways, specification of the messages for the IHAWK is analogous to the process discussed in Section III, 2-2.

3-3. IHAWK Operator Goals.

The IHAWK operators' goals were shown in Table II-3.

MOPADS MESSAGE DESCRIPTION

MESSAGE ID ELEMENT Page 1 of 1

Element	Description	Value
1	* Receiver CRN	—
2	Operator Type	6 or 7
3	Functional Type	1
4	Message Subtype	14
5	Message Priority	—

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	1
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	—
4	ATDL Code (Unused)	—
5	* Time Message Sent	—
6	* Message Number	—
7	* Sender CRN	—
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	—

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# Words = 2
2	Track Column Number
3	Method of Fire = 1 Shoot-Look-Shoot = 2 Ripple Fire

MESSAGE SUBTYPE DESCRIPTION

Method of Fire Command. Shoot-Look-Shoot means shoot one then evaluate before shooting another. Ripple means shoot two missiles.

Name: Jack Walker Date: 8/4/83	System Module: IHAWK Project: MOPADS

From	To	From	To
TCO(27)	FCO		

Figure III-10. Example IHAWK Message.

3-4. IHAWK Operator Task Models.

Task models for the IHAWK have been developed in the same way as those for the AN/TSQ-73. Figure III-11 is the operator definition for the TCA. Each of the operators, tasks, and task nodes has been specified in the same way as for the AN/TSQ-73.

All of the careful data collection and model development information for the AN/TSQ-73 and IHAWK system modules have been delivered to the Army in four volumes (Goodin & Polito (1983a,b; Goodin & Walker, 1983a,b).

4-0 AIR SCENARIOS

In addition to representing the air defense system, MOPADS must have a model of the air battle that the air defense system fights. The MOPADS software has facilities to represent the salient features of the air battle environment.

The data required for the air scenario representation are the following.

1. The coordinate system reference point.
2. The locations of all air defense units and protected sites (critical assets).
3. The assignments of critical assets to the fire units who are to protect them.
4. The characteristics of each radar or observer that acquires information about aircraft.
5. The characteristics and flight paths of all aircraft.

4-1. The Coordinate System.

MOPADS assumes a flat earth and uses rectangular coordinates. The coordinates, (x, y, z), of all points in the system are given with respect to a user specified reference point. This point may be specified as a longitude and latitude if a specific terrain system is to be specified. Once the reference point is chosen, all other coordinates are specified with reference to it. The units of the x and y coordinate are nautical miles, and the units of z are feet. The +x axis is east, the +y axis is north, and +z is up.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
46	Tactical Control Assistant	1	Operator ID	-	-
		2	Copy Row Number	-	
		3	Operator Type	4	
		4	Current Track Column Pointer	0	
		5	Internal Mark Time	0	
		6	Self-Clearing Indicator	0	
		7	Last Task Node Branched From	0	
		8	Unused	0	
		9	IFF Mode (1=Manual,2=Auto) Not currently used (All IFF is manual)	1.0	
		10	Unused	0.0	
		11	Unused	0.0	
		12	Unused	0.0	
		13, 14, 15	Branching	0.0	

Name: Riley Goodin
Date: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT ENTITIES

Page 1 of 1

Figure III-11. Example IHAWK Operator Definition Form.

4-2. Locations of Air Defense Units, Critical Assets, and Asset-Fire Unit Assignments.

Figure III-12 shows an example layout of the reference point and of protected assets. With this basic configuration, a variety of air defense configurations and air battles can be simulated. One possible configuration is shown in Figure III-13. This figure shows a group AN/TSQ-73 with two battalions. The battalions each have two IHAWK fire units. The critical assets are assigned to fire units as follows: IHAWK 1 protects site 1, IHAWK 2 protects site 2, and IHAWKS 3 and 4 both protect site 3. The circles around the units represent their area of radar coverage.

Many possible configurations could be arranged to protect the three critical assets, and MOPADS allows the MOPADS user to specify the location and assignments of the air defense components.

4-3. Characteristics of Viewers.

Each air defense unit may "own" one or more "viewers." Viewers are usually radars (and in the current implementation, they are always radars), but in the case of REDEYE or VULCAN, for example, the viewer might be a human observer.

Each viewer has the following characteristics.

1. maximum range
2. minimum and maximum altitude
3. probability of detection
4. barriers to view
5. a sector of interest

The characteristics above serve to restrict a viewer's ability to detect aircraft. The maximum range and altitude restrictions are self explanatory. The probability of detection is the probability that the viewer will detect an aircraft that is otherwise in its field of view. MOPADS assumes that once an aircraft is detected it remains detected so long as it is in the viewer's field of view.

The MOPADS user may specify barriers-to-view that block out part of a viewer's ability to detect aircraft. The barriers approximate terrain and other limitations that preclude a radar or observer from seeing everything within range. Figure III-14 shows how barriers may be specified. Two types of barriers may be specified: line barriers and wedges.

COORDINATE AND ASSET DATA

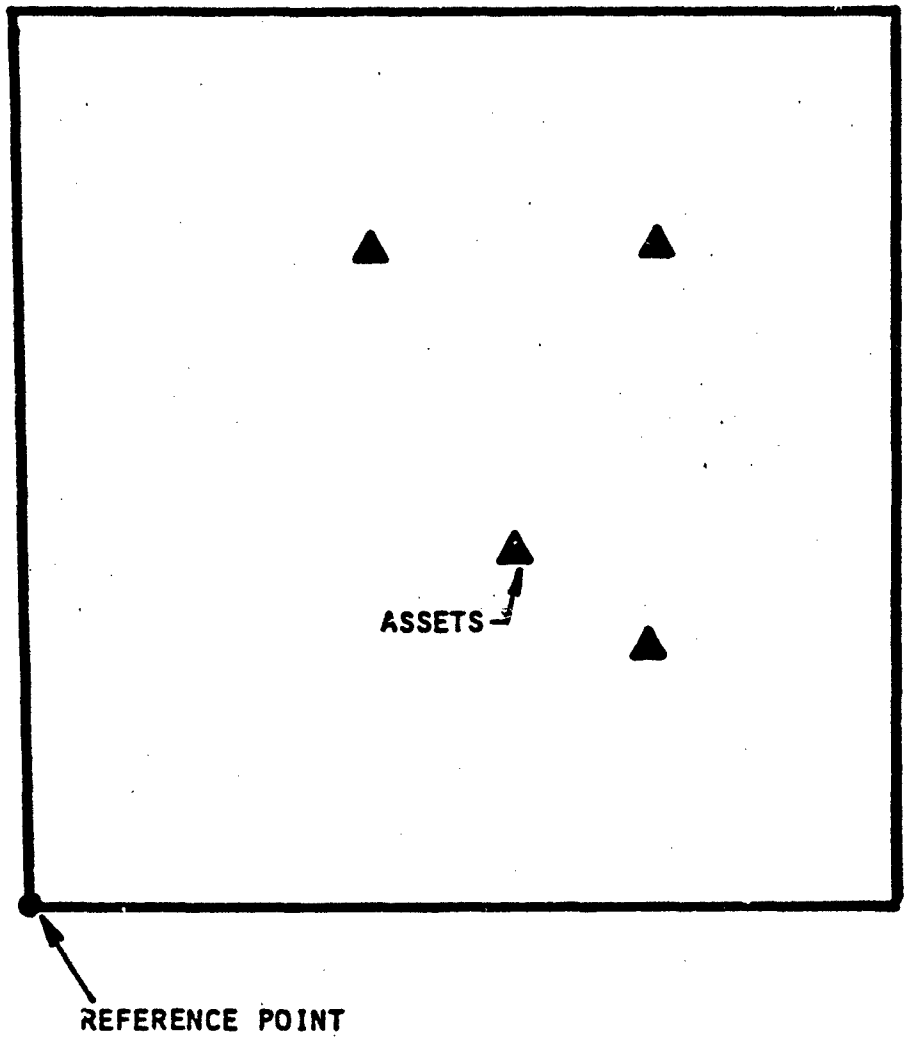
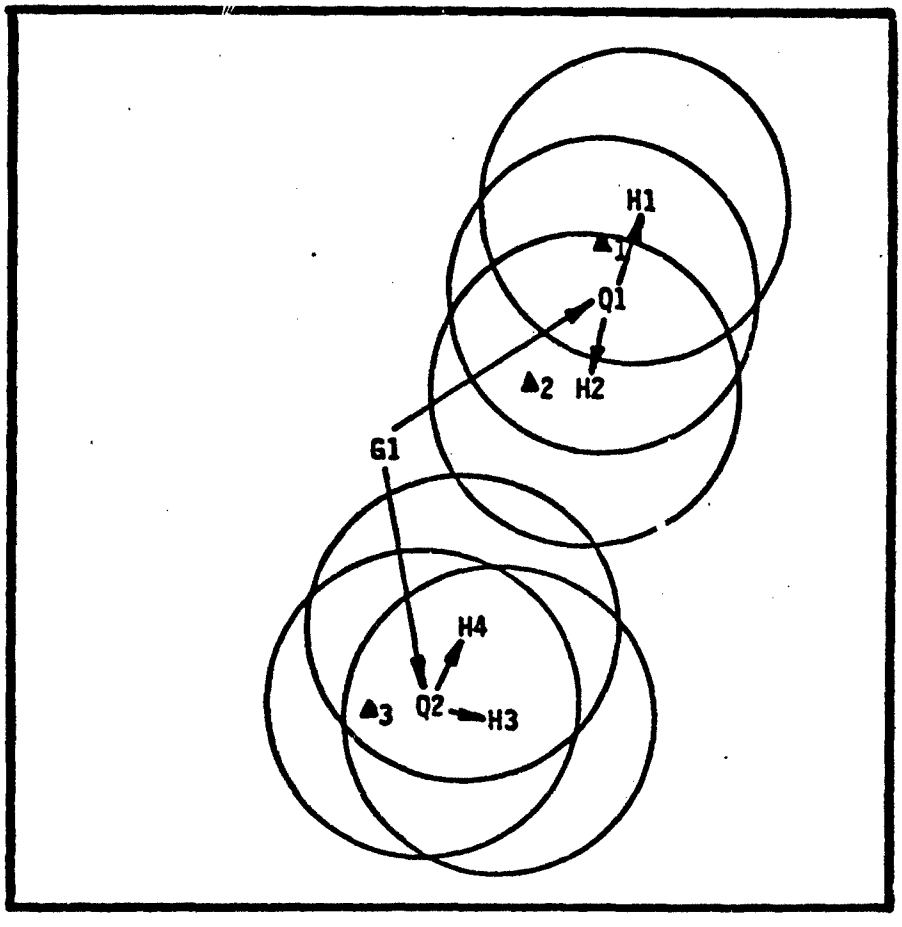


Figure III-12.Example Critical Asset Configuration.

AIR DEFENSE CONFIGURATION

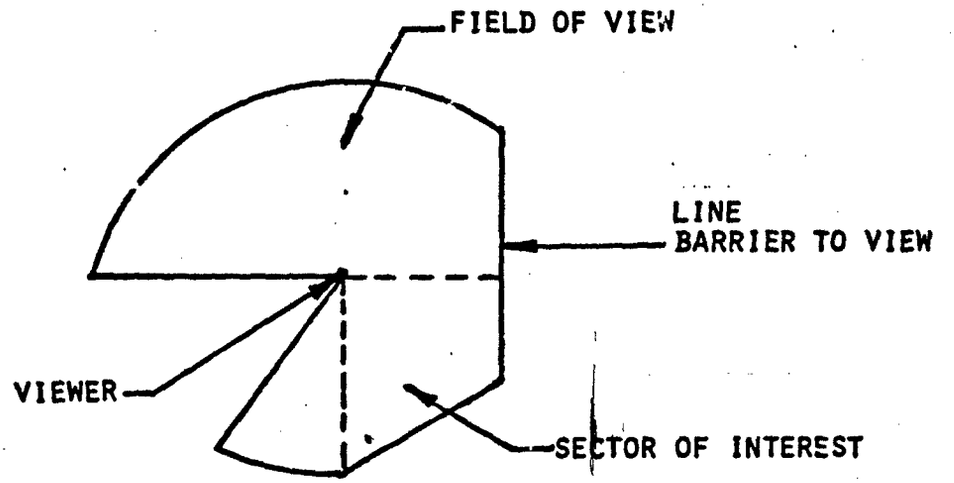
1 GROUP(G) 2 Q-73's(Q) 4 IHAWKS(H)



▲ - CRITICAL ASSET

Figure III-13. Example Air Defense Configuration

PLAN VIEW



SIDE VIEW

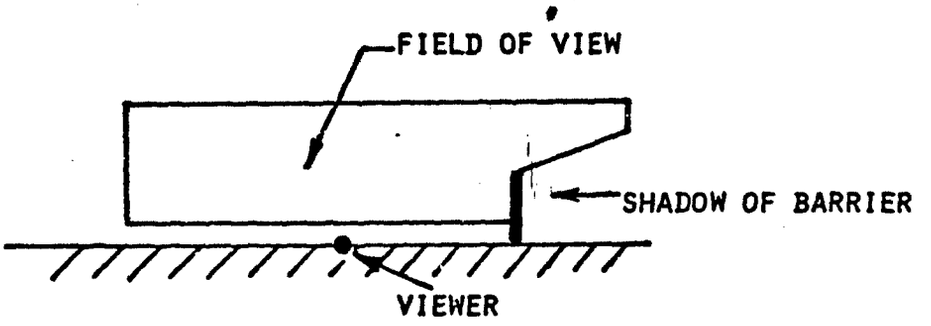


Figure III-14. Viewers and Barriers-to-View.

In the plan view of Figure III-14, two line barriers are shown to the east and southeast of the viewer. Everything farther away from the viewer than the line barrier is hidden from view if it is in the shadow of the barrier. The side view in Figure III-14 demonstrates this. The area to the east of the barrier and below the shadow is hidden from the viewer (note, the side view does not show a complete detail of the viewer in the plan view).

Line barriers may be positioned anywhere in the maximum range of the viewer and the end points may have different altitudes. MOPADS assumes a linear variation of altitude from one end of the barrier to the other. Using multiple line barriers, it is possible to create complex viewing areas that approximate actual radar viewing patterns.

Wedge barriers are simply pie shaped sections in which nothing is visible. An example is shown in Figure III-14 in the plan view to the west of the viewer. The user specifies the start and end compass headings of the wedge.

The system of restricting the field of view described above is not a perfect representation of radar vision limits, but it permits a reasonable approximation for modeling purposes.

The IHAWK permits the operators to specify a "sector of interest" which is a pie shaped segment of its viewing area. The IHAWK computer will automatically process tracks in this sector. The sector of interest has no effect on the radar's ability to acquire aircraft. It serves only to delineate a high interest area to the computer. MOPADS has a facility to specify a sector of interest for each viewer. In the current implementation, the sector of interest is used only by IHAWK.

4-4. Characteristics and Flight Paths of Aircraft.

Aircraft flight paths are represented as sequential piece-wise linear segments. The start and end coordinates for each segment are specified along with the speed of the aircraft on the segment. The x, y, and z velocities are computed by MOPADS and assumed constant along the segment. The MOPADS modeler may specify as many segments for an aircraft as desired, and as many aircraft as desired may be included in an air scenario. Curvilinear tracks may be approximated as sequences of linear segments.

The following data are specified for each aircraft:

a.	Category	A track may be "hostile," "friendly," or "other." An "other" track is a track that cannot be classified as friendly.
b.	Identifying Number	The user may specify a unique number for each track.
c.	Multiplicity	The number of aircraft in the flight.
d.	Aircraft Type	Code values for a variety of aircraft types are provided. See Table III-4.
e.	Whether The End Point Of The Segment Is A Target	(for hostile tracks only) - This item specifies that the end point of a flight segment is an air defense unit or a critical asset that the air defense system is attempting to protect.
f.	Probability of Destruction	(for hostile tracks only) - If the end of the segment is a target, the hostile track will attack it. This item is the probability that the site is destroyed.
g.	Jamming On A Segment	The user may specify that a hostile track is employing ECM on a segment. This information is currently not used by MOPADS. It is provided for future enhancements.

Table III-4. Aircraft Type Codes.

TARGET TYPE CODES

CODE NO.	NAME	COUNTRY	MISSION
1	F4C	USA	
2	F100	USA	
3	T33	USA	
4	OTHER JET		
5	U1A	USA	
6	U6A	USA	
7	OTHER PROP		
8	01A	USA	
9	OH23	USA	
10	OTHER HELICOPTER		
11	TANK		
12	JEEP		
13	TROOP		
14	APC		
15	TRUCK		
16	ZERO		
17	HALFTRACK		
18	F14	USA	
19	F15	USA	
20	F16	USA	
21	F18	USA	
22	MIG21	USSR	
23	MIG23	USSR	
24	MIG25	USSR	
25	SOLDIER(FOOT)	ANY	
26	MIG-27	USSR	
27	SU-17	USSR	
28	QIANG Ji-5	PRC	Ground Attack
29	R-2358	FRANCE	Military surveillance
30	MIRAGE 3E	FRANCE	Fighter
31	MIRAGE F1	FRANCE	Fighter
32	MIRAGE 2000	FRANCE	Fighter
33	MIRAGE 4000	FRANCE	Fighter
34	MIRAGE 4	FRANCE	Bomber
35	MIRAGE 5	FRANCE	Ground Support
36	MIRAGE 50	FRANCE	Fighter
37	AW.660	GREAT BRITAIN	Military Transport
38	698	GREAT BRITAIN	Bomber
39	HS748	GREAT BRITAIN	Military Transport
40	HS.780	GREAT BRITAIN	Military Transport
41	P.1099	GREAT BRITAIN	Ground Attack

Table III-4. (continued)

42	IAI202	ISRAEL	Military Transport
43	MIRAGE 3C	ISRAEL	Fighter
44	KfirC2	ISRAEL	Fighter
45	G.222	ITALY	Military Transport
46	F1045	ITALY	Interceptor
47	MB.326K	ITALY	Strike
48	S.M.1019E	ITALY	Military STOL
49	F-1	JAPAN	Fighter
50	C.207A	SPAIN	Military Transport
51	SF-5A	SPAIN	Fighter
52	HA-220	SPAIN	Ground Attack
53	35XD	SWEDEN	Ground Attack
54	JA37	SWEDEN	Fighter
55	J-1	YUGOSLAVIA	Strike
56	Light (e.g.blinking)		
57	Digit (digit on a display)		
58	MIG-17	USSR	
59	MIG-19	USSR	
60	SU-7	USSR	
61	SU-9PM	USSR	
62	SU-11	USSR	
63	SU-15	USSR	
64	SU-19	USSR	
65	SU-20	USSR	
66	YAK-28P	USSR	
67	YAK-36	USSR	
68	TU-28P	USSR	
69	IL-28	USSR	
70	M-4	USSR	
71	TU-16	USSR	
72	TU-20	USSR	
73	TU-22	USSR	
74	TU-26	USSR	
75	IL-38	USSR	
76	TU-126	USSR	
77	MIG-15	USSR	
78	L-29	USSR	
79	L-39	USSR	
80	J-5	USSR	
81	J-6	USSR	
82	J-7	USSR	
83	J-8	USSR	
84	H-5	USSR	
85	H-6	USSR	
86	CJ-6	USSR	
87	Y-11	USSR	
88	MIRAGE III	FRANCE	

The organization of scenario information in the data base is shown in Figure III-15. A directory is created called CRITICAL-ASSET-CONFIGURATION which contains the specification of the coordinate reference point and the location of all critical assets. This information is contained in an entry COORDINATE-AND-ASSET-DATA. Then one or more air scenarios (which consist of aircraft tracks) can be specified. Each air scenario contains records for hostile, friendly, and other tracks. Thus, for a single configuration, the MOPADS modeler can create a menu of air scenarios that attack it (see Figure III-16). The MOPADS user then selects from this menu, the scenario that he desires to simulate.

Similarly, more than one CRITICAL-ASSET-CONFIGURATION can be created in the data base, so the user also has a menu of such configurations to choose from. When a simulation is designed, the MOPADS user will select the CRITICAL-ASSET-CONFIGURATION and the air scenario within the asset configuration that he desires to simulate.

4-5. The Control System Module.

Every MOPADS simulation will automatically contain a system module that does not represent an air defense unit. This system module, called the Control System Module, is a SAINT model that drives the air scenario. This small SAINT model initiates the tracks at the proper time and simulates their flight paths to their termination points. It also performs statistics collection and information management on all track data.

The control system module also ends the simulation at the time specified by the user. The module consists of six SAINT task nodes and appropriate FORTRAN support programs.

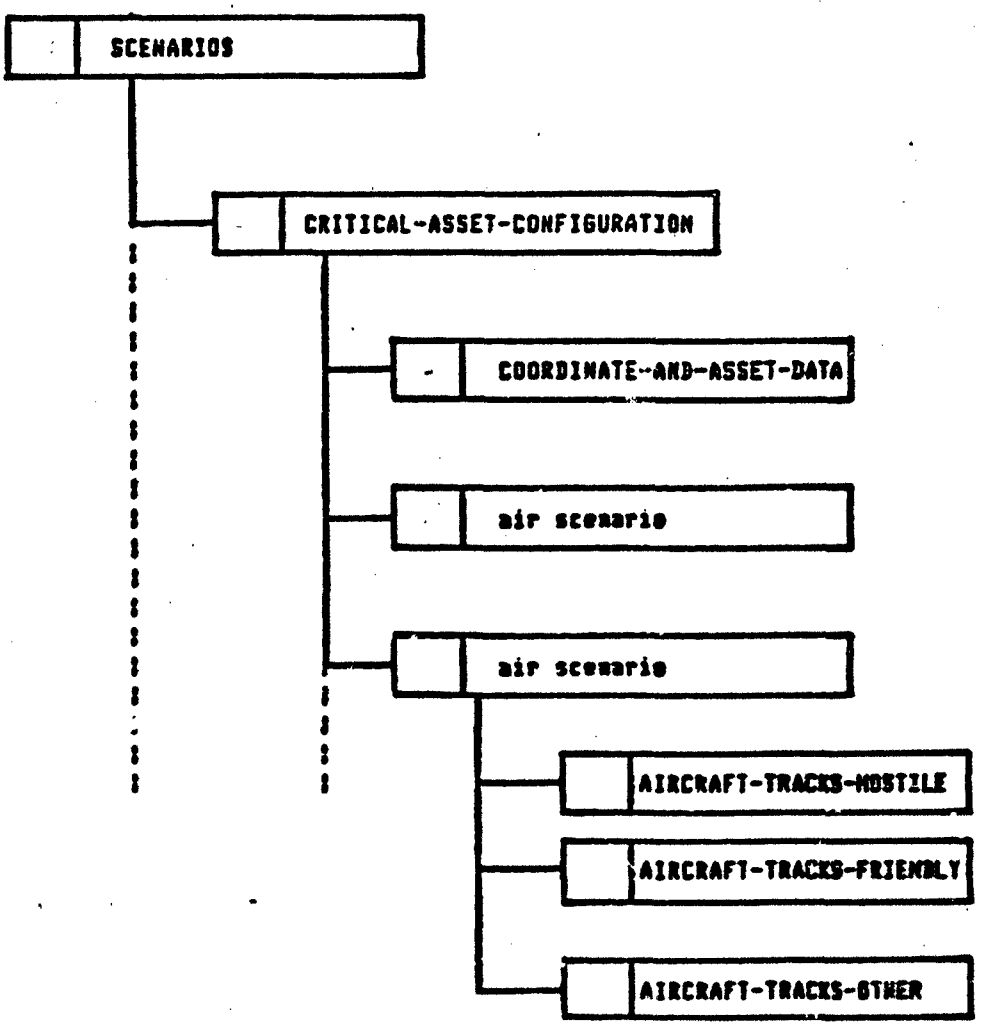


Figure III-15. Data Base Organization of Air Scenario Information.

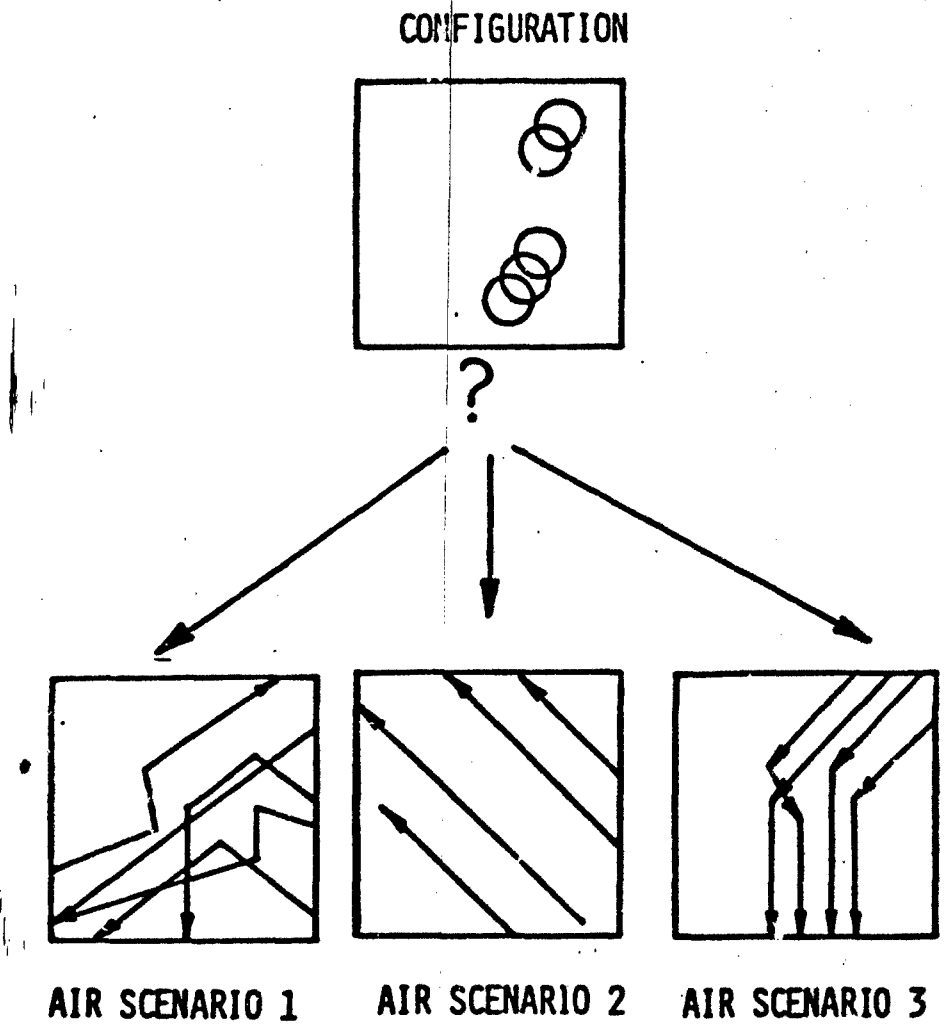


Figure III-16. Selection of Air Scenarios.

IV. MOPADS PROGRAM IMPLEMENTATION

1-0 MSAINT

The SAINT simulation language needed to be modified to accommodate the MOPADS requirement for "run-time configurable" air defense systems. In other words, it must be possible for the MOPADS user to specify the number of air defense units to be simulated and their hierarchical structure. As has been discussed, the logical issues related to this requirement were resolved by modeling the intercommunication among components as messages which are sent through the MOPADS data base. At the computer programming level, however, several issues still needed to be addressed.

Since the MOPADS modeler cannot know in advance how many copies of, say, the IHAWK system module may be required by a MOPADS user, a software scheme is needed to replicate the system modules an arbitrary number of times. SAINT task nodes are numbered (e.g., 1, 2, 3...), so that merely duplicating the task networks would result in duplicate node numbers which are unacceptable to the SAINT processor. The most straightforward approach to this problem is to develop a preprocessor that would renumber the nodes as the network duplicates are created. On the surface, this appears to be an acceptable solution, but it has several drawbacks:

a. Large amounts of computer memory are used for network storage. Replicating essentially identical information will require large computing resources to run MOPADS models.

b. Technical problems result in matching data base information (much of which is keyed to individual nodes) to the (now unpredictable) node number of replicated networks.

c. Complex cross-indexing schemes would have to be developed for the FORTRAN insert programs that are an integral part of each system module so that it would be possible to know which copy was being processed at any given time.

d. Problems (b) and (c) above are compounded when more than one system module type are included in the network.

Using the approach just described would have required a complex programming activity, complex indexing designs for the data base, and complex programming in the FORTRAN inserts. Also, it would have heavily taxed the computing resources.

An alternative scheme is to modify SAINT to "share" a single network representation among multiple realizations of "copies" of the network (Polito & Walker (1982)). This saves substantial computing resources and allows the same node numbers to be used for each copy. This approach has been implemented and, in fact, the node numbers for each system module may be selected independently of the node numbers of all other system modules. The programming to perform this task was, of course, complex, but this task will be performed only one time, and it greatly simplified the design and programming of virtually all other software modules. In particular, the development of air defense system modules is made much simplier because:

a. node numbers may be selected freely without knowledge of node numbers used in other system modules,

b. FORTRAN insert programs do not have to cross index node numbers to determine the copy, and

c. complex cross-indexing of data base information is not required.

This scheme represents the multiple copies of system modules in an elegant manner that simplifies model development and substantially reduces requirements for computing resources.

2-0 MOPADS DATA BASE

The MOPADS software must maintain a great deal of information about a variety of subjects:

a. data which describes the Air Scenario to be simulated,

b. data which describes the tactical scenario which includes location and characteristics of air defense units, the command and control system, and the coordinate reference system,

c. data which describes the characteristics of the operators of air defense systems and the environment in which they function,

d. data which describes the dynamic relationships of operator tasks, and

e. data which represents statistics collected during simulations.

All of this information must be maintained in an easily accessed and edited form, and it must be addressed in a way that permits multiple data sets to co-exist without confusion.

The most effective way to accomplish this is with a unified data management system or a data base. The Data Base Control System (DBCS) module of the MOPADS software performs this function.

The DBCS is a non-traditional data base manager. However, the organization of the MOPADS data base file most nearly resembles a hierarchical data base. It is non-traditional in the sense that rapid access of datum elements is needed during MOPADS simulations. The DBCS utilizes a data address that is passed into and out of the DBCS which permits rapid access of required data. The address precludes most hierarchical searches in the data base file to locate data; thus, it eliminates many disk accesses. Furthermore, the DBCS dynamically retains the most frequently accessed data (not merely the most recently accessed) in main memory, so disk reads are minimized. These features make the DBCS a reasonably fast tool for storage and retrieval of data.

The DBCS is a collection of subprograms which any application program may use to create and manipulate a data base file. The DBCS has no main program, so it has no stand-alone capability. It was designed to provide rapid access to data for the MOPADS system. The DBCS can be used in settings other than MOPADS, because it has no built in structure that is unique to MOPADS. It does not, however, have many traditional data base features because of its specialized target environment. It also imposses a somewhat greater burden on application programs than other similar data base systems. The DBCS requires the application program to remember data base address information which is used by the DBCS to implement fast data access.

The DBCS provides capabilities to perform the following functions on data base files:

1. Open/Close DB Files
2. Add/Delete/Rename Directories
3. Add/Delete/Extend/Shorten Data Lists
4. Read/Write Data Lists
5. Set/Change the Data Base Protection Mode
6. Search Directories for Particular Data Lists or Directories.
7. Set/Access/Change Labels of Data Lists and Data List Elements
8. Read/Write External Format Data Files for Portability of Data Base Files
9. Set/Access Various DBCS Options
10. Print Contents of Data Lists and Directories.

In order to reduce the number of disk accesses, the DBCS computes an access frequency for each record, and then keeps the most frequently accessed records in core. Since patterns of record accessing may change over the life of a data base, an exponentially smoothed access frequency (SAF) is calculated and used as the basis for core residence decisions.

Every record in main memory (whether it has been resident for some time or has just been read) has a SAF value which is approximately comparable. In other words, it represents the smoothed access frequency of the record based upon nearly the same criteria. The DBCS uses these values to determine which records will have a tendency to be retained in main memory.

A set of data base application programs (DBAP) have been written to implement the particular data base used by MOPADS. As mentioned earlier, DBCS is generic and has no structural elements that reflect the details of the contents of the MOPADS data base. The DBAP, however, is specialized to MOPADS and utilizes the DBCS to manipulate and manage the data. The DBAP provides many high level data base access functions for the other MOPADS software modules. A schematic of the software structure is shown in Figure IV-1. In particular, DBAP programs are provided for the Human Factors Moderator Functions to access operator characteristics, so the moderator functions do not need to "know" structural information of how MOPADS stores data.

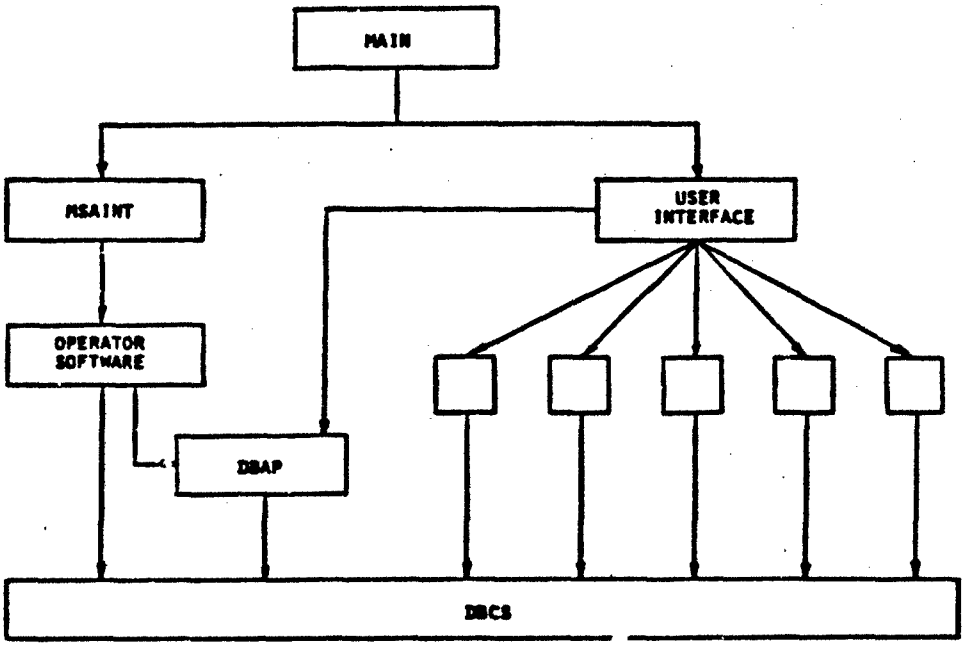


Figure IV-1. DBCS and MOPADS.

Examples of DMB programs are:

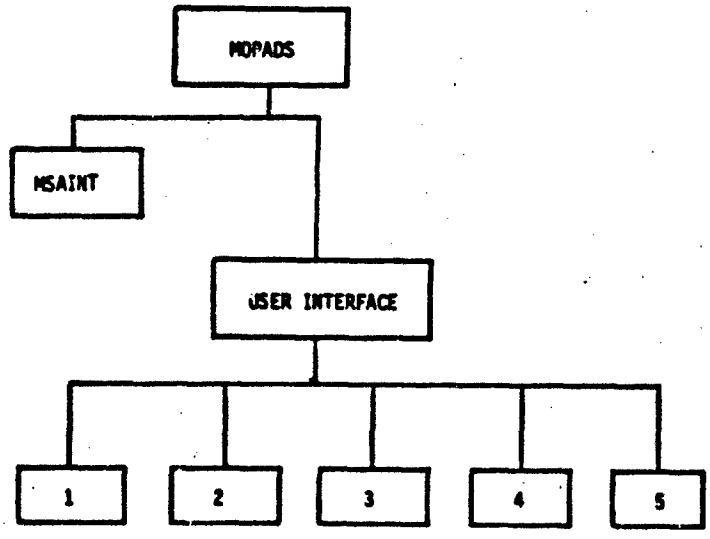
1. Create a MOPADS data base on a new data base file.
2. Copy portions of one MOPADS data base to another.
3. Store/retrieve elements of an operator's state vector.
4. Generate MOPADS error messages related to the data base.
5. Locate all air defense components in a specified simulation data set.

The DBCS is documented in Polito (1983d). The DBAP is documented in Polito (1983e).

3-0 THE MOPADS USER INTERFACE

3-1. Organization of the MOPADS User Interface.

The MOPADS user interface has five subprocesses which are shown schematically in Figure IV-2. Each of the subprocesses provides facilities for using and modifying the MOPADS data base.



Subprocesses:

1 - Create Simulation Data Set
2 - Set Up Simulation Run Data
3 - Playback Statistics
4 - Create/Edit Air Scenario
5 - Create/Edit Reference System Module

Figure IV-2. Structure of the MOPADS User Interface.

Three of the subprocesses are for use primarily by the MOPADS user. These are the first three: Create Simulation Data Set, Set Up Simulation Run Data, and Examine Statistics. The remaining two subprocesses, Create/Edit Air Scenario and Create/Edit Reference System Module, are primarily for the use of the MOPADS modeler. The functions of the subprocesses are:

1. Create Simulation Data Set - This subprocess allows the MOPADS user to specify a command and control configuration with individually edited copies of reference air defense system modules.

2. Set Up Simulation Run Data - This subprocess allows the MOPADS user to specify data for a MOPADS simulation. The user specifies a simulation data set (previously constructed with subprocess one above), selects the air scenario which will be used, and assigns critical assets to air defense units to be defended.

3. Examine Statistics - This subprocess allows the user to review and selectively print outputs from a MOPADS simulation.

4. Create/Edit Air Scenario - This subprocess provides facilities for the MOPADS modeler to create new air scenarios for the MOPADS user to simulate.

5. Create/Edit Reference System Module - This subprocess provides facilities for the MOPADS modeler to create new system modules for the use of MOPADS users. This process allows the menu of available air defense system modules to be expanded.

Each subprocess is an interactive, command-driven processor.

3-2. Create Simulation Data Set.

Using this subprocess, the MOPADS user can create an air defense configuration that may be simulated. Figure IV-3 shows a schematic of the data base operations performed in this subprocess.

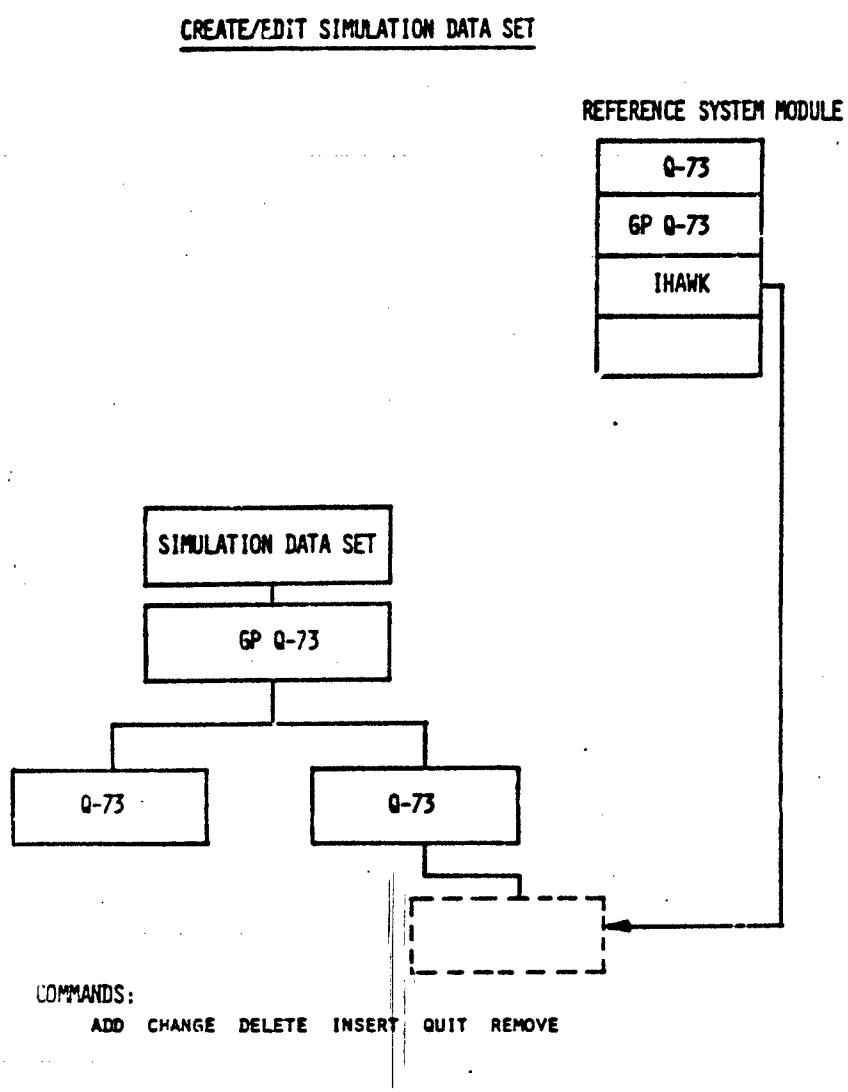


Figure IV-3. Data Base Functions of the "Create Simulation Data Set" Subprocess.

The menu of reference system modules is created by MOPADS modelers using the Create/Edit Reference System Module process. The result is a set of complete system specifications for an AN/TSQ-73, IHAWK, etc. The reference system module contains default values for all human factors and system parameters.

When using the Create Simulation Data Set, the user issues commands that cause copies of the reference system module to be placed in the command and control structure devised by the user. Thus more than one IHAWK or AN/TSQ-73 may be represented in the command and control structure. Each of these copies is called a working air defense system module. The working system modules can be individually edited to reflect differences in human factors parameters or system options.

The commands unique to this subprocess are shown below:

ADD - Create a new simulation data set

CHANGE - Edit the parameters of a working system module

DELETE - Delete an entire simulation data set and all of its working system modules

INSERT - Copy a reference system module to a specified position in the simulation data set

REMOVE - Delete a specified working system module and all of its subordinate units

QUIT - Leave this subprocess of the user interface

Multiple simulation data sets can be created and stored simultaneously in the MOPADS data base. This provides a capability to create a menu of air defense configurations that can be simulated.

3-3. Set Up Simulation Run Data.

With this subprocess, the user completes the information needed to perform a simulation. Figure IV-4 shows the data basic operations. An entry called a Tactical Scenario is created that belongs to a particular Simulation Data Set. The user selects a particular critical asset configuration and air scenario for that asset configuration. Furthermore, he assigns critical assets to fire units for defense and specifies certain technical parameters such as the start and end times of the simulation. Once this information is stored in the data base, it is necessary only to specify the simulation data set and the tactical scenario identifiers to MOPADS in order to perform a simulation.

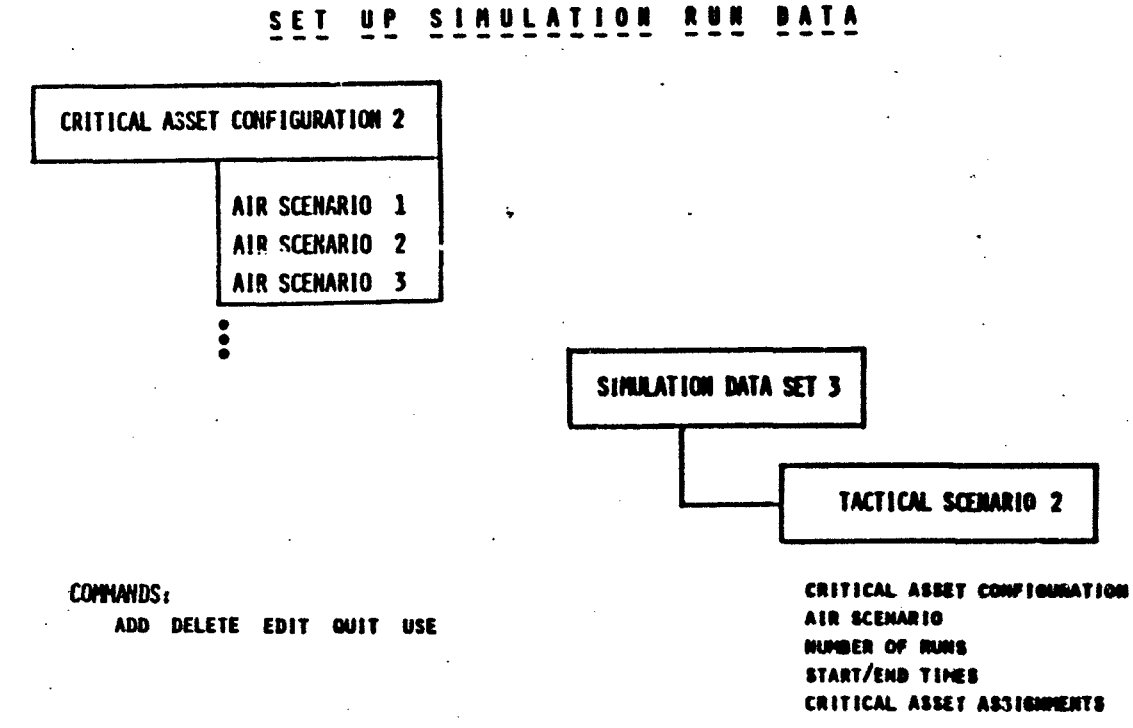


Figure IV-4. Data Base Functions of the "Set Up Simulation Run Data" Subprocess.

The commands unique to this subprocess are:

ADD - Create a Tactical Scenario entry

DELETE - Delete a Tactical Scenario entry

EDIT - Set/revise parameters specified in the Tactical Scenario entry

USE - Select a Simulation Data Set

QUIT - Leave this subprocess

More than one Tactical Scenario entry may exist in one Simulation Data Set. Also, the statistics produced by a simulation are stored in the Tactical Scenario entry, so the output of a simulation is always stored with the data specification that produced it.

3-4. Examine Statistics.

This subprocess is used by MOPADS users to preview and print the results of MOPADS simulations. MSAINT writes row statistical data to the data base after each simulation run. It is physically stored in entries of the Tactical Scenarios. This subprocess has commands that permit the user to selectively print statistics. For example, it is possible to print task node statistics for a particular IHAWK or for all IHAWKs. Similarly, the user can print all operator statistics for one Q-73 or operator statistics for one type (e.g., TD) for all Q-73's.

The commands unique to this subprocess are:

DISPLAY - Print the labels of all Operators and all Working System Modules.

PRINT - Print statistics.

SHOW - Print the Current Simulation Data Set Number, Tactical Scenario Number, and Run Number.

USE - Change the Current Simulation Data Set, Tactical Scenario, and/or Run Number.

3-5. Create/Edit Air Scenario.

This subprocess is used by the MOPADS modeler to create new air scenarios for the MOPADS user. Figure III-15 is a schematic of the organization of the data base information for Air Scenario data. A Critical Asset Configuration entry contains the basic data such as the coordinate reference point and the locations of all critical assets. Then one or more air scenarios can be created with reference to this coordinate and asset system. Each air scenario may contain tracks classified as hostile, friendly, and other.

Commands unique to this subprocess are:

ADD-AIR - Creates a new air scenario entry with tracks

ADD-ASSETS- Creates a new critical asset configuration entry

DELETE - Deletes a single air scenario or an entire Critical Asset Configuration entry

GET - Copies an air scenario or an entire Critical Asset Configuration from a reference data base to the working data base.

RENAME - Renames an air scenario or Critical Asset Configuration

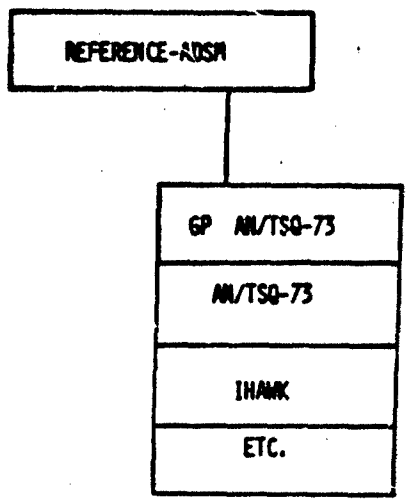
SAVE - Copies an air scenario or an entire Critical Asset Configuration from the working data base to a reference data base.

The user interface contains provisions to maintain reference type information such as scenario data in a reference data base so that it is not subject to accidental loss. It can be copied into working data base files as needed. The GET and SAVE commands perform this function. Use of this subprocess is described in Polito (1983a).

3-6. Create/Edit Reference System Module.

This subprocess is used by the MOPADS modeler to enter data for new models of air defense systems into the data base. Figure IV-5 shows the organization of the reference system module information in the data base. When a new system module is being created, and all of the data have been prepared, then this subprocess is used to enter default human factors and system option parameters for the module. This effectively increases the menu of system modules available to the MOPADS user for performing simulations.

CREATE/EDIT REFERENCE AIR DEFENSE SYSTEM MODULE



COMMANDS:

ADD CHANGE DELETE GET QUIT RENAME SAVE USE

Figure IV-5. Data Base Organization for the "Create/Edit Reference System Module" Subprocess.

The commands unique to this subprocess are:

ADD - Create a new reference air defense system module

CHANGE - Edit the parameters of a reference system module

DELETE - Delete a reference system module

GET - Copy a reference system module from a reference data base to the working data base

RENAME - Rename a reference system module

SAVE - Copy a reference system module from the working data base to a reference data base

USE - Specify a particular reference system module as "current"

QUIT - Leave the subprocess

Use of this subprocess is described in Polito (1983b).

3-7. Basic Data Base Commands.

In addition to the commands discussed for each subprocess, there are several basic commands that can be entered from any subprocess. Those commands provide low level editing or data base handling services for the user. The basic data base commands are discussed below.

CLOSE - The CLOSE command will close either the working or reference data base. It can be used to switch to a new data base file.

CURRENT - The CURRENT command will display the label, ID or both of the current directory and/or data list on either data base.

DEPOSIT - DEPOSIT is a low level editing command that allows any element of the current data list to be changed. DEPOSIT interactively requests element numbers and new values.

DIRECTORY - DIRECTORY shows the contents (all owned directories and/or data lists) of the current directory on either data base. It shows the labels, ID's, and directory positions of the contents. This information is useful for the SELECT command.

EXAMINE - EXAMINE will display selected contents of the current data list to the terminal or to the MOPADS non-interactive output file. If the latter is selected, the data list label and other information will also be printed.

HELP - HELP will print the prompts and options for the prompts for the specified command.

MENU - MENU has no prompts. It will print all commands available in the current subprocess.

OPEN - OPEN will open a data base file as either the working or reference data base. OPEN will not automatically close the current data base. CLOSE must be used explicitly before OPEN to switch data base files.

PLINK - PLINK will change the current directory to the owner of the directory which was current when PLINK was issued.

SELECT - SELECT changes the current directory or data list to one that is owned by the directory that is current when SELECT is issued. The desired directory or data list is selected by specifying one (and only one) of the following: 1 - its directory position, 2 - its label, or 3 - its ID. This information is obtained with the DIRECTORY command.

TERMINATE - TERMINATE has no prompts It will close all open data bases and terminate execution. This is the normal way to end a User Interface session.

3-8. Conversing With the MOPADS User Interface.

The User Interface is primarily a command driven processor that waits for the user to issue instructions. It does, however, have aspects of menu driven systems in that some commands result in menus being presented to the user. Also, the command processor (FFSP described in Goodin & Polito (1983a)) permits menu-like presentations of commands.

The regular mode for entering commands is shown below.

command,prompt1=response1/prompt2=response2/...

The commands and prompts are keywords recognized by MOPADS. The responses are particular values for the prompts. For example, consider this:

```
OPEN,FILE=MOPADS.DBF/STATUS=OLD
```

OPEN is the command. FILE and STATUS are prompts recognized by MOPADS and MOPADS.DBF and OLD are values for the prompts.

Certain prompts for a command may have default values that will be used if the prompt is not entered. In the example above, another prompt, DB, specifies which data base is to be associated with the file. Its default is WORKING, so by not entering it on the command line, WORKING is automatically selected. If the default value is not desired, then the prompt must be explicitly entered on the command line.

If the user fails to enter responses to all required prompts, MOPADS will interactively prompt for them. For example,

```
OPEN,STATUS=OLD

FILE[NO DEFAULT] = MOPADS.DBF
```

While processing the OPEN command, MOPADS found that the required prompt, FILE, was not entered. It printed "FILE[NO DEFAULT]=" to prompt the user for a response. If the last non-blank character on a command line is a dash (-), MOPADS will interactively prompt for all unentered prompts, even those with defaults. For example,

```
OPEN,STATUS=OLD -
DB[WORKING] = REFERENCE
FILE[NO DEFAULT] = MOPADS.DBF
```

The dash caused "DB[WORKING] = " to be printed. The value between the brackets is the default for the prompt. The default can be selected by typing "DEF" as the response. DEF can also be entered on the command line; e.g.,

```
OPEN,DB=DEF/STATUS=OLD/FILE=MOPADS.DBF
```

The above demonstrates that the prompt-response groups can be entered in any order.

Also, a command can be cancelled at any time by typing "CANC" as a response or a prompt. For example,

```
OPEN,CANC

OPEN,FILE=CANC
```

Note that DEF and CANC are essentially reserved words. The user interface treats commas, blanks, and equal signs as interchangeable separators. Also, multiple separators are treated as a single separator. This means that the commas in the previous examples could be replaced by any combination of one or more blanks and commas. The same is true of the equal signs, but their use is recommended to make the command lines easy to read. The slashes are required separators between prompt-response groups, but they can be preceded or followed by blanks or commas.

A response may include separators (i.e., commas, blanks, equal signs, and slashes) if it is enclosed in quote marks ("). For example, on some computers file names contain embedded blanks, e.g.,

```
OPEN  FILE="MOPADS DBF"
```

Without the quote marks above, MOPADS will consider MOPADS DBF as two responses when only one is desired. (NOTE: A single prompt may have more than one response if the programmer specified it that way. In such a case, each response would be separated by a blank or comma. In the case above, however, where a single response is required, the quote marks must be used to embed the blank in the response.)

Any response may be enclosed in quotes, although there is no advantage in doing so unless a separator is to be embedded. Blank responses can be entered with " " where at least one blank appears between the quotes.

A generalization of entering only some of the prompts is to enter only the command name:

```
OPEN

DB[WORKING]=DEF

FILE[NO DEFAULT]=MOPADS.DBF

STATUS[OLD]=DEF
```

The User Interface will prompt for all responses. This method can be selected if the user does not remember the prompts.

For commands which the user issues frequently, a concise mode can be selected by preceding the command with "C-". In this case, the prompt= part of the syntax may be omitted. For example,

C-OPEN DEF/MOPADS.DBF

Responses must be entered in the same order as they are prompted in the command-name-only form. No response may be skipped, except that if all remaining responses have defaults and the defaults are desired, then the command line may be terminated (e.g., the STATUS response was omitted above since OLD was desired). The dash works in the concise mode in the same way as in other modes.

The following rules will formalize the previous discussion of how syntax is processed by FFSP.

1. The command-name-only form of a command may be used at any time by typing only the command name.

2. Blank responses and responses containing separators may be entered by enclosing them in quotes. To enter a blank response,type " " (including the quotes). At least one blank must be entered between the quote marks.

3. A command may be cancelled at any time by typing CANC for any prompt or response. You can not abbreviate CANC.

4. The user may elect to use the default value(s) by typing DEF for any response in a response list up to one field past the last response in the list.

5. Slashes (/) must be used to separate one prompt-response group from another. Blanks or commas may be used to separate all other fields. The equal sign should be used to separate prompts from their responses; however, it is not required.

6. Command and prompt names may be abbreviated to any non-ambiguous string of characters. For example, if there are two commands, DESIGN and DESCRIBE, they can be abbreviated DESI and DESC respectively. The commands may be abbreviated in longer forms. For example, the user may enter DESC, DESCR, DESCRI, DESCRIB, or DESCRIBE for the command DESCRIBE.

7. If a command line in regular or concise mode is ended with more than one dash, the last dash will signify to

the system to prompt the user for all the unentered responses. Other dashes will then be considered as part of a response.

8. Any multiple combination of commas and blanks is treated as a single separator. For example,

NAME = BILL WOLF and NAME = BILL , WOLF

are equivalent (here the response is a list of two character strings).

9. If the user enters an incorrect response or misues the syntax, FFSP will explain the error and prompt interactively for all remaining responses.

10. Concise mode is signified by preceding the command name with "C-" (without the quotes).

The user interface is designed to be easy to use and to support novice, intermediate, and expert users. The HELP and MENU commands and the command-name-only form provide extensive help for novice users. Conversely, the concise mode allows frequently used commands to be entered without excessive typing. Furthermore, the ability to use abbreviations for commands and prompts permits experienced users to reduce typing effort in even the regular mode. Examples of User Interface commands and terminal sessions are given in Polito (1983a,b).

4-0 UTILITIES AND SUPPORTING SOFTWARE

4-1. MOPADS Utilities.

A set of utility programs have been developed to support the rest of the MOPADS software (Polito & Goodin, 1983). These programs provide standard services in loading and copying arrays, managing auxiliary array storage, opening files, and encoding/decoding character strings in a machine independent scheme. These low level services are used by virtually all other MOPADS software modules.

4-2. FFSP and FFIN2.

Two software modules are used to provide interactive terminal functions for the user interface. FFIN2(Polito, 1983c) is a set of format free input programs that predate MOPADS. FFIN2 provides extensive error checking for input entered from the terminal, and it protects the user from abnormal termination due to mistyping.

FFSP (Free Format Syntax Process) uses FFIN2 to interpret the syntax of the user interface described in Section 3-7 above. The FFSP is general in nature in that is is entirely data driven. This allows the set of commands now recognized by MOPADS to be expanded easily if the need requires. Indeed, FFSP can be used to interpret commands in a setting entirely removed from MOPADS.

All form, content, and syntax error checking is performed at the lowest level in FFIN2 or FFSP. This relieves the user interface software which implements the commands from the chore of performing these checks for each separate command type. The user interface programs can avoid duplication and concentrate on detection errors in meaning rather than form.

V. GUIDE TO MOPADS DOCUMENTATION

1-0 MOPADS VOLUME 1

This report comprises MOPADS Volume 1.1. It is the final report for the project and provides an introduction to the concepts used in MOPADS. Subsequent MOPADS volumes contain user and refernece material for the MOPADS modeler.

1-1. Volume 1 Contents.

Volume	Title and Description
1.1	MOPADS Final Report This report describes, in a non-technical manner, the MOPADS modeling system and the MOPADS software. The methodology is described and a simplified description of the software implementation is given.

2-0 MOPADS VOLUME 2

There are no documents in MOPADS Volume 2.

3-0 MOPADS VOLUME 3

MOPADS Volume 3 contains reports that provide user documentation. They are mandatory reading for individuals who will design, perform, and analyze simulations using MOPADS. These documents provide sufficient information for a MOPADS user to exercise the models that exist in MOPADS.

3-1. Volume 3 Contents.

Volume	Title and Description
3.1	User Guide for the AN/TSQ-73 System Module This document describes the AN/TSQ-73 model. It provides information required by a MOPADS user such as a) the number and type of operators, b) the default human factors and system parameters, c) the operator goals, d) other data requirements, and 3) a discussion of the implementation.
3.2	User Guide for the IHAWK System Module This document is analogous to Volume 3.1 except for the IHAWK system.
3.3	Performing MOPADS Simulations This document contains user instructions on how to use the MOPADS User Interface to set up and perform MOPADS simulations. It also contains information on analyzing the outputs from simulation. It's intended audience is the MOPADS user.

4-0 MOPADS VOLUME 4

MOPADS Volume 4 is a collection of documents for the MOPADS modeler who will design and develop MOPADS models of new air defense systems and integrate them with the rest of the MOPADS system.

4-1. Volume 4 Contents.

Volume	Title and Description
4.1	MOPADS Architecture Manual This document contains a description of the modeling framework and the software structure of MOPADS. It is prerequisite reading for the MOPADS modeler.
4.2	FORTRAN Style and Documentation Requirements All new FORTRAN code written for the MOPADS project has been written following the standards specified in this document. It is recommended that all follow-on developments also use this standard.
4.3	Documentation Requirements and Development Guidelines for MOPADS Air Defense System Modules This document describes the documentation procedures that have been used to develop the AN/TSQ-73 and IHAWK system modules. It is recommended that the procedures in this report be used for all follow-on developments also.
4.4	Development Methodology for MOPADS Air Defense This document describes a step-by-step methodology for developing air defense system modules. It has been used for the AN/TSQ-73 and IHAWK system modules. It is recommended that these procedures be used in all follow-on developments.
4.5	MSAINT User's Guide: Changes and Additions to the SAINT User's Manual

	This document describes changes made to the SAINT simulation language to support MOPADS. It is written as a supplement and addendum to existing SAINT documentation. Therefore, the reader must be familiar with the SAINT language.
4.6	Creating Reference Air Defense System Modules This document describes the procedures for using the MOPADS user interface to enter data for a new air defense system module into the MOPADS data base.
4.7	Creating MOPADS Air Scenarios This document describes the procedures for using the MOPADS user interface to enter new air scenario and critical asset configuration data into the MOPADS data base.

5-0 MOPADS VOLUME 5

MOPADS Volume 5 is a collection of reference documents that describe the methodology and software modules of MOPADS. They are intended as reference reports of primary interest to the MOPADS modeler, although MOPADS users may find some of them interesting.

5-1. Volume 5 Contents.

Volume	Title and Description
5.1	A Summary of the Literature on Quantitative Human Performance Models This document contains summary sheets of the literature search conducted of the human performance literature for MOPADS. It is comprised primarily of raw data used for the data base described in Volume 5.2.
5.2	A Data Base for Quantitative Human Performance Modeling

This document describes a computerized system for organizing the information described in Volume 5.1 and the use of the computer program to identify relevant literature to develop the MOPADS skills taxonomy.

5.3 Selected Operator Functions and Tasks for the AN/TSQ-73 Missile Minder

This working document contains raw task flow chart data extracted from AN/TSQ-73 system documentation.

5.4 Selected Operator Functions and Tasks for the Improved HAWK Missile Battery

This document is analogous to Volume 5.3 except for the IHAWK.

5.5 The Underlying Person Model Behind HOMO (Human Operator Model)

This document describes the skill taxonomy for modeling air defense operators.

5.6 HOMO Establishment of Performance Criteria for Non-Decision Making Tasks

This document describes the moderator function approach developed for MOPADS, and the way in which task times are moderated by breaking tasks down into component skills.

5.7 MOPADS Task Sequencing Structure

This document describes the methodology used for operator task sequencing. Operators are represented in MOPADS as goal seekers. This report describes the procedures used to model the goal seeking behavior.

5.8 MOPADS Documentation Style Manual

This report describes the documentation style used for writing and typing MOPADS reports.

5.9 MOPADS Utility Programs

This report documents the MOPADS software module UTIL which contains general program utilities.

5.10 Human Factors Moderator Functions

This report documents the MOPADS software module which implements the human factors moderator functions.

5.11 MOPADS Free-Format Syntax Processor (MOPADS/FFSP)

This report documents the MOPADS software module which interprets the MOPADS user interface command syntax.

5.12 MOPADS User Interface (MOPADS/UI)

This report documents the MOPADS software module that implements the user interface.

5.13 The MOPADS Data Base Control System (MOPADS/DBCS)

This report documents the MOPADS software module that manipulates the MOPADS data base file.

5.14 MOPADS Free-Format Input Program (MOPADS/FFIN2)

This report documents the MOPADS software module that performs low level format free input for FFSP (see Volume 5.11) and the user interface.

5.15 Documentation Manual for the AN/TSQ-73 System Module

This report contains detailed documentation of the AN/TSQ-73 system module. Complete SAINT subnetworks for each operator task are shown with cross references between SAINT task node numbers, operator task numbers, and references to Army documentation. Also, all SAINT user FORTRAN inserts are documented.

5.16 Documentation Manual for the IHAWK System Module

This document is analogous to Volume 5.15 except for the IHAWK system module.

5.17 The MOPADS Data Base

This report specifies the contents and format of the MOPADS data base file.

5.18 The MOPADS Data Base Application Programs (MOPADS/DBAP)

This report documents the MOPADS software module that contains utilities that interact with the MOPADS data base through the DBCS (Volume 5.13).

5.19 Documentation Manual for the MOPADS Control Module (MOPADS/CNTRL) and The MOPADS Common System Module Module Programs (MOPADS/CSMP).

This document is analogous to Volumes 5.15 and 5.16 for a system module called the Control System Module. This module is not an air defense system module. Rather it is a special SAINT subnetwork which manages aircraft tracks and drives the software that updates track position and ATDL information. In addition, certain programs common to all system modules are documented.

VI. REFERENCES

Finley, D. L., Obermayer, R. W., Bertone, C. M., Meister, D., & Muckler, F. A. Human performance prediction in man-machine systems (NASA-CR-1614) August 1970.

Goodin II, J. R., & Polito, J. MOPADS free-format syntax processor (MOPADS/FFSP) (MOPADS Vol. 5.11). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Goodin II, J. R., & Polito, J. User guide for the AN/TSQ-73 system module (MOPADS Vol. 3.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Goodin II, J. R., & Polito, J. User's guide for the IHAWK system module (MOPADS Vol. 3.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Goodin II, J. R., & Walker, J. L. Documentation manual for the AN/TSQ-73 system module (MOPADS Vol. 5.15). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Goodin II, J. R., & Walker, J. L. Documentation manual for the IHAWK sytem module (MOPADS Vol. 5.16). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft.Bliss, TX, 1983 (b)

Laughery, K. R. A data base for quantitative human performance modeling (MOPADS Vol. 5.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981.

Laughery, K. R. & Ditzian, J. L. The underlying person model behind HOMO (MOPADS Vol. 5.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981.

Laughery, K. R. & Ditzian, J. L. A summary of the literature on quantitative human performance models (MOPADS Vol. 5.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1982.

Newell, A. & Simon, H. Human problem solving. New York: A Prentice Hall Bool, 1972.

Pew, R. W., Baron, S., Seehrer, C. E., & Miller, D. C. A critical review and analysis of performance models applicable to man-machine systems evaluation, Bolt-Beranek & Newman, Cambridge, MA (BBN-3446). Air Force Office of Scientific Research, Bolling Air Force Base, DC, 1977. (TR No. AFOSR-TR-77-0520)

Polito, J. Creating MOPADS air scenarios (MOPADS Vol. 4.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Polito, J. Creating reference air defense system modules (MOPADS Vol. 4.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Polito, J. MOPADS free-format input programs (MOPADS/FFIN2) (MOPADS Vol. 5.14). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Polito, J. The MOPADS data base control system (MOPADS/DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d)

Polito, J. The MOPADS data base application programs (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (e)

Polito, J., & Goodin II, J. R. MOPADS utility programs (MOPADS/UTIL) (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. & Walker, J. L. An implementation of "sharable' networks in a process oriented simulation language, presented at the Joint National ORSA/TIMS Conference, San Diego, CA, October 25-27, 1982.

Siegel, A. I., Pfeiffer, M. G., Kopstein, F. S., Wilson, L. G., & Ozkaptan, H. Human performance in continuous operations (ARI Research Product No. 808-4A) Applied Psychological Services, Inc., Wayne, Pennsylvania. Army Research Institute, Alexandria, VA, 1979. (NTIS No. AD-808-6131)

U. S. Army. Technical manual (TM 9-1430-652-10-3) Operator's manual: initialization and operating procedures, guided missile air defense system (AN/TSQ-73, August 1978, Change 6, July 9, 1981.

U. S. Army. Technical manual (TM 9-1430-1526-12-1) Operator and organizational maintenance manual improved guided missile battery control control AN/TSQ-11 1430-01-042-4910, improved HAWK air defense guided missile system, June 30, 1979.

Walker, J. L., & Polito, J. Development methodology for MOPADS air defense system modules (MOPADS Vol. 4.4). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1982 (a)

Walker, J. L., & Polito, J. Documentation requirements and development guidelines for MOPADS air defense system modules (MOPADS Vol. 4.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

VII. CHANGE NOTICES

APPENDIX A

MOPADS FINAL REPORT:

USER GUIDE FOR THE AN/TSQ-73 SYSTEM MODULE

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology and Abbreviations

BN	Battalion
Q-73	AN/TSQ-73
TD	Tactical Director
TDA	Tactical Director Assistant
TD1, TD2	The operator of a Battalion AN/TSQ-73

1. INTRODUCTION

1-0 PURPOSE

This document is a user manual for the AN/TSQ-73 system module. It contains details of the implementation of the MOPADS model of the AN/TSQ-73 system. This information is found nowhere else in the MOPADS documentation. Actually, two system modules exist for the AN/TSQ-73: one for Battalion operators and one for Group operators. Both are described in this report.

The contents of the operator state vectors, environmental state vectors, and task data are given in this report. Section II contains a brief description of the AN/TSQ-73 system. Section III presents the contents of the operator state vectors. The environmental state vector is described in Section IV. The operator tasks and task sequencing assumptions are given in Section V. Finally, Section VI describes assumptions used in developing the module.

2-0 INTENDED AUDIENCE

MOPADS users and MOPADS modelers should read this report. The language and level of detail used in the discussion is oriented to the MOPADS user, however. Programming and implementation details are not included here. This information is discussed in detail in MOPADS volumes four and five.

3-0 OTHER READING

The reader is presumed to be familiar with the MOPADS system. This implies that he/she should have read the final report, Polito & Laughery 1983, prior to reading this document. In addition, the reader should be familiar with the main MOPADS user manual, Polito (1983a). More technical data on the AN/TSQ-73 system module is found in Goodin & Walker 1983.

II. SYSTEM DESCRIPTION

The AN/TSQ-73 (Q-73) is the command center for all the air defense systems in MOPADS. The primary mission is to protect vital assets such as airfields, and ammunition supply centers from enemy air attacks. The Q-73 performs this mission by controlling various fire units so that the mission is carried out effectively and as efficiently as possible. The Q-73 has the ability to detect, identify, track and direct engagements against aircraft flying at various altitudes.

MOPADS can represent two echelons of command in the Q-73's. The highest level is the Group Q-73 which may control one or more Battalion Q-73's. The Battalion Q-73 may have several IHAWK fire units under its control. The Group Q-73's serve as checks on the actions of the Battalion Q-73's so that two Battalion do not engage the same aircraft.

During an air scenario the Battalion Q-73 will work with its fire units in detecting and identifying aircraft. The Battalino may assign the aircraft to different fire sections depending on which one is the most capable of handling the engagement. The Battalion will always have the final word on whether or not to fire at a target it assigned to a fire section.

There are two operators manning the Group and Battalion Q-73. The MOPADS Q-73 system modules represent these operators. The operators are described in the following forms. On the forms the MOPADS operator type is shown in the column labeled OPERATOR NUMBER, and the MOPADS label for each operator is shown in parentheses in the column labeled OPERATOR TITLE.

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
1(BN)	TACTICAL DIRECTOR (TD)	The TD directs the activities of the Battalion Q-73. He directs the IHAWK fire units to engage and disengage targets. Only the TD has authority to direct that a track be fired upon.	
8,9(GRP)	TACTICAL DIRECTORS(2) (TD1, TD2)	The tactical directors at the Group Q-73 have equivalent authority to those at Battalion. In addition, they may direct the Battalion to engage or disengage targets. In the MOPADS module, they use the authority to insure that friendlies are not fired upon and to insure that fire units from different Battalions do not engage the same track.	
2(BN)	TACTICAL DIRECTOR ASSISTANT (TDA)	Performs all detection,tracking, and identification tasks under normal conditions. If the clear alerts function requires a full time operator, then the TDA performs this function.	
NOTE: The BN Q-73 modeled will always have one TD and one TDA while Group uses 2 operators with authority analogous to a TD.			
NAME: JACK WALKER DATE: 7/20/83		AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS	
OPERATOR DEFINITIONS			

III. OPERATOR DATA

1-0 TACTICAL DIRECTOR (TD)

1-1. Human Factors Parameters.

Table III-1 shows the default values for the human factors parameters of the Battalion Tactical Director (TD). This data is part of the operator state vector for the TD. Discussions of these variables are contained in Polito & Laughery 1983, Laughery 1983, & Polito (1983b).

The last elements of this list, X-SCREEN-CENTER, Y-SCREEN-CENTER, and SCREEN-RANGE, are set when the system module is copied into a command and control configuration.

Table III-1. TD Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
6.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
5.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
6.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE

Table III-1 (continued)

4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.2080000E-01	TARGET-HEIGHT
0.2080000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
800.0000	DISPLAY-RESOLUTION
0.5000000E-01	DISPLAY-BACKGROUND-HEIGHT
0.5000000E-01	DISPLAY-BACKGROUND-WIDTH
0.5000000E-01	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

1-2. Goals, Goal Priority Functions, and Objective Parameters.

There are eight goals defined for the AN/TSQ-73 operators. Figure III-1 explains the goals and the goal states. Not all operators have all goals. The TD has goals 1, 2, 3, 6, 7, and 8.

Goals one and two reflect the operator's desires to protect himself and defend the critical assets. For the TD, the critical assets that he is trying to protect are those assigned to his fire units.

Each Track is evaluated to calculate its time to arrive at the AN/TSQ-73 location and at each of the protected sites. The minimum of these times is the track's threat. Thus, the most threatening track has the smallest value for its threat. Goal three motivates the TD to attack enemy aircraft even if they do not pose an immediate high threat.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
1	Self Defense- maximize the minimum time to arrive of any threatening track.	The minimum time for any unassigned, not-receding, hostile or unknown track to arrive if it immediately turned inbound.
2	Protect Critical Assets - maximize the minimum time to arrive at a critical asset of any threatening track.	The minimum time for any unassigned, not-receding, hostile or unknown track to arrive at any protected site if it immediately turned inbound to the site.
3	Attack Threatening Tracks	The number of unassigned, visible, hostile or unknown, not-receding tracks and the number of covered tracks.
4	Minimize the number of unidentified tracks	The number of unidentified tracks visible or visible to owned fire units.
5	Minimize the number of uninitiated video contacts	The number of visible video contacts
6	Conserve ammunition	The number of tracks being engaged by more than one fire unit

NAME: Joseph Polito
DATE: 8/25/83
AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 1 of 2

Figure III-1. AN/TSQ-73 Operator Goals.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
7	Respond to Communications	The maximum priority of any outstanding message that the operator may receive and that no one else is processing.
8	Protect Friendly Tracks	The number of friendly tracks engaged or assigned.

NAME: Joseph Polito
DATE: 8/25/83

MAIN DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 2 of 2

Figure III-1 (continued)

Goals four and five cause the TDA to initiate and identify tracks. Goal six will motivate the TD to prevent more than one of his fire units from simultaneously engaging the same track. Goal seven causes the operators to respond to communications from within their own unit and from other units. Goal eight will cause engagements against friendly aircraft to be terminated. The situation can arise if a friendly track appears to be a high threat before it can be identified.

Goal priority functions must be specified for the goals which specify how the goal priority changes with differing values of the goal states. As discussed in Polito & Laughery 1983 and Polito (1983c), this is done by specifying two points on the goal priority function above and below the range of satisfaction. This data for the TD is shown in Figure III-2.

Table III-2 shows how this data appears in the operator state vector. Each goal is represented by 15 values. The first of which is the goal number. If this number is negative then the operator does not have that goal. The data shown in Figure III-2 is given in the elements LITTLE-M, BIG-M, and elements GOAL-STATE-LOW-1 through PRIORITY-HIGH-2. The values of LITTLE-A, LITTLE-B, BIG-A, and BIG-B are calculated. See Polito (1983a) for warnings on changing these numbers. Infinity is represented in the operator state vector by the number 10^{10}.

Objective function parameters for the TD are contained in his operator state vector, Table III-1. The objective function is specified as "one" which means that the TCO selects the task which offers the greatest improvement in his goal state. If objective function two is specified, the TCO attempts to obtain the greatest goal improvement per unit time. Finally, the TD considers all of his goals when selecting his next task (i.e., GOALS-CONSIDERED is six).

Table III-2. TD Goal Priority Function Parameters.

1.000000	GOAL
90.00000	LITTLE-M
0.1000000E+11	BIG-M
0.9290031E-30	LITTLE-A
15.93882	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
1.500000	GOAL-STATE-LOW-1
10.00000	PRIORITY-LOW-1
10.00000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2

Table III-2 (continued)

2.000000	GOAL
90.00000	LITTLE-M
0.1000000E+11	BIG-M
0.8003029E-31	LITTLE-A
16.47427	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
1.500000	GOAL-STATE-LOW-1
9.500000	PRIORITY-LOW-1
10.00000	GOAL-STATE-LOW-2
1.800000	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2
3.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
2.000000	BIG-A
0.7195524E-02	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
2.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.010000	PRIORITY-HIGH-2
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2
6.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
2.500000	BIG-A
0.1659562	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
2.500000	PRIORITY-HIGH-1
3.000000	GOAL-STATE-HIGH-2
3.000000	PRIORITY-HIGH-2
7.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
1	90, ∞	1.5	10	10	2.0	-	-	-	-
2	90, ∞	1.5	9.5	10	1.8	-	-		-
3	-∞, 0	-	-	-	-	1	2	2	2.01
6	-∞, 0	-	-	-	-	1	2.5	3	3
7	0, 0	-1	1	-2	2	1	1	2	2
8	-∞, 0	-	-	-	-	1	9	2	9.01

OPERATOR: TD/1

NUMBER OF GOALS CONSIDERED: 6

NAME: POLITO
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 Battalion
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION Page ___of __

Figure III-2. Goal Priority Function Data for the TD.

Table III-2 (continued)

1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2
8.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
9.000000	BIG-A
0.1602134E-02	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
9.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
9.010000	PRIORITY-HIGH-2

1-3. Display Data.

Table III-3 is the display data for the TD. The first element is the currently hooked item if one exists. The other elements are explained below.

VECTORS (0 - no vectors, 1 - velocity vectors on, 2 - Time to go vectors on)

SCREEN-ALPHA (0 - Off, 1 - On)

ENGAGEMENT-MARKERS (0 - Off, 1 - On)

PAIRING-LINES (0 - Off, 1 - On)

MAP (0 - Off, 1 - On)

SCREEN-RANGE (default is 100 nautical miles)

Table III-3. TD Display Data.

0.0000000E+00	HOOKED-ITEM
1.000000	VECTORS
0.0000000E+00	SCREEN-ALPHA
1.000000	ENGAGEMENT-MARKERS
1.000000	PAIRING-LINES
1.000000	MAP
100.0000	SCREEN-RANGE
-9999.000	SCREEN-X
-9999.000	SCREEN-Y
0.0000000E+00	UNUSED

The coordinates of the center of the screen are not specified until the system module is copied into a command and control configuration. At that time, the user may specify the position of the center of the TD's viewing area. If the user fails to specify this data (by editing the operator state vector), then MOPADS will set it equal to the position of the AN/TSQ-73.

2-0 TACTICAL DIRECTOR ASSISTANT (TDA)

2-1. Human Factors Parameters.

Table III-4 contains the human factors parameters for the Tactical Director Assistant (TDA). The comments in Section 1-1 apply equally to the data for the TDA.

2-2. Goals, Goal Priority Functions, and Objective Parameters.

The TDA has goals four, five, and seven from Figure III-1. Goals four and five causes the TDA to initiate and identify tracks, and goal seven causes him to attend to communications. The goal priority data for the TDA is shown in Figure III-3. Table III-5 shows how this data appears in the operator state vector.

Table III-4. TDA Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
6.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
5.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
3.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
4	$-\infty$, 0	-	-	-	-	5	1.5	50	7
5	$-\infty$, 0	-	-	-	-	5	1.3	50	6
7	$-\infty$, 0	-1	1	-2	2	1	1	2	2

OPERATOR: TDA/2

NUMBER OF GOALS CONSIDERED: 3

NAME: POLITO
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 Battalion
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION

Figure III-3. Goal Priority Function Data for the TDA.

Table III-4 (continued)

2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.2080000E-01	TARGET-HEIGHT
0.2080000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
800.0000	DISPLAY-RESOLUTION
0.5000000E-01	DISPLAY-BACKGROUND-HEIGHT
0.5000000E-01	DISPLAY-BACKGROUND-WIDTH
0.5000000E-01	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

Table III-5. TDA Goal Priority Function Parameters.

4.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
0.5110644	BIG-A
0.6690068	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
5.000000	GOAL-STATE-HIGH-1
1.500000	PRIORITY-HIGH-1
50.00000	GOAL-STATE-HIGH-2

Table III-5 (continued)

7.000000	PRIORITY-HIGH-2
5.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
0.4463566	BIG-A
0.6642079	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
5.000000	GOAL-STATE-HIGH-1
1.300000	PRIORITY-HIGH-1
50.00000	GOAL-STATE-HIGH-2
6.000000	PRIORITY-HIGH-2
7.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2

The TDA has objective function one as does the TCO, and he considers all three goals in task sequencing.

2-3. Display Data.

The TDA has the same display data as does the TD. See Table III-3.

3-0 GROUP AN/TSQ-73 OPERATORS (TD1 and TD2)

3-1. Human Factors Parameters.

Table III-6 contains the human factors parameters for the Group AN/TSQ-73 (these operators are labeled TD1 and TD2. The comments in Section 1-1 apply equally to the data for TD1 and TD2. These operators are identical and each have authority analogous to the Battalion TD.

Table III-6. TD1 and TD2 Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
6.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
5.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
3.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.2080000E-01	TARGET-HEIGHT
0.2080000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
800.0000	DISPLAY-RESOLUTION
0.5000000E-01	DISPLAY-BACKGROUND-HEIGHT

Table III-6 (continued)

0.5000000E-01	DISPLAY-BACKGROUND-WIDTH
0.5000000E-01	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

3-2. Goals, Goal Priority Functions, and Objective Parameters.

TD1 and TD2 have goals six, seven, and eight from Figure III-1. These operators respond to messages because of goal seven just as do the Battalion operators. They also perform a coordination function for the Battalions due to goals six and eight just as the Battalions perform a coordination function for their fire units. The goal priority function data for TD1 and TD2 is shown in Table III-7 and Figure III-4. TD1 and TD2 use objective function one and consider all goals in task sequencing.

3-3. Display Data.

The TD1 and TD2 have the same display data as does the TDA. See Table III-3.

Table III-7. TD1 and TD2 Goal Priority Function Parameters.

6.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
2.500000	BIG-A
0.1659562	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
2.500000	PRIORITY-HIGH-1

Table III-7 (continued)

3.000000	GOAL-STATE-HIGH-2
3.000000	PRIORITY-HIGH-2
7.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2
8.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
9.000000	BIG-A
0.1602134E-02	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
9.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
9.010000	PRIORITY-HIGH-2

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
6	$-\infty$, 0	-	-	-	-	1	2.5	3	3
7	0, 0	-1	1	-2	2	1	1	2	2
8	$-\infty$, 0	-	-	-	-	1	9	2	9.01

OPERATOR: TD/1 TD/2

NUMBER OF GOALS CONSIDERED: 3

NAME: POLITO
DATE: 9/15/83

AIR DEFENSE SYSTEM MODULE: GROUP AN/TSQ-73
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION

Figure III-4. Goal Priority Function Data for TD1 and TD2.

IV. ENVIRONMENTAL DATA

Table IV-1 contains the AN/TSQ-73 environmental state vector. The various parameters in this vector are explained in Polito (1983a). The user should note that the default sampling option is three (deterministic, moderated). The Battalion and Group AN/TSQ-73 system modules have identical default environmental state vectors. The x, y, and z positions are set when the module is copied into a particular command and control structure.

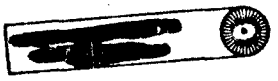
Table IV-1. The Environmental State Vector.

23.00000	POINTER-TO-HF-DATA
0.0000000E+00	SYSTEM-MODE
0.0000000E+00	OPERATOR-MODE
0.0000000E+00	UNUSED
1.000000	METHOD-OF-CONTROL
2.000000	WEAPONS-CONTOL-STATUS
0.0000000E+00	UNUSED
0.0000000E+00	INITIAL-AMMUNITION-HOT
0.0000000E+00	INITIAL-AMMUNITION-COLD
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	X-POSITION
0.0000000E+00	Y-POSITION
0.0000000E+00	Z-POSITION
3.000000	SAMPLING-OPTION
22.00000	DRY-BULB-TEMPERATURE
50.00000	RELATIVE-HUMIDITY
6.000000	AIR-MOVEMENT-RATE
45.00000	NOISE-LEVEL
50.00000	WORKING-AREA-ILLUMINATION
2.000000	NUMBER-ON-DUTY
0.0000000E+00	VIBRATION
10.00000	AMBIENT-VAPOR-PRESSURE
0.0000000E+00	NOISE-PREDICTABILITY

V. OPERATOR TASKS

AN/TSQ-73 operator tasks are given below. Each task is numbered and has a title. A brief description of each task is given here and the operator that performs the task is identified. Complete descriptions and MSAINT models of the tasks are contained in Goodin & Walker (1983).

Task Number	Title or Description	Operator
1	Scan the screen, displays, etc. - TD and TDA monitor action on displays and the CRT for new targets or tasks to perform.	TD/TDA
2	Cancel Sceondary Assignment (BN only) - The TD cancels a secondary assignment made to a fire unit.	TD
3	Send Terminate Commands - The TD of the Group or Battalion Q-73 sends a CEASE ENGAGED or HOLD FIRE command to a lower echelon of command.	TD
4	Clear, Hold Fire, Effective, or Status (BN only) - The Battalion Q-73 receives information from the IHAWK fire unit concerning status (active or inactive), engagement effectiveness, etc.	TD
5	Perform Hooking Procedure - The operator will number hook all sites and fire units and will position hook all tracks.	TD/TDA
7	Enter ID Data (BN only) - The TDA enters the results of the IFF procedure (see Task 8).	TDA
8	Interrogate A Target (BN only) - The TDA performs the procedure to identify a target as friend, foe, or neutral.	TDA



Task Number	Title or Description	Operator
10	Send Command Message (BN only) - The TD gives permission to fire on a target assigned to a fire unit by the Q-73.	TD
15	Assign Weapons (BN only) - The TD assigns a high threat target to one of the fire units.	TD
20	Receive Command (BN only) - The operators receive a CEASE ENGAGED command from a higher echelon of command.	TD/TDA
22	Clear Alerts (BN only) - The TD will perform the actions necessary to clear the alert in question.	TD
26	Receive Miscellaneous Messages - A zero time task used to receive any miscellaneous messages.	TD
30	Task Sequencing	TD/TDA

VI. OTHER FEATURES

1-0 IFF

FUNCTION IFFQ (Goodin & Walker 1983) determines the outcome of an IFF challenge. Currently it is assumed that the IFF is 100% accurate. This can easily be changed by re-writing IFFQ.

FUNCTION IFMODQ (Goodin & Walker 1983) determines what IFF mode is selected by the TDA. Currently, mode 2 is always selected. More complex selection processes can be represented by re-writing IFMODQ.

2-0 HOOKING

The current model assumes that tracks are always position hooked and that sites (e.g., fire units) are number hooked. This assumption is manifested by setting information attributes at the nodes prior to performing the hooking task. More complex selections can be represented by re-writing these programs (e.g., T15W).

3-0 ASSIGNING TRACKS TO FIRE UNITS (TASK 15)

The current model assumes that the TD selects tracks to assign himself. He does not use track assignments from the ADP. See the network and user code for Task 15 (Goodin & Walker 1983).

4-0 TRACK INITIATION AND IDENTIFICATION

The current model assumes auto-initiate and manual identification modes. Thus goal five is not operative. SUBROUTINE GEV5Q (Goodin & Walker 1983) which evaluates the goal state for this goal simulates the auto-initiate activity.

VII. STATISTICS

Figure VIII-1 shows the statistics maintained by the AN/TSQ-73 system module. These statistics are in addition to the task fraction statistics collected by MOPADS for all system modules.

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
Time-Persistent		NUMINCOV	Number of Tracks Visible	
Count			Number of Assignments	Once at end of run
"			Num Tracks Destroyed	" " " " "
"			Friendly Tracks Destroyed	" " " " "

NAME: Joseph Polito DATE: 1/10/84	AIR DEFENSE SYSTEM MODULE: Battalion AN/TSQ-73 PROJECT: MOPADS

SAINT USER STATISTICS

Figure VII-1. User Statistics for the AN/TSQ-73.

VIII. REFERENCES

Goodin, J. R. & Polito, J. Documentation manual for the MOPADS control module (MOPADS/CNTRL) and the MOPADS common system module programs (MOPADS/CSMP) (MOPADS Vol. 5.19). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983

Goodin II, J. R. & Walker, J. L. Documentation manual for the AN/TSQ-73 system module (MOPADS Vol. 5.15). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Laughery, K. R. HOMO establishment of performance criteria for non-decision making tasks (MOPADS Vol. 5.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritkser & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Polito, J. MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. MOPADS task sequencing structure (MOPADS Vol. 5.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c).

Polito, J. & Laughery, K. R. MOPADS final report (MOPADS Vol. 1.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

IX. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1E-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

X. CHANGE NOTICES

APPENDIX AA

MOPADS FINAL REPORT:

DOCUMENTATION MANUAL FOR THE MOPADS CONTROL MODULE (MOPADS/CNTRL) AND THE MOPADS COMMON SYSTEM MODULE PROGRAMS (MOPADS/CSMP)

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

MOPADS Abbreviations

CSMP	Common System Module Programs

I. SYSTEM DESCRIPTION

1-0 THE CONTROL SYSTEM MODULE

The control system module is a SAINT network model similar in structure to Air Defense System Modules. It is included automatically in every MOPADS simulation by the MOPADS software, and its purpose is to control the movement of aircraft in the simulation. There are no operators in this system module. The small network which makes up this module is used solely to schedule aircraft arrivals, checkpoints and departures from the simulation. It also schedules the end of the simulation.

2-0 THE COMMON SYSTEM MODULE PROGRAMS

The Common System Module Programs (CSMP) software module is a collection of utility programs used by all of the MOPADS system modules. These programs perform high level functions, so they are not included in the MOPADS utilities, see Polito & Goodin (1983), and many of them do not involve the MOPADS data base directory, so they are not included in the Data Base Application Program (DBAP), Polito (1983a). These programs are described in Section VIII of this document.

II. OVERVIEW OF THE SAINT MODEL

The need for the control system module arises from the way tracks are handled by MOPADS. Aircraft flight paths are represented as piece-wise linear segments. Each segment represents a portion of the flight path along which the aircraft parameters (speed, direction, etc.) remain constant. The transition points between segments are called checkpoints.

All of the aircraft tracks are stored in the MOPADS data base prior to the simulation, Polito (1983b). There may be many tracks, and it is impractical to provide array storage to hold all segments of all tracks in main memory simultaneously. The control system module performs two functions to manage the aircraft track data.

1. It periodically scans the track information in the data base for tracks which will initiate in the near future. For each such track it finds, it schedules its initiation checkpoint.

2. When a track reaches a checkpoint, it reads the data for the next segment from the data base and schedules the next checkpoint.

Between checkpoints the positions of the aircraft are updated automatically by the continuous modeling capbility of SAINT. The equations of motion of the aircraft are represented by difference equations. This is adequate since aircraft parameters are constant between checkpoints.

In addition to aircraft data management, the control system module also schedules the end of the simulation. The end time specified by the user is used in a small subnetwork to cause a SAINT sink node completion at the appropriate time.

III. MODEL DESCRIPTION FORMS

1-0 ENTITIES

The control system module creates an entity that periodically examines the data base for aircraft that will initiate in the next period. It also creates an entity for each track that it finds. These track entities complete a task each time the aircraft reaches a checkpoint. Finally, one entity is created to end the simulation at the appropriate time.

2-0 RESOURCES

No SAINT resources are used.

3-0 VARIABLES

The control system module has no internal COMMON variable.

4-0 MONITORS

The control system module has no Monitor.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
STARTTR	An entity is created that periodically (every DELT minutes) examines the data base for tracks which will initiate in the next DELT minutes	1	not used		
		2	not used		
		3	0		
		4	0		
		5-15	not used		
INITIATE	Entities are created for each track in the system. These entities perform "tasks" which are the flying of one track segment	1	-1		
		2	not used		
		3	NTRACK Column number (negative if the track has not initiated yet)		
		4	Air Scenario ID of the track		
		5-15	not used		

Name: Riley Goodin
Date: 5-13-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT ENTITIES

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
CNTRLTIM	One entity is created here and routed to node ENDTIME which has a duration equal to the simulation time (TFINK-TSTRTK). It controls the length of time to be simulated.	1-15	not used		

Name: Riley Goodin
Date: 5-13-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT ENTITIES

Page 2 of 2

5-0 TASK DESCRIPTIONS

There are six task nodes in the Control System Module.

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
1	Source node to create the periodic review entities	--	--	LABL TIME	STARTTR DS,2	

NAME: Riley Goodin
DATE: 5-18-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
1	STARTTR	--	0	0	1	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
2	This task is used to find all the tracks that will initiate in the time interval (TNOW + DELT). For each track that will initiate in TNOW + DELT the 3rd information attribute is set negative and the time for this task is Ø. When all the tracks have been found that will initiate in TNOW + DELT the 3rd information attribute is set to Ø.Ø and the time for this task is set to DELT.			LABL TIME	INITIATE UF,2	

NAME: Riley Goodin
DATE: 5-16-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION Page 1 of 1

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
2	INITIATE	--	--	--	UF	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
3	This task is used to determine the time between checkpoints for a track. When the last checkpoint for a track has been reached the 3rd information attribute is set to Ø.Ø. For all other checkpoints this value is greater than Ø.Ø.			LABL TIME	CHKPOINT UF,1	

NAME: Riley Goodin
DATE: 5-16-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
3	CHKPOINT	--	--	--	UF	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
4	This task node is a sink node used to eliminate expired tracks			LABL	KILLTRK	
				TIME	DS,2	

NAME: Riley Goodin
DATE: 5-16-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION

Page 1 of 1

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
4	KILLTRK	--	0	0	1	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
5	This task node is used to create a single entity that will control the length of the simulation.			LABL TIME	CNTRLTIM DS,2	

NAME: Riley Goodin
DATE: 5-18-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
5	CNTRLTIM	--	0	0	1	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIREMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
6	This task node is the sink node that determines how much time will be simulated. The length of this task is equal to TFINK-TSTRTK.			LABL TIME	ENDTIME UF,3	

NAME: Riley Goodin
DATE: 5-16-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT TASK DESCRIPTION

Page 1 of 1

SKILL REQUIREMENTS (CATEGORY,WEIGHT)	TASK SPECIFIC DATA (CODE, VALUE)	MSAINT RESOURCE REQUIREMENTS
none	none	none

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
6	ENDTIME	--	--	--	UF	--

NAME: Polito
DATE: 11-10-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

6-0 STATISTICS

User time persistent statistics and count statistics are collected in the control system module.

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
Time Persistent (Regular)	1	AVGNHOST	Average Number of Hostile Tracks In The System	
"	2	AVGNFRND	Average Number of Friendly Tracks In The System	
"	3	AVGNUNKN	Average Number of Other Tracks In The System	
"	4	AVGNTRKS	Average Number of Tracks In The System	

NAME: Riley Goodi../Jack Walker
DATE: 5-16-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT USER STATISTICS

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
Count	1		Number of sites destroyed	One at end of run
"	2		Number of sites attacked	" " " " "
"	3		Number of hostile tracks that are not completely destroyed	" " " " "
"	4		Total number of hostile tracks that entered the system	" " " " "
"	5		Total number of friendly tracks that entered the system	" " " " "
"	6		Total number of "other" tracks that entered the system	" " " " "
"	7		Number of hostile aircraft destroyed	" " " " "
"	8		Number of friendly aircraft destroyed	" " " " "

NAME: Polito
DATE: 8-03-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT USER STATISTICS

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
Count	9		Number of "other" aircraft destroyed	One at each end
"	10		Number of hostile tracks attacked	" " " "
"	11		Number of friendly tracks attacked	" " " "
"	12		Number of "other" tracks attacked	" " " "

NAME: Polito
DATE: 8-03-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT USER STATISTICS

7-0 USER FUNCTIONS

Three execution time user functions are used in the control system module. No initialization (input) user functions are used.

8-0 MODERATOR FUNCTIONS

No moderator functions are used by the control system module.

USER FUNCTION NUMBER	TASK NUMBER(S) WHERE CALLED (USERF ONLY)	DESCRIPTION
1	3	Computes the time to the next checkpoint for each track
2	2	Returns the time to the next periodic examination of the data base to look for tracks entering into the system. This occurs every DELT time units, so UF(2) returns either DELT or zero. Zero is returned when a track is found to enter into the system.
3	6	Returns the time interval for the simulation, e.g.,the end time minus the start time. This user function is used to schedule the end of the simulation.

WHICH USER FUNCTION? USERIN: USERF: X

NAME: Polito
DATE: 11-14-83

AIR DEFENSE SYSTEM MODULE: CONTROL
PROJECT: MOPADS

SAINT USER FUNCTIONS Page 1 of 1

IV. USER WRITTEN SUBPROGRAMS

1-0 DATA STRUCTURE

The control system module has no internal data structure that is common over subprograms. All data is passed to and from subprograms by formal parameters.

2-0 FLOW OF CONTROL

The entry point to the user written programs for the control system module is through the MSAINT user function, USERF. USERF calls the user function for the control system module, USERFC.

For user function one, USERFC calls SUBROUTINE CHKTRC which schedules the next checkpoint for a track. For user function two, USERFC calls SUBROUTINE INTTRC which checks for new tracks to enter from the data base.

No initialization of the control system module is needed, and no processing with SUBROUTINE UTASK is performed. Therefore, SUBROUTINE INITC and UTCNLC are present but have no executable code.

Some of these programs are called by SUBROUTINE STATE between checkpoints to determine if a track is visible to any of the air defense system's viewers (radars). More discussion of this topic is given in Section 5-0 below.

3-0 FILES

The Control System Module does not open or close any external file. Certain error messages are written directly to the MOPADS line printer output file, and the MOPADS data base is indirectly accessed with data base application programs, Polito (1983a).

4-0 SUBPROGRAMS

Internal documentation for the subporgrams that make up the Control System Module is shown on the following pages.

```
      SUBROUTINE CHKTRC (UF)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE:  THIS SUBROUTINE IS CALLED BY FUNCTION USERF TO DETERMINE
C            THE TIME THAT WILL ELAPSE BEFORE THE NEXT CHECKPOINT FOR
C            AN AIRCRAFT IN THE AIR SCENARIO.
C
C**OUTPUT PARAMETERS:  UF=TIME TO FLY TO NEXT CHECKPOINT
C
```

```
      SUBROUTINE CKSVC(NTCOL,X,IYN)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      CKSVC WILL CHECK TO SEE IF THE SAME VIEWER WHICH PREVIOUSLY
C      COULD SEE A TRACK CAN STILL SEE IT
C--INPUT PARAMETERS:
C      NTCOL-TRACK COLUMN IN NTRACK
C      X(3)-CURRENT POSITION OF THE TRACK
C--OUTPUT PARAMETERS:
C      IYN-YES/NO
C          1-YES, THE SAME VIEWER CAN STILL SEE THE TRACK
C          2-NO, IT CAN'T
C            IF THE TRACK WAS PREVIOUSLY NOT IN ANY VIEWER'S VIEW,
C            IYN WILL ALWAYS BE 2.
```

```
      SUBROUTINE CKVWC(NCOL,X,IYN,ALT,SECTOR,PROB)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      GIVEN A VIEWER AND A POINT, CKVWC WILL DETERMINE IF THE
C      VIEWER CAN SEE THE POINT.
C--INPUT PARAMETERS:
C      NCOL-COLUMN IN THE NSITE/XSITE ARRAY OF THE VIEWER
C      X(3)-X,Y,Z POSITION OF THE POINT THE VIEWER IS TO SEE
C--OUTPUT PARAMETERS:
C      IYN-YES/NO
C          1-YES, THE VIEWER CAN SEE THE POINT
C          2-NO, IT CAN'T SEE THE POINT
C      ALT(2)-MIN AND MAX ALTITUDE THE VIEWER CAN SEE
C      SECTOR(2)-COMPASS AZIMUTH OF THE SECTOR OF INTEREST OF THE
```

```
C                VIEWER.  IYN IS NOT RELATED
C                TO SECTOR IN ANY WAY. SECTOR IS PROVIDED AS
C                INFORMATION ONLY.
C      PROB-PROBABILITY THAT THE VIEWER WILL ACQUIRE A NEW TARGET.

-----------------------------------------------------------------------

      SUBROUTINE CNREC (KODE)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE:  THIS SUBROUTINE MAY BE USED FOR SPECIAL OUTPUT FOR
C            CONTROL SYSTEM MODULE ERRORS.
C
C**INPUT PARAMETERS:  KODE=ERROR CODE VALUE (8000-8999)
C

-----------------------------------------------------------------------

      SUBROUTINE EOSPC(ICOL,NTRKCL,NTR,ASCEN,NASC)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      EOSPC WILL PERFORM END-OF-SEGMENT PROCESSING FOR TRACKS.
C      EOSPC CHECKS TO SEE IF THE END OF SEGMENT IS A TARGET
C      AND IF THE TRACK IS HOSTILE.  IF SO, IT WILL CHECK THE
C      PROBABILITY OF DESTRUCTION TO SEE IF THE TARGET IS
C      DESTROYED.  IF SO, IT WILL SET THE STATUS AS INACTIVE.
C--INPUT PARAMETERS:
C      ICOL-COLUMN IN NTRACK OF THE TRACK
C      NTRKCL(NTR)-CONTENTS OF COLUMN ICOL OF NTRACK
C      ASCEN(NASC)-CONTENTS OF CURRENT ROW OF THE DATA BASE FOR
C                  THE SEGMENT JUST COMPLETED.
```

```
      SUBROUTINE FDVWC(NTCOL,NVCOL)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      FDVWC WILL FIND A VIEWER THAT CAN SEE A SPECIFIED TRACK IF SUCH
C      A VIEWER EXISTS.
C--INPUT PARAMETERS:
C      NTCOL-TRACK COLUMN NUMBER IN NTRACK
C--OUTPUT PARAMETERS:
C      NVCOL-VIEWER COLUMN NUMBER IN NSITE THAT CAN SEE THE TRACK.
C            ZERO IF NONE. IF THE TRACK WAS PREVIOUSLY SEEN BY A
C            PARTICULAR VIEWER ON INPUT THEN NVCOL WILL BE THE SAME
C            VIEWER(ASSUMING THAT IT CAN STILL SEE THE TRACK)
C            FDVWC DOES NOT CHANGE ROW 10 OF NTRACK TO INDICATE
C            THE VIEWER WHICH SEES THE TRACK.
```

```
      SUBROUTINE INITC(NRUN)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      INITC IS THE INITIALIZATION PROGRAM FOR THE CONTROL SYSTEM
C      MODULE.
C--INPUT PARAMETERS:
C      NRUN-RUN NUMBER
C          0-INITC IS BEING CALLED AT THE START OF THE SIMULATION
C            PRIOR TO ANY RUNS
C          N-NRUN IS BEING CALLED PRIOR TO THE BEGINNING OF
C            RUN N
```

```
      SUBROUTINE INTTRC (UF)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE:   THIS SUBROUTINE IS CALLED BY THE USERF FUNCTION TO
C            SCHEDULE TRACKS INTO THE AIR SCENARIO.
C            INTTRC EXAMINES THE TRACK DATA IN THE DATA BASE
C            AND ENTERS TRACKS THAT WILL INITIATE IN THE
C            NEXT DELT TIME UNITS.
C
C**OUTPUT PARAMETERS:  UF=THE RESULTING VALUE OF THE USER FUNCTION
C                         VARIABLE
C                         0-A TRACK HAS BEEN FOUND THAT WILL
C                           INITIATE BETWEEN TNOW AND TNOW+DELT
```

```
C                          DELT-ALL TRACK IN THE NEXT DELT TIME INT
C                               INTERVAL HAVE BEEN PROCESSED.
C                               SCHEDULE THE NEXT CALL TO INTTRC
C                               DELT TIME UNITS IN THE FUTURE.
C
```

```
      SUBROUTINE UTCNLC(NT,NPLACE)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      UTCNLC WILL PROCESS CALLS FROM UTASK FOR THE CONTROL SYSTEM
C      MODULE.
C--INPUT PARAMETERS:
C      NT-TASK NODE NUMBER
C      NPLACE-TASK NODE OCCURANCE TIME(SEE UTASK)
```

```
      FUNCTION USERFC (IP)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO CALCULATE THE VALUE OF USERFC
C            DEPENDING ON THE VALUE OF IP.
C
C**INPUT PARAMETERS:   IP=THE FUNCTION NUMBER
C                         1 IS AN AIRCRAFT CHECKPOINT
C                         2 IS AN AIRCRAFT INITIATION
C                         3 IS THE TOTAL TIME TO BE SIMULATED
C
C**OUTPUT PARAMETERS:  USERFC=THE VALUE OF THE PARTICULAR USER FUNCTION
C                           CALLED
C
```

```
      FUNCTION USERNC(ICODE)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       USERNC EVALUATES THE INPUT USER FUNCTION FOR THE CONTROL
C       SYSTEM MODULE
C--INPUT PARAMETERS:
C      ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C      USERNC-VALUE OF THE USER FUNCTION
```

5-0 USER INSTRUCTIONS

As discussed earlier, some of the programs in the User Written Programs are called primarily from SUBROUTINE STATE. These programs are concerned with determining which viewers (radars) can "see" the tracks:

CKSVC	-	This program determines if the same viewer which could previously see a track can still see it. Use of this program is more efficient than doing a complete examination of viewers.
CKVWC	-	Checks one viewer and one track to determine if the viewer can see the track.
FDVWC	-	Finds a viewer which can see a particular track.

Finally, at the end of each segment, SUBROUTINE EOSPC is called from CHKTRC to perform end-of-segment processing.

6-0 ERROR PROCESSING

The Control System Module writes a few informative warning messages directly to the MOPADS line printer output file. These are non-fatal warnings that have to do with anomolies detected in the air scenarios.

Other errors are processed with the MSAINT error processing program, SERR. Error codes 8000-8999 are reserved for the Control System Module. The error codes defined at this time are shown below.

Error Code	Error Conditions	Subprogram(s)
8000	Track Storage Arrays Full (NTRACK, SS, SSL)	INITRC
8001	NTRID and NTRID2 arrays full	INITRC
8002	A Track from the data base is not of the correct type (i.e., Hostile, Friendly, Other)	CHKTRC
8003	Invalid User Function Code	USERFC,USERNC

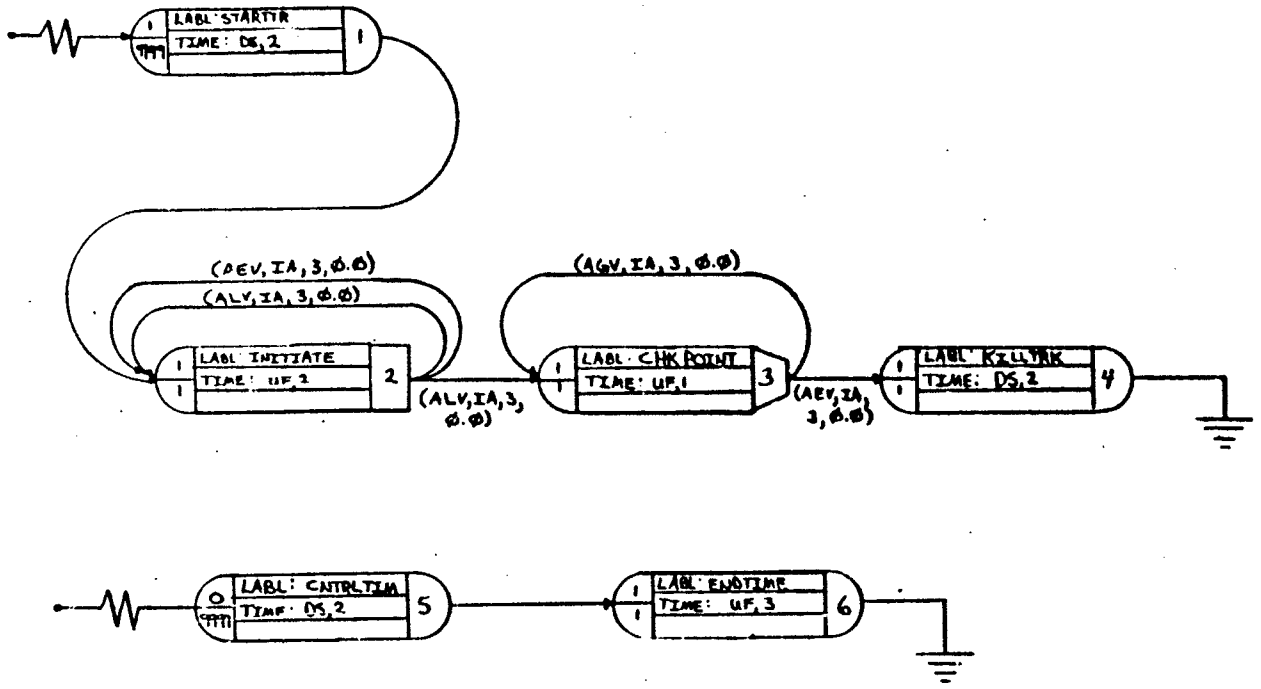
Finally, some errors that involve data from the data base are processed with the DBAP, Polito (1983a), error processing program, ERRORA. These errors have a text message displayed and print out the offending data base entry. See Polito (1983a) for a description of ERRORA.

7-0 COMMON VARIABLE DEFINITIONS

The Control System Module has no variables in COMMON areas.

V. LISTING OF SAINT NETWORK DATA INPUT

```
GEN,CONTROL,5.12,1983,1000*
POP,0,0,10,4,20*
DIS,1,CO,.001*
DIS,2,CO,0.0*
UTI,1,AVGNHOST,0.0*
UTI,2,AVGNFRND,0.0*
UTI,3,AVGNUNKN,0.0*
UTI,4,AVGNTRKS,0.0*
*
*TRACK HANDLING TASKS
*
TAS,1,STARTTR,0,9999,DS,2,(10)SO*
DET,1,2*
*
TAS,2,INITIATE,1,1,UF,2*
CAL,2,2,AEV,0.0,3,IA,,2,ALV,0.0,3,IA,,3,ALV,0.0,3,IA*
*
TAS,3,CHKPOINT,1,1,UF,1,*
CFI,3,3,AGV,0.0,3,IA,,4,AEV,0.0,3,IA*
*
TAS,4,KILLTRK,1,1,DS,2*
*
*CONTROL NETWORK FOR ENDING THE SIMULATION AT TIME TFINK-TSTRTK
*
TAS,5,CNTRLTIM,0,9999,DS,2,(10)SO*
DET,5,6*
*
TAS,6,ENDTIME,1,1,UF,3,(10)SI*
*
FIN*
$
```



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: CONTROL
--	--	PROJECT: MOPADS

MSAINT TASK NETWORKS

VI. NON-SAINT DATA REQUIREMENTS

1-0 DATA REQUIREMENTS

The Control System Module requires data on the tracks which make up the air scenario. It also operates on the viewer information. The track information is used to schedule tracks into the MOPADS simulations, and to schedule each track's checkpoints. The viewer data is used through calls from SUBROUTINE STATE to determine which radars can see the tracks.

2-0 DATA SOURCE AND DESCRIPTION

Aircraft track information is obtained from the MOPADS data base. The contents and method of specifying this information is discussed in Polito (1983b).

Data for the viewers (radars) is specifying by the user when the simulation data set is created. A discussion of this information can be found in Polito (1983c).

VII. OUTPUT REPORTS

The Control System Module produces no separate output reports.

VIII. COMMON SYSTEM MODULE PROGRAMS

1-0 PURPOSE

As discussed earlier, the Common System Module Programs (CSMP) contain programs used by all of the system modules. The MOPADS suffix for the CSMP is "Y" so all of the programs written for this software module end in "Y." The MSAINT user written programs are also incluced in this module, so some of the programs described here do not end in "Y." For example, the MSAINT programs ENDIT, INTLC, MODRF, PRIOR, STATE, UACCPT, UERR, UINPT, USCOND, USERF, USERIN, USTART, and UTASK are contained in the CSMP.

2-0 DATA STRUCTURE

The CSMP has very little internal data structure since it is comprised mainly of programs each of which performs a separate function. It does, however, contain storage arrays for MOPADS count statistics and operator task fraction statistics. Those arrays are described in Section 8-0 below.

3-0 FLOW OF CONTROL

The CSMP has no internal structure of its own and, therefore, has no flow of control that needs description. The programs are called primarily as utilities from other modules.

4-0 FILES

The CSMP does not open or close any external files. Indirect access is made to the Data Base Application Program (DBAP), Polito (1983a), and the Data Base Control System (DBCS), Polito (1983d), thorugh subprogram calls.

5-0 SUBPROGRAMS

The following pages contain descriptions of all subprograms that are part of the CSMP. Each program description contains an explanation of its purpose, parameters, and alternate returns.

```
      BLOCK DATA BLOCKY
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      BLOCKY IS USED TO INITIALIZE COMMON FOR THE COMMON SYSTEM MODULE
C      PROGRAMS. IN PARTICULAR, THE COUNTER STATISTICS FOR EACH
C      SYSTEM MODULE MUST BE LOADED IN THIS PROGRAM.
```

```
      SUBROUTINE CCOBSY(NT,MPLACE)
C
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE: THIS SUBROUTINE, CALLED BY UTASK FOR EACH TASK NODE.
C           AT EACH TASK OCCURRENCE TIME, FIGURES OUT IF THE
C           OPERATOR TASK CUMULATIVE TIME STATISTICS NEED TO BE
C           MARKED OR RECORDED. IF SO THE APPROPRIATE ACTION
C           IS TAKEN.
C
C**INPUT PARAMETERS: NT - CURRENT TASK NODE NUMBER
C                    MPLACE - TASK OCCURRENCE TIME
```

```
      SUBROUTINE CHTDY(NPTRK,ICOL,VAL)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      CHTDY WILL CHANGE ONE ELEMENT OF TRACK DATA FOR A SPECIFIED
C      TRACK IN THE TRACK DATA DL'S OF ALL ADSM'S
C--INPUT PARAMETERS:
C      NPTRK-COLUMN IN NTRACK OF THE TRACK IN QUESTION
C      ICOL-ELEMENT OF THE TRACK DATA LIST FOR TRACK NPTRK TO
C           CHANGE
C      VAL-NEW VALUE FOR ELEMENT ICOL
```

```
      SUBROUTINE CLMEMY(IDOP)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       CLMEMY WILL CLEAR AN OPERATOR'S STACK OF ALL BUT THE
C       CURRENT TASK. THIS IS USEFUL TO DESTROY AN OPERATOR'S
C       MEMORY AFTER HE HAS BEEN INTERRUPTED.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
```

```
      SUBROUTINE COMEY(KODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       COMEY WILL PERFORM SPECIALIZED ERROR OUTPUT FOR
C       ERRORS GENERATED BY THE COMMON SYSTEM MODULE
C       PROGRAMS.  COMEY IS CALLED FROM UERR.
C--INPUT PARAMETERS:
C       KODE-ERROR CODE. CODES FOR THE COMMON SYSTEM
C            MODULE PROGRAMS ARE IN THE RANGE
C            7000-7999
```

```
      SUBROUTINE CSTATY(NCOP,ISTAT,XINC)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       CSTATY WILL RECORD OBSERVATIONS OF COUNTER STATISTICS.
C       COUNTER STATISTICS ARE SIMPLY ACCUMULATIONS OF VALUES
C       (USUALLY COUNTS) THAT OCCUR DURING A SIMULATION RUN.
C       THEY ARE REPORTED THROUGH THE MOPADS DATA BASE AT THE END
C       OF EACH RUN. NO STATISICAL MEASURES ARE GIVEN (E.G.
C       STANDARD DEVIATION) BECAUSE ONLY THE TOTAL VALUE IS
C       AVAILABLE FOR THE RUN.
C       EXAMPLES ARE A) NUMBER OF AIRCRAFT SHOT DOWN DURING A RUN,
C       AND B) CUMULATIVE IDLE TIME DURING A RUN.  SEE ALSO
C       BLOCKY,INITY,AND STDBY.
```

```
C--INPUT PARAMETERS:
C       NCOP-COPY ROW NUMBER OF THE COMPONENT TO RECORD AN
C            OBSERVATION FOR.
C       ISTAT-INDEX OF THE STATISIC TO BE RECORDED.
C       XINC-OBSERVATION TO RECORD.  ESSENTIALLY, CSTATY
C            ACCUMULATES THE XINC'S AS FOLLOWS:
C                  COUNT(ISTAT,NCOP)=COUNT(ISTAT,NCOP)+XINC
C            AND REPORTS COUNT(ISTAT,NCOP) AT THE END OF EACH
C            RUN.
```

```
      SUBROUTINE CVTKY(NCOPI,NCTRK,IYN)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       CVTKY WILL CHECK TO SEE IF THE VIEWERS OF A SPECIFIED AD
C       UNIT CAN SEE A PARTICULAR TRACK.
C--INPUT PARAMETERS:
C       NCOPI-COPY ROW NUMBER OF THE UNIT
C       NCTRK-COLUMN IN NTRACK OF THE TRACK
C--OUTPUT PARAMETERS:
C       IYN-YES/NO
C           1-YES, THE TRACK IS IN VIEW OF ONE OF THE UNITS VIEWERS.
C           2-NO, IT ISN'T
```

```
      SUBROUTINE DELTRY(NPTRK)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       DELTRY WILL DELETE A TRACK FROM THE TRACK DATA LIST OF
C       EVERY WORKING ADSM
C--INPUT PARAMETERS:
C       NPRTK-COLUMN NUMBER IN NTRACK OF THE TRACK TO DELETE
```

```
      SUBROUTINE ENDIT(NRUN)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       ENDIT IS CALLED AT THE END OF EACH ITERATION TO PERFORM
C       ALL END OF RUN PROCESSING.
C--INPUT PARAMETERS:
C       NRUN-RUN NUMBER JUST COMPLETED
```

```
      FUNCTION FLTY(NFCOL,NTCOL)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       FLTY WILL COMPUTE THE FLIGHT TIME OF THE IHAWK MISSILE
C       (MINUTES). FLTY DOES NOT CHECK TO SEE IF EITHER NFCOL OR
C       NTCOL IS VALID.
C--INPUT PARAMETERS:
C       NFCOL-NFSTOR COLUMN OF THE FIRE UNIT
C       NTCOL-NTRACK COLUMN OF THE TRACK
```

```
      SUBROUTINE GEVALY(IDOP,NADSM,NCOPI,NGS,IOPR,GS,NNG)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       GEVALY WILL EVALUATE AN OPERATOR'S GOAL STATE VECTOR
C
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NADSM-SYSTEM MODULE TYPE
C       NCOPI-COPY ROW NUMBER
C       NGS-ACTUAL LENGTH OF GS
C--INPUT/OUTPUT PARAMETERS:
C       IOPT(2)-DBAA OF THE OPERATOR STATE VECTOR
C--OUTPUT PARAMETERS:
C       GS(NGS)-GOAL STATE VECTOR
C       NNG-THE NUMBER OF GOALS THE OPERATORS OF THE SYSTEM HAVE.
C            (I.E. ONLY THE FIRST NNG ELEMENTS OF GS HAVE MEANINGFUL
C             INFORMATION). IF THE OPERATOR DOES NOT HAVE GOAL I,
C             THEN GS(I)=-1.E10.
```

```
      SUBROUTINE GPRIY(IDOP,NADSM,NCOPI,GS,NNG,IOPR,GSP)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       GPRIY WILL EVALUATE GOAL PRIORITIES GIVEN THE GOAL STATES
C       FOR AN OPERATOR.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NADSM-SYSTEM MODULE TYPE
C       NCOPI-COPY ROW NUMBER
C       GS(NNG)-CURRENT GOAL STATE VECTOR
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DATA OF THE OPERATOR STATE VECTOR
C--OUTPUT PARAMETERS:
C       GSP(NNG)-GOAL PRIORITY VECTOR. IF THE OPERATOR DOES NOT
C                HAVE GOAL I, THEN GSP(I)=0.

------------------------------------------------------------------------

      SUBROUTINE GSOURY(IDOP,NSNODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       GSOURY WILL RETURN THE SOURCE NODE WHERE A PARTICULAR
C       OPERATOR IS CREATED.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C--OUTPUT PARAMETERS:
C       NSNODE-SOURCE NODE NUMBER WHERE THE OPERATOR IS CREATED

------------------------------------------------------------------------

      SUBROUTINE INITY(NRUN)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       INITY WILL INITIALIZE THE COMMON SYSTEM MODULE PROGRAMS.
C
C--INPUT PARAMETERS:
C       NRUN-RUN NUMBER ABOUT TO BEGIN
C            0-INITY IS BEING CALLED PRIOR TO PERFORMING ANY RUN.
C            N-INITY IS BEING CALLED PRIOR TO RUN N.

------------------------------------------------------------------------

      SUBROUTINE INTLC
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       INTLC IS THE STANDARD SAINT UER INPUT PROGRAM FOR DOING
C       ANY REQUIRED USER INITIALIZATION. IT IS NOT USED BY
C       MOPADS. THIS IS A DUMMY PROGRAM. SEE SUBROUTINES
C       USTART AND INITY IN THIS SECTION.
```

```
      SUBROUTINE IROW1Y
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       IROW1Y WILL INITIALIZE ALL DATA LISTS IN WORKING ADSM'S OF
C       THE SIMULATION DATA SET THAT HAVE WORKING AND INITIAL VALUE
C       ROWS.  IN OTHER WORDS, SOME DL'S (E.G. OSV'S) HAVE THEIR
C       INITIAL VALUES STORED IN ROW 2. PRIOR TO EACH SIMULATION
C       RUN, THE VALUES ARE COPIED TO ROW 1. ROW 1 IS USED DURING
C       THE SIMULATION AND MAY BE CHANGED.
```

```
      SUBROUTINE IROW2Y(IADSM)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       IROW2Y WILL INITIALIZE ALL 2 ROW DATA LISTS IN A WORKING
C       ADSM. SEE IROW1Y FOR FURTHER DETAILS.
C       IROW2Y ALSO INITIALIZES THE RESOURCE DATA LIST.
C--INPUT/OUTPUT PARAMETERS:
C       IADSM(4)-DRAA OF THE ADSM TO INITIALIZE.
```

```
      SUBROUTINE MODRF(MFN,MNODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       MODRF IMPLEMENTS THE SAINT MODERATOR FUNCTIONS FOR MOPADS.
C       THE MODRF CODES ARE NOT RENUMBERED FOR EACH SYSTEM MODULE.
C       IN OTHER WORDS, ONLY ONE STREAM OF MODRF CODES IS USED FOR
C       ALL SYSTEM MODULES.
C--INPUT PARAMETERS:
C       MFN-MODRF CODE
C           CODE 1 IS RESERVED FOR THE HUMAN FACTORS MODERATOR
C           FUNCTION MODULE
C       MNODE-CURRENT NODE NUMBER
```

```
      SUBROUTINE MODRFY(MNODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       MODRFY IMPLEMENTS THE MOPADS MODERATORS FOR TIME-TO-PERFORM
C       OPERATOR TASK ELEMENTS BY CALLING THE APPROPRIATE HUMAN
C       FACTORS MODULE PROGRAMS.
C--INPUT PARAMETERS:
C       MNODE-CURRENT MODE
```

```
      SUBROUTINE MXMSGY(MSID,NCOL,IUB,MCOL)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       MXMSGY WILL FIND THE HIGHEST PRIORITY MESSAGE WHOSE PRIORITY
C       IS LESS THAN IUB.  THUS, MXMSGY CAN BE USED TO FIND THE
C       SECOND, THIRD, ETC. MOST IMPORTANT MESSAGE.
C--INPUT PARAMETERS:
C       MSID(5,NCOL)-ID'S OF MESSAGES
C       IUB-UPPER BOUND ON MESSAGE PRIORITY. ANY MESSAGE WITH
C           PRIORITY .GE. IUB WILL BE IGNORED IN THE SEARCH.
C--OUTPUT PARAMETERS:
C       MCOL-COLUMN NUMBER IN MSID OF THE HIGHEST PRIORITY MESSAGE.
```

```
      SUBROUTINE NEWTKY(NPTRK)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       NEWTKY WILL PUT A NEW TRACK ENTRY INTO EVERY WORKING ADSM'S
C       TRACK DATA BL. IF THE TRACK IS NOT IN RADAR COVERAGE,
C       RETURN WITH NO ACTION.
C--INPUT PARAMETERS:
C       NPTRK-COLUMN IN NTRACK WHERE THE TRACK IS STORED
```

```
      SUBROUTINE OEVALY(IOOP,NADSM,NCOPI,OS,NNS,JTASK,IOPR,OSJ,TJ,*)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       OEVALY WILL COMPUTE THE EXPECTED GOAL STATE VECTOR FOR A
C       PARTICULAR OPTION FOR A SPECIFIC OPERATOR
```

```
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       MADSN-SYSTEM MODULE TYPE
C       MCOPI-COPY ROW NUMBER
C       GS(NNG)-CURRENT GOAL STATE VECTOR
C       JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OEVALY WILL
C             EVALUATE EXPECTED GOAL STATES GIVEN THAT THE
C             OPERATOR PERFORMS TASK JTASK.
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-BBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM PERFORMING
C                TASK JTASK
C       TJ-EXPECTED TIME (MINUTES) TO PERFORM TASK JTASK.  IF THE
C          OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C          THEN TJ=-1.
C--ALTERNATE RETURNS:
C       1-JTASK IS GREATER THAN THE MAXIMUM TASK NUMBER THAT THE OPERATOR
C         PERFORMS.  OEVALY WILL BE CALLED REPEATEDLY WITH GREATER
C         VALUES OF JTASK UNTIL THIS CONDITION OCCURS.

------------------------------------------------------------------------

      SUBROUTINE OPREY(IOPP)
C
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE: ERROR CHECKING TO ASSURE THAT NON-OPERATORS ARE
C**          NOT PROCESSED AS OPERATORS
C
C**INPUT PARAMETER: IOPP - ENTITY ID
C

------------------------------------------------------------------------

      FUNCTION PRIOR(ITASK)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       PRIOR IS WRITTEN TO SATISFY CALLS BY SAINT. IT IS NOT
C       USED BY MOPADS. THIS FUNCTION IS A DUMMY.
C--INPUT PARAMETERS:
C       ITASK-TASK NODE NUMBER
```

```
      SUBROUTINE PSHNTY(IOPT,IBOP,TSKNEW,LEN)
C
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C**PURPOSE:  THIS SUBROUTINE IS USED TO REMOVE (POP) THE CURRENT TASK
C            FROM THE OPERATOR TASK STACK.IT THEN PLACES A SET OF
C            TASKS TO BE PERFORMED ONTO THE TASK STACK.THE TASKS ON
C            THE TASK STACK WILL BE PERFORMED ACCORDING TO THE LIFO
C            SERVING DISCIPLINE.
C
C**INPUT PARAMETERS:   IOPT=OPTION TO CLEAR THE OPERATOR'S MEMORY
C                           (1=CLEAR MEMORY,0=DO NOT CLEAR MEMORY)
C                      IBOP=OPERATOR ID
C                      TSKNEW(5,LEN)=ARRAY CONTAINING TASK DATA
C                      LEN=NUMBER OF TASKS TO ADD TO TASK STACK
C
```

```
      FUNCTION PSY(PSCN,TF)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      PSY WILL COMPUTE THE SUCCESS PROBABILITY FOR A TASK OR
C      A TASK ELEMENT.
C--INPUT PARAMETERS:
C      PSCN-NOMINAL PROBABILITY OF SUCCESS
C      TF-OPERATOR'S TASK ERROR FACTOR OR TASK ELELMENT ERROR
C         FACTOR
C--METHOD
C      LET DELTA=(1-PSCN)*TF
C      THEN PSY=MAX(.1,PSCN+DELT)
C
```

```
      SUBROUTINE RNGY(NTCOL,IBOP,RANGE,IYN)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      RNGY WILL DETERMINE IF A SPECIFIED TRACK IS VISIBLE TO AN
C      OPERATOR,  IN OTHER WORDS IT WILL CHECK IF IT IS WITHIN
C      HIS SCREEN RANGE.  RNGY DOES NOT CHECK TO SEE IF THE
C      TRACK IN ALSO VISIBLE TO A VIEWER.
```

```
C--INPUT PARAMETERS:
C       NTCOL-THE COLUMN NUMBER OF THE TRACK IN NTRACK
C       IDOP-THE OPERATOR'S ID
C       RANGE-THE SCREEN RANGE OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       IYN-YES/NO
C           1-YES, THE OPERATOR CAN SEE THE TRACK
C           2-NO, HE CAN'T

------------------------------------------------------------------------

      SUBROUTINE SHELLY(N,DATA,NDEX)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       SHELLY WILL SORT DATA LARGEST TO SMALLEST WITH A SHELL SORT.
C       DATA IS NOT ACTUALLY DISTURBED. NDEX IS MADE INTO A CROSS
C       REFERENCE TO DATA IN THE CORRECT SORTED ORDER.
C--INPUT PARAMETERS:
C       N-LENGTH OF DATA AND NDEX
C       DATA(N)-ARRAY TO BE SORTED
C--OUTPUT PARAMETERS:
C       NDEX(N)-INDEX ARRAY TO DATA. ON OUTPUT THE I-TH LARGEST
C               ELEMENT OF DATA WILL BE DATA(NDEX(I))

------------------------------------------------------------------------

      SUBROUTINE SMSGY(MROWL,MREC,MOPR,IDIST,NVAR,MSNO)
C
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE: TO HANDLE SENDING OF ALL MESSAGES FROM USER CODE.  ALSO
C**         STORES IN A LOCAL ARRAY ALL STATIC MESSAGE CHARACTERISTICS.
C**         THIS ARRAY, NMMSG, MUST BE CHANGED IF ANY OF THESE
C**         MESSAGE CHARACERISTICS ARE CHANGED BY AN ANALYST OR IF
C**         ANY NEW MESSAGES ARE ADDED.
C
C**INPUT PARAMETERS: MROWL - MESSAGE ROW NUMBER (ROW IN NMMSG)
C                    MREC - RECEIVER COPY ROW NUMBER
C                    MOPR - RECEIVER OPERATOR TYPE (UNUSED IF COLUMN
C                           1 OF NMMSG IN ROW MROWL IS NOT 0)
C                    IDIST - MESSAGE DISTRIBUTION CODE (FOR CODE
C                            DEFINITIONS SEE SUBROUTINE SNDMSA)
C                    NVAR - ARRAY OF LENGTH 3 CONTAINING THE VARIABLE
C                           MESSAGE. COLUMNS 11-13 IN NMMSG CONTAIN
C                           THE CODES DEFINING THE VARIABLE MESSAGE.
C
```

```
C**OUTPUT PARAMETERS: MSNO - THE MESSAGE SEQ NUMBER ASSIGNED TO THE MESS
C                           BY SUBROUTINE SNDXSA
C
C**LOCAL VARIABLES: NNMSG(50,15) - THIS ARRAY CONTAINS STATIC
C                                  INFORMATION FOR EACH MESSAGE TYPE.
C                                  EACH FUNCTIONAL TYPE - SUBTYPE
C                                  COMBINATION HAS A ROW IN THE ARRAY.
C                                  THE MESSAGE ROW NUMBER THEN REFERS
C                                  TO A UNIQUE MESSAGE TYPE AND GIVES
C                                  THE ROW CONTAINING INFO FOR THAT
C                                  MESSAGE. COLUMN DEFINITIONS ARE
C                                  GIVEN BELOW.
C
C            COLUMN DEFINITIONS FOR THE NNMSG ARRAY
C            --------------------------------------
C            1 - RECEIVER OPERATOR TYPE (0 IF MUST BE CALCULATED)
C            2 - FUNCTIONAL TYPE
C            3 - MESSAGE SUBTYPE
C            4 - MESSAGE PRIORITY
C            5 - COMMUNICATION NETWORK (1-VOICE, 2-ATDL)
C            6 - ACKNOWLEDGMENT REQUIRED (1-YES, 2-NO)
C            7 - ATDL CODE (UNUSED)
C            8 - UNUSED
C            9 - UNUSED
C            10 - NO. OF WORDS IN THE VARIABLE MESSAGE (NOT COUNTING IT
C            11-13 - VARIABLE MESSAGE TYPE CODES (FOR NVAR WORDS 1-3)
C                    (THESE CODES ARE USED TO CHECK THE VALUES OF
C                     NVAR TO ASSURE THAT LEGAL VALUES WERE PASSED IN.
C                     THE CODES ARE DEFINED BELOW WITH LEGAL VALUES
C                     SHOWN IN PARENTHESES)
C
C            CODES FOR NNMSG COLUMNS 11-13
C            -----------------------------
C              1 - WHICH FIRE SECTION (0-EITHER,1-A,2-B,3-BOTH)
C              2 - TRACK COLUMN POINTER (1 TO NCOLS)
C              3 - MESSAGE NUMBER ACKNOWLEDGING (1 TO LASTM)
C              4 - FU EFFECTIVE STATUS (1-KILL,2-NO KILL)
C              5 - IHIPIR STATUS (1-LOCK,2-NO LOCK)
C              6 - RAID SIZE (1-ONE,1-FEW,3-MANY)
C              7 - IHIPIR LOCKING METHOD (1-MANUAL,2-AUTO)
C              8 - METHOD OF FIRE (1-SHOOT-LOOK-SHOOT,2-RIPPLE FIRE)
C
C            14 - TASK NODE NUMBER SENT FROM (0 IF VARIABLE, BECAUSE
C                 THIS INFORMATION IS NOT REALLY NEEDED TO SEND THE
C                 MESSAGE)
C            15 - USAGE STATUS =1 MESSAGE CURRENTLY BEING USED BY AT LE
C                                 ONE SYSTEM MODULE
C                              -1 MESSAGE NOT BEING USED
```

```
      SUBROUTINE STATE
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C
C**PURPOSE:  THIS IS SUBROUTINE STATE FOR THE MOPADS MSAINT SOFTWARE.
C            IT IS USED TO UPDATE THE POSITIONS OF THE AIRCRAFT IN THE
C            AIR SCENARIO.
```

```
      SUBROUTINE STDBY(NRUN)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C      STDBY WILL DUMP COUNTER STATISTICS IN THE MOPADS DATA BASE
C      AFTER A RUN.  STDBY DOES NOT RE-INITIALIZE THE COUNTER
C      STATISTICS ARRAYS AFTER A DUMP.
C--INPUT PARAMETERS:
C      NRUN-RUN NUMBER JUST COMPLETED
```

```
      SUBROUTINE THR1Y(IDOP,NCOPI,KTRK,XLB,TOA,KPTRK,IST)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C
C      THR1Y WILL LOCATE THE TRACK THAT IS MOST THREATENING TO AN
C      AIR DEFENSE COMPONENT. THIS SUBPROGRAM IS USED TO EVALUATE
C      THE SELF DEFENSE GOAL. THR1Y WILL ALSO EVALUATE
C      A SINGLE TRACK BASED ON THE VALUE OF KTRK.
C--INPUT PARAMETERS:
C      IDOP-THE OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE COMPONENT
C      KTRK-TRACK COLUMN NUMBER IN NTRACK
C           0-EVALUATE ALL TRACKS
C           K-EVALUATE TRACK K ONLY
C      XLB-LOWER BOUND ON TOA.  THE TRACK WITH THE MINIMUM TIME
C          OF ARRIVAL BUT GREATER THAN XLB WILL BE RETURNED.  XLB.GT.0
C          IS USED TO EXCLUDE THE MOST THREATENING TRACK WHEN
C           ESTIMATING THE IMPACT ON GOALS OF ASSIGNING THE MOST
C           THREATENING TRACK TO AN FU.
```

```
C--OUTPUT PARAMETERS:
C       TOA-TIME OF ARRIVAL OF THE MOST THREATENING TRACK
C       KPTRK-NTRACK COLUMN OF THE MOST THREATENING TRACK
C       IST-TRACK STATUS(0-NO TRACK,1-UNIDENTIFIED,2-IDENTIFIED AS HOSTIL
C                        OR UNKNOWN,3-2NDARY ASSIGNED)

------------------------------------------------------------------------

      SUBROUTINE THR2Y(IDOP,NCOPI,KTRK,XLB,TOA,KPTRK,IST)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C
C       THR2Y WILL LOCATE THE TRACK THAT IS MOST THREATENING TO AN
C       AIR DEFENSE COMPONENT'S PROTECTED SITES. THIS SUBPROGRAM IS
C       USED TO EVALUATE THE GOAL OF PROTECTING CRITICAL ASSETS.
C       THR2Y WILL ALSO EVALUATE ONLY ONE TRACK BASED ON THE
C       VALUE OF KTRK.
C--INPUT PARAMETERS:
C       IDOP-THE OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE COMPONENT
C       KTRK-TRACK COLUMN NUMBER IN NTRACK
C            0-EVALUATE ALL TRACKS
C            K-EVALUATE TRACK K ONLY
C       XLB-LOWER BOUND ON TOA.  THE TRACK WITH THE MINIMUM TIME
C           OF ARRIVAL BUT GREATER THAN XLB WILL BE RETURNED.  XLB.GT.0
C           IS USED TO EXCLUDE THE MOST THREATENING TRACK WHEN
C           ESTIMATING THE IMPACT ON GOALS OF ASSIGNING THE MOST
C           THREATENING TRACK TO AN FU.
C--OUTPUT PARAMETERS:
C       TOA-TIME OF ARRIVAL OF THE MOST THREATENING TRACK
C       KPTRK-NTRACK COLUMN OF THE MOST THREATENING TRACK
C       IST-TRACK STATUS(0-NO TRACK,1-UNIDENTIFIED,2-IDENTIFIED AS HOSTIL
C                        OR UNKNOWN,3-2NDARY ASSIGNED)

------------------------------------------------------------------------

      SUBROUTINE TOAY(POS,NTR,ETA)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GIVEN A POINT AND A TRACK, TOAY WILL COMPUTE THE GROUND
C       DISTANCE BETWEEN THEM AND THE TIME TILL THE AIRCRAFT
C       ARRIVES AT THE POINT IF IT TURNS IMMEDIATELY TOWARD IT
C       (USING THE AIRCRAFT'S CURRENT SPEED). IF THE AIRCRAFT IS
C       OUTBOUND, AN INFINITE TIME WILL BE RETURNED.
C--INPUT PARAMETERS:
C       POS(3)-X,Y,Z OF THE POINT (Z-FEET, X AND Y-N.MI.)
C       NTR-COLUMN IN NTRACK OF THE TRACK
```

```
C--OUTPUT PARAMETERS:
C       ETA(2)-(1)=TIME TILL ARRIVAL. IF THE CURRENT HEADING OF THE
C                  TRACK IN OUTBOUND FROM THE POINT, THEN ETA(1)=
C                  1.E10.
C               (2)=GROUND DISTANCE(N.MI.) FROM THE POINT TO THE
C                  TRACK.
```

```
      SUBROUTINE TOTY(IDOP)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       TOTY WILL COMPUTE THE TIME-ON-TASK FOR AN OPERATOR AND UPDATE
C       HIS OSV. TOTY IS COMPUTED AS (TNOW-TSTRTK)/60+(THE INITIAL
C       VALUE OF TIME ON TASK)
C--INPUT PARAMETERS:
C       IDOP-THE OPERATOR'S ID
```

```
      SUBROUTINE TSEQY(IDOP,NXTAS,KASE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       TSEQY WILL SELECT THE NEXT TASK FOR AN OPERATOR.  IT IS
C       CALLED TO INITIATE THE TASK SEQUENCING PROCEDURE.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C--OUTPUT PARAMETERS:
C       NXTAS-NUMBER OF THE NEXT OPERATOR TASK TO PERFORM. IF NO
C             TASK WILL REDUCE THE OPERATOR'S OBJECTIVE FUNCTION,
C             NXTAS=0
C       KASE-OUTPUT CONDITION
C            1-NXTAS IS THE NEXT TASK ON THE STACK.
C            2-NXTAS IS NOT FROM THE STACK. THE CALLING PROGRAM
C              NEEDS TO LOAD THE STACK WITH APPROPRIATE TASK
C              NUMBER(S).
```

```
      SUBROUTINE UACCPT
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       UACCPT IS NOT USED BY MOPADS. THIS PROGRAM IS A
C       DUMMY TO SATISFY SAINT EXTERNAL REFERENCES.
```

```
      SUBROUTINE UERR(KODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       UERR WILL PERFORM ADDITIONAL MOPADS ERROR PROCESSING ON
C       A SYSTEM MODULE SPECIFIC BASIS.
C--INPUT PARAMETERS:
C       KODE-ERROR CODE GENERATED FROM SYSTEM MODULE SOFTWARE
```

```
      SUBROUTINE UINPT
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       UINPT IS CALLED ONCE PRIOR TO THE START OF A MOPADS
C       SIMULATION TO COMPLETE INITIALIZATION OF MOPADS ARRAYS
C       AND THE DATA BASE.
C
```

```
      SUBROUTINE USCOND(IHON)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       USCOND IS NOT USED BY MOPADS. THIS PROGRAM IS A DUMMY
C       TO SATISFY SAINT EXTERNAL REFERENCES.
```

```
      FUNCTION USERF(ICODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       USERF EVALUATES THE MOPADS EXTENSION USER FUNCTIONS
C--INPUT PARAMETERS:
C       ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C       USERF-VALUE OF THE USER FUNCTION
```

```
      FUNCTION USERIN(ICODE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       USERIN EVALUATES THE MOPADS INPUT USER FUNCTIONS
C--INPUT PARAMETERS:
C       ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C       USERIN-VALUE OF THE USER FUNCTION

      SUBROUTINE USTART(NRUN)
C--MODULE: MOPADS CONTROL SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.19
C--PURPOSE:
C       USTART IS CALLED AT THE START OF EACH SIMULATION RUN BY
C       MSAINT TO PERFORM ANY INITIALIZATION NEEDED.
C--INPUT PARAMETERS:
C       NRUN-RUN NUMBER ABOUT TO BEGIN.

      SUBROUTINE UTASK(NT,NPLACE)
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C
C
C**PURPOSE: TO BE THE MASTER PROCESSOR OF USER CODE FOR ALL
C**          SYSTEM MODULES.  CALLS A ROUTINE FOR EACH SYSTEM
C**          MODULE, WHICH CALLS TASK MODE-SPECIFIC ROUTINES
C**          FOR ALL TASKS REQUIRING UNIQUE PROCESSING.
C
C**INPUT PARAMETERS: NT - TASK NUMBER
C**                  NPLACE - TASK OCCURANCE TIME
C**                         = 1 FIRST PREDECESSOR COMPLETION
C**                             (ONLY IF DIFFERENT FROM THE RELEASE
C**                              TIME.  THIS NORMALLY WON'T BE THE CASE)
C**                         = 2 RELEASE TIME
C**                         = 3 START TIME (SAME AS RELEASE TIME IF
C**                             THE TASK REQUIRES NO RESOURCES)
C**                         = 4 COMPLETION TIME
C**                         = 5 CLEARING TIME ( A TASK CANNOT BE
C**                             BOTH COMPLETED AND CLEARED)
C
```

```
      SUBROUTINE WHTIDY (NEGNUM)
C
C--MODULE: MOPADS COMMON USER CODE
C--REFERENCE: MOPADS VOLUME 5.19
C**PURPOSE:  THIS SUBROUTINE IS USED TO ASSIGN OPERATOR ID NUMBERS TO
C            CONTOL ENTITIES CREATED DURING THE SIMULATION.
C
C**OUTPUT PARAMETERS:  NEGNUM=NEGATIVE OPERATOR ID REPRESENTING THE
C                              CONTROL ENTITY
C
```

6-0 USER INSTRUCTIONS

The subprograms in the CSMP fall into several groupings based on their functions:

1. Track Processing.

CHLTY	-	changes elements of the Track Data data list in the data base
CVTKY	-	checks if an air defense unit viewers can see a track
DELTRY	-	deletes a track from all Track Data data lists
NEWTRY	-	inserts a new track into all Track Data data lists
RNGY	-	determines if a track is visible to an operator
FLTY	-	computes missile flight time

2. Task Sequencing.

GEVALY	-	evaluates an operator's goals
GPRIY	-	evaluates goal priority functions
MXMSGY	-	finds an operator's highest priority outstanding messages
OEVALY	-	computes expected goal states for various operator tasks
THR1Y,THR2Y	-	locates high threat tracks
TOAY	-	computes a track's threat
TSEQY	-	selects next task for an operator

3. Statistics. -

CCOBSY	-	marks time for MOPADS Operator Task Fraction Statistics.
CSTATY	-	records observations of count statistics

STDLY	-	dumps count statistics to the data base

4. Initialization.

BLOCKY	-	block data
GSOURY	-	find source node for an operator
INITY, IROW1Y, IROW2Y	-	initialization of arrays

5. Operator Processing.

CLMEMY	-	clear an operator's memory
MODRFY	-	human factors moderator function access
PSHMTY	-	manage an operator's memory stack
PSY	-	compute success probability for a task
SMSGY	-	send messages
TOTY	-	compute operator's time-on-task

6. MSAINT User Called Programs.

ENDIT, INTLC, MODRF, PRIOR, STATE, UACCPT, UERR, UINPT, USCOND, USERF, USERIN, USTART, and UTASK	-	see Walker (1983) for details

7. Miscellaneous.

COMEY	-	error processing
OPREY	-	error checking
SHELLY	-	perform a shell sort on an array
WITIDY	-	assign an ID to a control entity

7-0 ERROR PROCESSING

When appropriate, CSMP programs use the DBAP error processing, ERRORA, for error processing. At other times, the standard MSAINT program SERR is used. Error codes 7000 to 7999 have been reserved for use by the SMCP. Defined codes are as follows:

Error Code	Error Conditions	Subprogram(s)
7000	Invalid System Module Type	USERF,USERIN
7001	Invalid Copy Row Number	CSTATY
7002	Invalid Count Statistics Index	CSTATY
7003	Screen range evaluation requested for an operator for whom that information is not defined	RNGY
7004	Action requested for an operator in an inactive unit	RNGY
7005	NTRID/NTRID2 arrays full	NFST3A,NCOPYA, NFST2A
7006	NSITE array full	NFST3A,NCOPYA, NFST2A
7007	NFSTOR full	NCOPYA
7008	A system module type that cannot be classified as a fire unit or a non-fire unit has been encountered	INITY,NCOPYA, NFST3A,THR2Y
7009	NDBAD array full	NCOPYA,NFST2A
7010	too many operators	NCOPYA
7011	too many operators in one system module	NCOPYA
7012	local array IDATA must be increased to MNRW	NFST2A
7013	too many protected sites for IHAWK	NFST2A
7014	too many display parameters	INITY
7015	Invalid System Module type	GEVALY,OEVALY

7016	Attempt to evaluate operator goals	GEVALY,OEVALY
7017	Variable NNG is greater than the number of operator goals	GPRIY
7018	JTASK.LE.0	OEVALY
7019	Invalid moderator function code	MODRF
7020	Incorrect distribution type for moderator function	MODRFY
7021	Invalid skill number	MODRFY
7201	Invalid message row number	SMSGY
7202	Illegal receiver copy row number	SMSGY
7203	Illegal receiver operator ID	SMSGY
7204	Illegal variable message value	SMSGY
7205	Non-operator passed as operator	various
7207	Attempt to send an inactive message	SMSGY
7208	Skill weights do not sum to 1.0 or 100	MODRFY

8-0 COMMON VARIABLE DEFINITIONS

Two labeled COMMON areas are used by the CSMP. /COM1Y/ contains numeric data and /COM2Y/ contains CHARACTER data. In addition, certain FORTRAN 77 PARAMETER's are used with these COMMON areas.

Variable	Definition	Value	COMMON
MTASKY	Maximum number of operator tasks for any operator	40	PARAMETER
MXSTAY	Maximum number of count statistics for any system module	50	PARAMETER
CLABY (MXSTAY,4) * 25	Labels of count statistics CLABY (I,J) is the label of statistic I for system module type J	-	/COM2Y/
COUNT (MXSTAY, 30)	Storage for count statistics COUNTY(I,J) - count for statistic I, copy J	-	/COM1Y/

DVWY	Maximum time between general viewer updates of tracks	0.25	/COM1Y/
MAXMSY	Maximum row dimension of array NNMSG in SUBROUTINE SMSGY	34	/COM1Y/
MAXOPY	Maximum operator type	9	/COM1Y/
MXSTY(4)	Number of count statistics defined for each system module type	-	/COM1Y/
TASKY(150, MTASKY)	TASKY(I,J) is the cumulative time operator I spends doing task J	-	/COM1Y/
TTFNY(30)	End time for each copy to compute task fraction statistics. Usually end of run time, but may be earlier if the unit was destroyed by enemy action	-	/COM1Y/
TVWNY	Time of the next general viewer update of tracks		/COM1Y/

IX. REFERENCES

Polito, J. MOPADS data base application program (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Polito, J. Creating MOPADS air scenarios (MOPADS Vol. 4.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Polito, J. The MOPADS data base control system (MOPADS/DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d)

Polito, J. & Goodin II, J. R. MOPADS utility programs (MOPADS/UTIL) (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Walker, J. L. MSAINT user's guide: changes and additions to the SAINT user's manual (MOPADS Vol. 4.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

X. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 8345
Albuquerque, New Mexico 87198

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.#1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Caispan Corporation
1327 Spruce St., Rm. 108
Boulder, Colorado 80301

XI. CHANGE NOTICES

APPENDIX B

MOPADS FINAL REPORT:

USER GUIDE FOR THE IHAWK SYSTEM MODULE

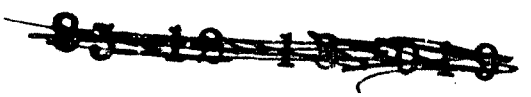


TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

LIST OF FIGURES

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology and Abbreviations

ASO	Azimuth Speed Operator
BCC	Battery Control Central
FCO	Firing Console Operator
PCP	Platoon Command Post
TCA	Tactical Control Assistant
TCO	Tactical Control Officer

I. INTRODUCTION

1-0 PURPOSE

This document is a user manual for the IHAWK system module. It contains details of the implementation of the MOPADS model of the IHAWK system. This information is found nowhere else in the MOPADS documentation.

In particular, the contents of the operator state vectors, environmental state vectors, and task data are given in this report. Section II contains a brief description of the IHAWK missile system. Section III presents the contents of the operator state vectors. The environmental state vector is described in Section IV. The operator tasks and task sequencing assumptions are given in Section V. Finally, Section VI describes assumptions used in developing the module.

2-0 INTENDED AUDIENCE

MOPADS users and MOPADS modelers should read this report. The language and level of detail used in the discussion is oriented to the MOPADS user, however. Programming and implementation details are not included here. That information is discussed in detail in MOPADS volumes four and five.

3-0 OTHER READING

The reader is presumed to be familiar with the MOPADS system. This implies that he/she should have read the final report, Polito & Laughery(1983), prior to reading this document. In addition, the reader should be familiar with the main MOPADS user manual, Polito (1983a). More technical data on the IHAWK system module is found in Goodin & Walker 1983.

II. SYSTEM DESCRIPTION

The IHAWK missile battery is comprised of the the BCC, IFF equipment, several radars (IPAR, IHIPIR,CWAR) and missiles and launchers. The primary mission of the IHAWK missile battery is to protect vital assets such as airfields, ammunition supply centers, etc., from enemy air attacks. The IHAWK has the ability to detect, identify, track, and engage aircraft flying at various altitudes.

The various radars serve different operations for the IHAWK. The IPAR is used to detect medium to high altitude aircraft. The CWAR is used to detect high-speed low flying aircraft. The IHIPIR tracks targets that are being engaged. The IHAWK battery has two firing sections (A and B), each with its own set of three launchers. Each launcher is capable of holding three missiles that are ready to launch. Additional missiles are kept on pallets near the launchers so that they may be reloaded once they have expended all of their "hot" missiles. The launchers are capable of firing one missile at a time.

The BCC is responsible for control of the IHAWK missile battery's firing operations. There are five operators manning the BCC. The following forms describe these operators, their missions, and their duties.

The IHAWK system module represents the operators of the Improved Battery Control Central (IBCC) unit. In the following forms, the MOPADS operator type is shown in the column labeled OPERATOR NUMBER, and the MOPADS label for each operator is shown in parentheses in the column labeled OPERATOR TITLE.

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
3	TACTICAL CONTROL OFFICER (TCO)	Operates the Tactical Control Console with the TCA (Tactical Control Assistant). He oversees the entire operation of the BCC. He performs all voice communication with Battalion and processes all commands sent to the IHAWK BCC. He is responsible for the control of the number of missiles to be used to engage aircraft and he has the final authority over whether to fire missiles at a target or not.	Tactical Control Console (TCC), TCO Panel Assembly, BCC Status Indicator, Tracking Lever and Tracking Lever Designate Panel, Communication Network
4	TACTICAL CONTROL ASSISTANT(TCA)	Assists the Tactical Control Officer at the Tactical Control Console. He is responsible for initiating IFF with targets and placing identifying information on the TCC CRT with a grease pencil.	Tactical Control Console (TCC), TCA Panel Assembly, BCC Status Indicator
5	ASIMUTH SPEED OPERATOR (ASO)	Detection of CWAR targets at the CWTDC. These targets are low-altitude,high-speed aircraft that are typicall not visible to Battation radars and IPAR radar in the IHAWK.	Continuous Wave Target Detection Console (CWTDC)

NAME: J.Riley Goodin II
DATE: 1/28/84

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR DEFINITIONS

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
6	FIRING CONSOLE OPERATOR A (FCO-A)	Tracks targets, estimates raid size, fires missiles, and evaluates target intercept for Fire Section A.	Firing Console (FC)
7	FIRING CONSOLE OPERATOR B (FCO-B)	Same as FCO A except for Fire Section B.	Same as FCO-A

NAME: J.Riley Goodin II
DATE: 1/28/84

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR DEFINITIONS

Page 2 of 2

III. OPERATOR DATA

1-0 TACTICAL CONTROL OFFICER (TCO)

1-1. Human Factors Parameters.

Table III-1 shows the default values for the human factors parameters of the Tactical Control Officer (TCO). This data is part of the operator state vector for the TCO. Discussions of these variables are contained in Polito & Laughery (1983), Laughery (1983), and Polito (1983b).

The last elements of this list, X-SCREEN-CENTER, Y-SCREEN-CENTER, and SCREEN-RANGE, are set when the system module is copied into a command and control configuration.

Table III-1. TCO Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
3.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
1.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
6.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO

Table III-1 (continued)

0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.6250000E-01	TARGET-HEIGHT
0.6250000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
480.0000	DISPLAY-RESOLUTION
0.1000000	DISPLAY-BACKGROUND-HEIGHT
0.1000000	DISPLAY-BACKGROUND-WIDTH
0.1000000	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

1-2. Goals, Goal Priority Functions, and Objective Parameters.

There are eight goals defined for the IHAWK operators. Figure III-1 explains the goals and the goal states. Not all operators have all goals. The TCO has goals 1, 2, 3, 5, 6, and 7. The first three goals involve assessing the threat priority of tracks according to the criteria specified in Table III-1. Goal one is effective only if the IHAWK is working with a Battalion AN/TSQ-73 which is assigning tracks to it.

Each track is evaluated to calculate its time to arrive at the IHAWK and at the protected site. The minimum of these two times is the threat for the track. The TCO may elect to attack a track other than the one assigned by the Battalion if it becomes a high enough threat.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
1	Engage Assigned Tracks	The minimum time for any not receding, in-range, assigned, but not engaged Track to arrive at the IHAWK or its protected site if it immediately turned intound.
2	Self Defense	The minimum time for any hostile or unknown, not receding, in-range, unassigned Track to arrive at the IHAWK if it immediately turned inbound.
3	Protect Critical Assets	The minimum time for any hostile or unknown, not receding, in-range, unassigned Track to arrive at any protected site if it immediately turned inbound.
4	Minimize the number of unidentified Tracks.	The number of unidentified tracks.
5	Minimize the maximum priority of outstanding messages	The maximum priority of any outstanding message that the operator may receive and no one else is processing.

NAME: Joseph Polito
DATE: 8/25/83
AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 1 of 2

Figure III-1. IHAWK Operator Goals.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
6	Maximize the number of missiles available	The total number of hot and cold missiles available.
7	Protect friends	The number of friendly tracks being engaged.
8	Minimize the number of unidentified, low altitude tracks.	The number of unidentified, low altitude tracks.

NAME: Joseph Polito | AIR DEFENSE SYSTEM MODULE: IHAWK
DATE: 8/25/83 | PROJECT: MOPADS

OPERATOR GOALS DEFINITIONS Page 2 of 2

Figure III-1 (continued)

Goal five represents the action of the operators to respond to communications from other members of the IHAWK battery and from Battalion. The MOPADS software uses "messages" to represent these communications, and each message has an associated priority.

The IHAWK will not attack low threat targets if it is low on missiles. Goal six provides this disincentive to to engage when the threat is not high.

Pop-up targets that appear as high threat targets on CWAR may be engaged before being identified. Goal seven will cause the TCO to break off an engagement if a currently engaged target is determined to be friendly.

Goal priority functions must be specified for the goals which specify how the goal priority changes with differing values for the goal states. As discussed in Polito & Laughery (1983b) and Polito (1983c), this is done by specifying two points on the goal pricrity function above and below the range of satisfaction. This data for the TCO is shown in Figure III-2.

Table III-2 shows how this data appears in the operator state vector. Each goal is represented by 15 values the first of which is the goal number. If this number is negative (e.g., goal four) then the operator does not have that goal. The data shown in Figure III-2 is given in the elements LITTLE-M, BIG-M, and elements GOAL-STATE-LOW-1 through PRIORITY-HIGH-2. The values of LITTLE-A, LITTLE-B, BIG-A, and BIG-B are calculated. See Polito (1983a) for warnings on changing these numbers. Infinity is represented in the operator state vectory by the number 10^{10}.

Objective function parameters for the TCO are contained in his operator state vector, Table III-1. The objective function is specified as "1", which means that the TCO selects the task which offers the greatest improvement in his goal state. If objective function "2" is specified, the TCO attempts to obtain the greatest goal improvement per unit time. Finally, the TCO considers all of his goals when selecting his next task (i.e., GOALS-CONSIDERED is six).

1-3. Display Data.

Table III-3 is the display data for the TCO. As can be seen from the table, only three elements of display data are used by the current model. Screen range is 43 miles. The coordinates of the center of the TCO's screen are not specified until the system module is copied into a command and control configuration. At that time, the user may specify the position of the center of the TCO's viewing area. If the user fails to specify this data (by editing the operator state vector), then MOPADS will set it equal to the position of the IHAWK.

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
1	4, ∞	1.3	10	2.5	5	-	-	-	-
2	4, ∞	1.3	10	2.0	5	-	-	-	-
3	4, ∞	1.3	9.5	2.0	4.5	-	-	-	-
5	0,0	-1	1	-2	2	1	1	2	2
6	3, ∞	0	5	2.5	5	-	-	-	-
7	-∞,0	-	-	-	-	1	9.8	2	9.9

OPERATOR: TCO/3

NUMBER OF GOALS CONSIDERED: 6

NAME: POLITO
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION Page 1 of 1

Figure III-2. Goal Priority Function Data for the TCO.

Table III-2. TCO Goal Pricrity Function Parameters.

1.000000	GOAL
4.000000	LITTLE-M
0.1000000E+11	BIG-M
3.099663	LITTLE-A
1.179249	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
1.300000	GOAL-STATE-LOW-1
10.00000	PRIORITY-LOW-1
2.500000	GOAL-STATE-LOW-2
5.000000	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2
2.000000	GOAL
4.000000	LITTLE-M
0.1000000E+11	BIG-M
1.002502	LITTLE-A
2.309685	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
1.300000	GOAL-STATE-LOW-1
10.00000	PRIORITY-LOW-1
2.000000	GOAL-STATE-LOW-2
5.000000	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2
3.000000	GOAL
4.000000	LITTLE-M
0.1000000E+11	BIG-M
0.8011135	LITTLE-A
2.489846	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
1.300000	GOAL-STATE-LOW-1
9.500000	PRIORITY-LOW-1
2.000000	GOAL-STATE-LOW-2
4.500000	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2
-4.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
0.0000000E+00	BIG-A
0.0000000E+00	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
0.0000000E+00	GOAL-STATE-HIGH-1
0.0000000E+00	PRIORITY-HIGH-1
0.0000000E+00	GOAL-STATE-HIGH-2
0.0000000E+00	PRIORITY-HIGH-2

Table III-2 (continued)

5.000000	GOAL
0.00000000E+00	LITTLE-M
0.00000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2
6.000000	GOAL
3.000000	LITTLE-M
0.10000000E+11	BIG-M
5.000000	LITTLE-A
0.00000000E+00	LITTLE-B
0.00000000E+00	BIG-A
0.00000000E+00	BIG-B
2.500000	GOAL-STATE-LOW-1
5.000000	PRIORITY-LOW-1
0.00000000E+00	GOAL-STATE-LOW-2
5.000000	PRIORITY-LOW-2
0.00000000E+00	GOAL-STATE-HIGH-1
0.00000000E+00	PRIORITY-HIGH-1
0.00000000E+00	GOAL-STATE-HIGH-2
0.00000000E+00	PRIORITY-HIGH-2
7.000000	GOAL
-0.10000000E+11	LITTLE-M
0.00000000E+00	BIG-M
0.00000000E+00	LITTLE-A
0.00000000E+00	LITTLE-B
9.800000	BIG-A
0.14646700E-01	BIG-B
0.00000000E+00	GOAL-STATE-LOW-1
0.00000000E+00	PRIORITY-LOW-1
0.00000000E+00	GOAL-STATE-LOW-2
0.00000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
9.800000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
9.900000	PRIORITY-HIGH-2

Table III-3. TCO Display Data.

0.00000000E+00	UNUSED
0.00000000E+00	UNUSED
0.00000000E+00	UNUSED
0.00000000E+00	UNUSED
0.00000000E+00	UNUSED
0.00000000E+00	UNUSED
43.00000	SCREEN-RANGE
-9999.000	SCREEN-X
-9999.000	SCREEN-Y
0.00000000E+00	UNUSED

2-0 TACTICAL CONTROL ASSISTANT (TCA)

2-1. Human Factors Parameters.

Table III-4 contains the human factors parameters for the Tactical Control Assistant (TCA). The comments in Section 1-1 apply equally to the data for the TCA.

2-2. Goals, Goal Priority Functions, and Objective Parameters.

The TCA has goals four and five from Figure III-1. Goal four causes the TCO to identify unknown tracks, and goal five causes

Table III-4. TCA Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
14.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
3.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
1.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
2.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
-0.6250000E-01	TARGET-HEIGHT
0.6250000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED

Table III-4 (continued)

3.000000	DISPLAY-TARGET-LOCATION
0.00000000E+00	TARGET-LOCATION
480.0000	DISPLAY-RESOLUTION
0.1000000	DISPLAY-BACKGROUND-HEIGHT
0.1000000	DISPLAY-BACKGROUND-WIDTH
0.1000000	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

him to attend to communications from the TCO and the Azimuth-Speed Operator (ASO). The goal priority data for the TCA is shown in Figure III-3. Table III-5 shows how the data appears in the operator state vector.

The TCA has objective function one as does the TCO, and he considers both goals in task sequencing.

2-3. Display Data.

The TCA has the same display data as does the TCO. See Table III-3.

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
4	-∞,0	-	-	-	-	1	2	5	5
5	0,0	-1	1	-2	2	1	1	2	2

OPERATOR: TCA/4

NUMBER OF GOALS CONSIDERED: 3

NAME: Joseph Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION Page 1 of 1

Figure III-3. Goal Priority Function Data for the TCA.

Table III-5. TCA Goal Priority Function Parameters.

4.000000	GOAL
-0.10000000E+11	LITTLE-M
0.00000000E+00	BIG-M
0.00000000E+00	LITTLE-A
0.00000000E+00	LITTLE-B
2.000000	BIG-A
0.5693234	BIG-B
0.00000000E+00	GOAL-STATE-LOW-1
0.00000000E+00	PRIORITY-LOW-1
0.00000000E+00	GOAL-STATE-LOW-2
0.00000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
2.000000	PRIORITY-HIGH-1
5.000000	GOAL-STATE-HIGH-2
5.000000	PRIORITY-HIGH-2
5.000000	GOAL
0.00000000E+00	LITTLE-M
0.00000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2

3-0 AZIMUTH-SPEED OPERATOR (ASO)

3-1. Human Factors Parameters.

Table III-6 contains the human factors parameters for the Azimuth Speed Operator (ASO). The comments in Section 1-1 apply equally to the data for the ASO.

3-2. Goals, Goal Priority Functions, and Objective Parameters.

The ASO has goals five (receive messages) and eight from Figure III-1. Goal eight causes the ASO to pass low altitude targets to the TCO and TCA to be identified and engaged. The goal priority function data for the ASO is shwon in Figure III-4 and Table III-7. The ASO uses objective function one and considers both goals in task sequencing.

Table III-6. ASC Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
3.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
1.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMULTANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
2.000000	GOALS-CONSIDERED
1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.2080000E-01	TARGET-HEIGHT
0.2080000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
240.0000	DISPLAY-RESOLUTION
0.1000000	DISPLAY-BACKGROUND-HEIGHT
0.1000000	DISPLAY-BACKGROUND-WIDTH
0.1000000	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE

Table III-6 (continued)

1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

Table III-7. ASO Goal Priority Function Paremeters.

5.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-P
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2
8.000000	GOAL
-0.1000000E+11	LITTLE-M
0.0000000E+00	BIG-M
0.0000000E+00	LITTLE-A
0.0000000E+00	LITTLE-B
2.000000	BIG-A
0.3219281	BIG-B
0.0000000E+00	GOAL-STATE-LOW-1
0.0000000E+00	PRIORITY-LOW-1
0.0000000E+00	GOAL-STATE-LOW-2
0.0000000E+00	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
2.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.500000	PRIORITY-HIGH-2

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
5	0,0	-1	1	-2	2	1	1	2	2
8	$-\infty$,0	-	-	-	-	1	2	2	2

OPERATOR: ASO/5

NAME: Joseph Polito
DATE: 8/25/83

NUMBER OF GOALS CONSIDERED: 1

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION Page 1 of 1

Figure III-4. Goal Priority Function Data for the ASO.

3-3. Display Data.

The ASO has the same display data as does the TCO. See Table III-3.

4-0 FIRING CONSOLE OPERATOR (FCO)

4-1. Human Factors Parameters.

Table III-8 contains the human factors parameters for both of the Firing Console Operator (FCO). Each FCO has his/her own operator state vector, but, in the default case, they are identical. The comments in Section 1-1 apply equally to the data for the FCO.

4-2. Goals, Goal Priority Functions, and Objective Parameters.

The FCO's have only goal five from Figure III-1. The FCO's respond to communications from the TCO. The goal priority function data is shown in Table III-9 and Figure III-5. The FCO's also use objective function one, and they consider their one goal in task sequencing.

Table III-8. FCO Human Factors Parameters.

37.00000	CORE-TEMPERATURE
1.000000	CIO-VALUE
0.0000000E+00	TIME-ON-TASK
0.0000000E+00	DAYS-OF-DUTY
1.000000	SEARCH-DIMENSIONS
0.0000000E+00	NUMBER-FIRE-UNITS
100.0000	PERCENTAGE-RECOVERY
8.000000	PREVIOUS-WORK
16.00000	PREVIOUS-REST
0.0000000E+00	FLASH-INTENSITY
0.0000000E+00	TARGET-SPEED
57.00000	TARGET-TYPE
0.0000000E+00	TARGET-SIZE
3.000000	TARGET-COLOR
360.0000	SEARCH-AREA
0.0000000E+00	BINOCULAR-USAGE
0.0000000E+00	SLANT-RANGE-TO-TARGET
0.0000000E+00	TARGET-TRAJECTORY
1.000000	TARGET-BAKGRND-COMPLEXITY
0.0000000E+00	NUM-BACKGROUND-CHARACTERS
0.0000000E+00	MESSAGE-BACKLOG
1.000000	SIGNALS-PER-MINUTE
40.00000	HOURS-WORKED-PER-WEEK
0.0000000E+00	DAYS-WITHOUT-SLEEP
0.0000000E+00	DAYS-OF-NIGHT-DUTY
1.000000	SIMUL.ANEOUS-TASKS
1.000000	CONTRAST-RATIO
8.000000	AVE-HOURS-SLEEP
1.000000	OBJECTIVE-FUNCTION
1.000000	GOALS-CONSIDERED

Table III-8 (continued)

1.000000	TARGET-BRIGHTNESS
2.000000	NIGHTS
1.000000	SKY-GROUND-RATIO
0.0000000E+00	AIRCRAFT-ALTITUDE
4.000000	METEOROLOGICAL-RANGE
1.000000	THRESHOLD-CONTRAST
0.6250000E-01	TARGET-HEIGHT
0.6250000E-01	TARGET-WIDTH
0.0000000E+00	TARGET-DEPTH
0.0000000E+00	HORIZONTAL-RANGE
1.000000	NUM-OF-RESOLUTION-ELEM
1.000000	NUM-OF-SUSPECT-AREAS
0.0000000E+00	AIRCRAFT-SPEED
360.0000	FIELD-OF-VIEW
0.0000000E+00	OBSERVER-OFFSET
0.0000000E+00	UNUSED
5.000000	DISPLAY-TARGET-LOCATION
0.0000000E+00	TARGET-LOCATION
480.0000	DISPLAY-RESOLUTION
0.1000000	DISPLAY-BACKGROUND-HEIGHT
0.1000000	DISPLAY-BACKGROUND-WIDTH
0.1000000	DISPLAY-BACKGROUND-DEPTH
1.330000	DISTANCE-TO-DISPLAY
1.000000	DISPLAY-HEIGHT
1.000000	DISPLAY-WIDTH
10.00000	TARGET-NOISE-LEVEL
1.000000	TARGET-DURATION
20.00000	EXPERIENCE
0.1000000	SIGNAL-PROBABILITY
1.000000	REST-PERIODS
0.0000000E+00	TASK-ERROR-FACTOR
0.0000000E+00	TASK-ELEMENT-ERROR-FACTOR
0.0000000E+00	DAYS-SINCE-PRACTICE
1.000000	SENSE-OF-DIRECTION
20.00000	SKIN-TEMPERATURE
240.0000	TIME-IN-TEMPERATURE
20.00000	PREVIOUS-SKIN-TEMPERATURE
-9999.000	X-SCREEN-CENTER
-9999.000	Y-SCREEN-CENTER
-9999.000	SCREEN-RANGE

Table III-9. FCO Goal Priority Function Parameters.

5.000000	GOAL
0.0000000E+00	LITTLE-M
0.0000000E+00	BIG-M
1.000000	LITTLE-A
1.000000	LITTLE-B
1.000000	BIG-A
1.000000	BIG-B
-1.000000	GOAL-STATE-LOW-1
1.000000	PRIORITY-LOW-1
-2.000000	GOAL-STATE-LOW-2
2.000000	PRIORITY-LOW-2
1.000000	GOAL-STATE-HIGH-1
1.000000	PRIORITY-HIGH-1
2.000000	GOAL-STATE-HIGH-2
2.000000	PRIORITY-HIGH-2

4-3. Display Data.

The FCO has the same display data as does the TCO. See Table III-3.

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
5	0,0	-1	1	-2	2	1	1	2	2

OPERATOR: FCO A/6 and FCO B/7

NUMBER OF GOALS CONSIDERED: 1

NAME: Joseph Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

OPERATOR GOAL SPECIFICATION Page 1 of 1

Figure III-5. Goal Priority Function Data for the FCO.

IV. ENVIRONMENTAL DATA

Table IV-1 contains the IHAWK environmental state vector. The various parameters in this vector are explained in Polito (1983a). The user should note that the default sampling option is three (deterministic, moderated), and that each fire section has nine hot and nine cold missiles.

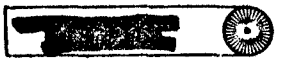
Table IV-1. User Statistics for the IHAWK.

0.0000000E+00	SYSTEM-MODE
0.0000000E+00	OPERATOR-MODE
0.0000000E+00	UNUSED
0.0000000E+00	METHOD-OF-CONTROL
0.0000000E+00	WEAPONS-CONTOL-STATUS
0.0000000E+00	UNUSED
9.000000	INITIAL-AMMUNITION-HOT
9.000000	INITIAL-AMMUNITION-COLD
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	UNUSED
0.0000000E+00	X-POSITION
0.0000000E+00	Y-POSITION
0.0000000E+00	Z-POSITION
3.000000	SAMPLING-OPTION
72.00000	DRY-BULB-TEMPERATURE
50.00000	RELATIVE-HUMIDITY
6.000000	AIR-MOVEMENT-RATE
45.00000	NOISE-LEVEL
50.00000	WORKING-AREA-ILLUMINATION
5.000000	NUMBER-ON-DUTY
0.0000000E+00	VIBRATION
10.00000	AMBIENT-VAPOR-PRESSURE
0.0000000E+00	NOISE-PREDICTABILITY

V. OPERATOR TASKS

IHAWK operator tasks are given below. Each task is numbered and has a title. A brief description of each task is given here and the operator that performs the task is identified. Complete descriptions and MSAINT models of the tasks are contained in Goodin & Walker 1983.

Task Number	Title or Description	Operator
1	ASO Standby Wait For Action - ASO monitors CWAR radar video for new targets.	ASO
2	Detect Target on CWTDC - ASO observes target video on CRT and hears target doppler in headset.	ASO
3	Establish Target Priority - ASO ranks the priority of the current target verses outstanding alerts and other target video.	ASO
4	Preempt Lower Priority Alert - ASO asks permission from TCO to cancel the current TCC alert because he has a higher threat target.	ASO
5	TCC Alert - ASO alerts TCA of the highest threat CWAR target that has not been previously alerted.	ASO
6	Mark Target As Accepted by TCC - After the ASO alerts TCA in Task 5 and the TCA accepts the target the ASO will mark the target on the CRT as accepted.	ASO
7	FCO, Standby Wait For Action - FCO monitors CRT and control panel and awaits assignment or engage commands from the TCO.	FCO
8	Track Target - FCO follows the action of the ADP as it tracks the target and attempts to acquire a IHIPIR lock.	FCO



Task Number	Title or Description	Operator
9	Obtain Lock On Target Manually - FCO performs the actions required to acquire an IHIPIR lock manually. (NOTE: This task is performed for targets that are assigned by the C-73, or are out-of-sector, or are CWAR only targets.)	FCO
10	Put Fire Section Out-of-Action - All hot missiles have been expended so FCO places the fire section out-of-action.	FCO
11	Estimate Raid Size - The FCO monitors doppler and video returns and makes an estimate as to the size of the raid (one, few, or many).	FCO
12	Select Launcher - The FCO manually selects a launcher to fire the next missile (A, B, or C).	FCO
13	Fire Missiles - The FCO fires the number of missiles indicated by the TCO's fire command.	FCO
14	Evaluate Target Intercept - The FCO monitors doppler and video returns and determines the outcome of the engagement.	FCO
15	Put Fire Section Back in Action - FCO returns the fire section to active after missile reloading.	FCO
16	Process Change Targets Command - FCO breaks the IHIPIR lock on the target according to orders from the TCO.	FCO
17	TCA Monitor CRT, Controls, Indicators - The TCA monitors the TCC screen and IFF returns awaiting target classification (friend, foe, neutral) or a new CWAR target from the ASO.	TCA
18	Accept CWTDC Target From ASO - The TCA accepts control of a high threat target picked up by the ASO on the CWTDC.	TCA

Task Number	Title or Description	Operator
19	IFF Challenge - The TCA monitors IFF returns and determines the IHAWK batteries identification of the target.	TCA
20	Mark on Reflection Plotter - The TCA indicates the battery's identification of the target on the reflection plotter.	TCA
21	TCO Monitor CRT, Controls, Indicators - The TCO monitors his CRT to locate new high threat targets. The TCO also monitors communication with the Q-73 and other operators in the BCC.	TCO
22	Detect and Assign IPAR Target - TCO will assign an in-sector IPAR target to a fire section.	TCO
23	Manually Assign Targets - TCO performs a manual assignment of a track to a fire section. (NOTE: This is used to assign Q-73, out-of-sector, and CWAR-only targets to a fire section.)	TCO
24	IHIPIR Tracking - TCO learns information concerning raid size and fire section readiness and determines which fire command to issue.	TCO
25	Send Cannot Comply to Q-73 - TCO has determined that IHAWK cannot act on target as instructed by the Q-73.	TCO
26	Higher Priority Target To Be Assigned - The TCO has located a target with a higher threat than a currently assigned target.	TCO
27	Process A Hostile (Give Fire Command) - The TCO orders the FCO to fire missiles at a target.	TCO
28	Assigned Target Determined Friendly - The TCO orders the FCO to break lock on the currently engaged target because it has been identified as friendly.	TCO

Task Number	Title or Description	Operator
29	Process A Friend - The TCO sends information along the ATDL specifying a target to be a positively identified "true" friend.	TCO
30	Evaluate Whether More Missiles Are To Be Fired - The TCO determines whether or not to continue an unsuccessful engagement.	TCO
31	Give ASO Permission To Cancel Alert - TCO allows the ASO to cancel the current TCC alert so a higher threat target may be alerted.	TCO
32	Receive Message - Task to allow TCO to receive messages from the Q-73 and the other IHAWK operators.	TCO
35	Process "HOLD FIRE" command - TCO performs necessary actions so IHAWK will comply to the HOLD FIRE command issued by the Q-73.	TCO
37	Process "CEASE ENGAGED" Command - TCO performs necessary actions so IHAWK will comply to the CEASE ENGAGED command issued by the Q-73.	TCO
40	Task Sequencing	ALL

VI. OTHER FEATURES

1-0 MISSILE FLIGHT

Missile flight times are calculated using the distance between the IHAWK and the aircraft at time of launch. A constant missile speed of approximately 2000 miles per hour. Missile flight time is calculated in SUBROUTINE FLTY (Goodin & Polito 1983). More complex flight time considerations can be obtained by re-writing this program.

FUNCTION KILLTW (Goodin & Walker 1983) determines whether or not a missile destroys its target. The current implementation assumes that missiles are always effective. This can be changed by re-writing KILLTW.

2-0 RADAR LOCK

Whether the IHIPIR obtains a lock on a particular track is determined by FUNCTION IDLCKW (Goodin & Walker 1983). Currently it is assumed that radar lock is always obtained, but this can be changed by re-writing IDLCKW.

3-0 ESTIMATION OF RAID SIZE

FUNCTION IRDSZW (Goodin & Walker 1983) determines the FCO's perception of the track multiplicity. Currently, it is assumed that the FCO is 100% accurate. This can be changed by re-writing IRDSZW.

4-0 IFF

FUNCTION IFFNW (Goodin & Walker 1983) determines the accuracy of an IFF challenge. Currently, it is assumed that IFF is 100% accurate. This can be changed by re-writing IFFNW.

5-0 TARGET SELECTION

The TCO uses the following procedure in selecting targets. First there must be three hot missiles available and one fire section ready.

Most Threatening Track Is:	Then
Greater than 2 minutes away (approx. 30 Km)	Select assigned Track from Battalion if one exists. Otherwise, do not engage.
Between 1.3 and 2.0 min. away (approx. 20-30 Km).	a) If the track is CWAR, select it over a track assigned by Battalion. b) If the track is IPAR only, select the track assigned by Battalion if one exists. Otherwise, engage the track.
Less than 1.3 min. away (approx. 20 Km)	Engage it instead of a track assigned by Battalion.

VII. STATISTICS

Figure VII-1 shows the statistics maintained by the IHAWK system module. These statistics are in addition to the task fraction statistics collected by MOPADS for all system modules.

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
Time-Persistent		NUMIPAR	Number of IPAR Tracks in radar coverage	
Time-Persistent		NUMCWAR	Number of CWAR Tracks in radar coverage	
Observation		THREAT	Threat of tracks when destroyed.	
Count			Number of missiles fired	One at end of run
"			Num of Tracks destroyed	" " " " "
"			Tracks fired-not destroyed	" " " " "
"			Self started engagements	" " " " "
"			Q-73 initiated engagements	" " " " "
"			Friendly tracks destroyed	" " " " "
"			Unidentified tracks destroyed	" " " " "
NAME: Joseph Polito DATE: 1/10/84			AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS	
SAINT USER STATISTICS				

Figure VII-1. User Statistics for the IHAWK.

VIII. REFERENCES

Goodin, J. R. & Polito, J. Documentation manual for the MOPADS control module (MOPADS/CNTRL) and the MOPADS common system module programs (MOPADS/CSMP) (MOPADS Vol. 5.19). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983

Goodin II, J. R. & Walker, J. L. Documentation manual for the IHAWK system module (MOPADS Vol. 5.16). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft.Bliss, TX, 1983

Laughery, K. R. HOMO establishment of performance criteria for non-decision making tasks (MOPADS Vol. 5.6). Pritsker & Associates, Inc. prepared for the U. S. Army Research Institute Field Unit, Ft.Bliss, TX, 1983

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Polito, J. MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. MOPADS task sequencing structure (MOPADS Vol. 5.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c).

Polito, J. & Laughery, K. R. MOPADS final report (MOPADS Vol. 1.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983

IX. DISTRIBUTION LIST

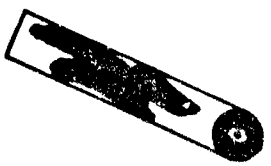
Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301



X. CHANGE NOTICES

APPENDIX C

MOPADS FINAL REPORT:

PERFORMING MOPADS SIMULATION

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

TABLE OF CONTENTS
(continued)

LIST OF FIGURES

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology and Abbreviations

ACN	ADSM Component Number.
ADSM	Air Defense System Module
Reference System Module	A code 11 directory in a MOPADS data base which contains default values for operator and system parameters. It is the "baseline" model for an air defense system.

DB	Data base.
ID	An identifier. ID's of MCPADS data base entries are lists of integer numbers.
ACC	ADSM character code. A single character value uniquely assigned to a reference system module.
DLCC	Data List Character Code. A one, two, or three character mnemonic assigned to certain types of data lists. For example, operator state vectors have a DLCC of "OP."
NECC	Number Equivalent of a Character Code. An integer number that corresponds to a short character string. The capital letters, A-Z, correspond to the integers 1-26 and the digits 0-9 correspond to the numbers 27-36. Thus the string P2B has an NECC of 162902 ("P" corresponds to 16, "2" to 29, and "B" to 02).
DL	Data List. A list of values.
DR	Directory. An index of other directories and/or data lists.

I. INTRODUCTION

1-0 PURPOSE

This document is a user manual for the MOPADS user. It introduces all of the concepts and procedures needed to execute MOPADS simulations. The MOPADS user interacts with the MOPADS software through the MOPADS User Interface. This interface stores and retrieves information from the MOPADS data base. Section II of this report explains the concepts related to the data base that the MOPADS user must know. Section III explains the procedures needed to execute MOPADS. Such things as data cards, job control language, and input/output files are discussed.

Section IV explains the procedures and conventions used in issuing commands to the User Interface. The MOPADS User Interface is an interactive program that combines aspects of menu driven and command driven orientations. Section IV explains the basics of how to interact with this processor.

Sections V and VI explain the contents and purpose of those data base entries that contain reference data on air defense units and air scenarios, respectively. Detailed discussions on how to create and edit this information in contained in Polito (1983a,b).

Section VII explains how to use the User Interface to set up MOPADS scenarios to be simulated. Section VIII contains instructions for specifying particular parameters for use during a simulation. Section IX explains how to examine the statistics produced by a MOPADS simulation.

2-0 INTENDED AUDIENCE

MOPADS users and MOPADS modelers should read this report. The language and level of detail used in the discussion is oriented to the MOPADS user, however. Programming and implementation details are not included here. This information is discussed in detail in MOPADS volumes four and five.

3-0 OTHER READING

The reader is presumed to be familiar with the MOPADS system. This implies that he/she should have read the final report, Polito & Laughery 1983, prior to reading this document. Specific instructions about the details of the implementation of the AN/TSQ-73 and IHAWK MOPADS model are contained in Goodin & Polito (1983a,b). These reports should be read after this document.

II. MOPADS CONCEPTS

1-0 THE STRUCTURE OF THE MOPADS DATA BASE

The MOPADS data base contains virtually all of the information required to set up, run, and review MOPADS simulations. During initial phases of development, a MOPADS modeler uses the User Interface (Polito, 1983a) to create reference air defense system modules (ADSM's) and air scenarios (Polito, 1983b). This information is stored in the MOPADS data base.

With the above building blocks, the MOPADS user creates simulation data sets that contain a specific command and control structure involving (perhaps) multiple copies of the reference system modules and specifying a particular air scenario. When this simulation data set is executed, the resulting statistics are stored in the data base. The MOPADS user can later examine these statistics and select those he/she desires to be printed.

The data base may contain one or more simulation data sets, so multiple analyses can be performed with one data base file. Also, the data base files provide ideal vehicles for archiving MOPADS data, because the data base contains all of the required data to duplicate a simulation.

The MOPADS data base is really a file of information organized in a systematic way. Collections of related data are called "data lists." Data lists (DL's) can be thought of as one or two-dimensional arrays of real, integer, or character data. Collections of data lists are called directories (DR's).

Directories are the highest level of organization in MOPADS data bases, and they are arranged in a hierarchical fashion. Figure II-1 shows the structure of the directories in the MOPADS data base. The "DR" in the boxes indicates a directory as opposed to a data list (data lists are not shown on this figure). Directory labels are shown in capital letters in the larger portions of the boxes. If the label section has an entry in lower case letters, it indicates that an arbitrary number of these directories may exist with user assigned labels. The numbers over the upper right corners are directory codes which are discussed in Section 2-0 below.

The data base is divided into three main types of information: scenarios, reference ADSM's, and simulation data sets. All of the scenario data is "owned" by a single directory labelled "SCENARIOS." The contents and structure of this information is discussed in Section 3-1 below.

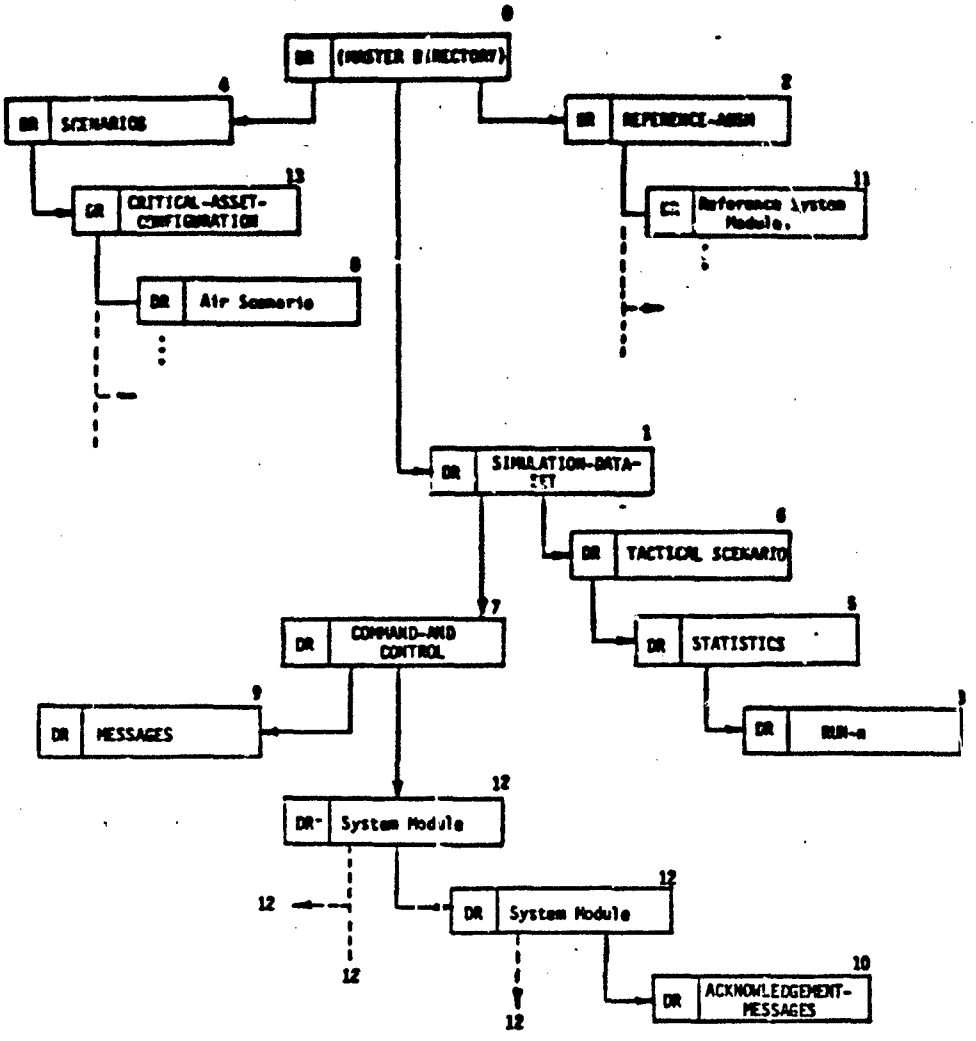


Figure II-1. Structure of MOPADS Data Base.

Reference Air Defense System Modules (See Section 3-2 below) are baseline models of the air defense units. At this writing, models exist for the AN/TSQ-73 and the IHAWK. These reference models are copied into Simulation Data Sets (Section 4-0) to form a command and control structure that is owned by a "COMMAND-AND-CONTROL" directory. Thus, a simulation data set with one Battalion AN/TSQ-73 and two IHAWK fire units could be created using the building blocks in the REFERENCE-ADSM directory.

Specific data concerning a simulation data set and the statistics resulting from a simulation are stored in a "TACTICAL-SCENARIO" directory. These DR's are discussed in Section 5-0, below.

It is essential that the MOPADS user understand the structure of the data base because many of the User Interface commands are concerned with editing the data base contents. Sections V, VI, VII, VIII, IX, and the remainder of this section are concerned with teaching the user what data is in the data base and how to manipulate it.

2-0 DIRECTORIES AND DATA LISTS

Directories and data lists (DR's and DL's) are the two organizational structures that comprise a MOPADS data base. A data list is exactly what its name implies; it is an ordered list of data. DL's are physically where information is stored in the data base. Data lists may be designated as being of type real, integer, or character. Abbreviations for each type are RDL (real), IDL (integer), and CDL (character). RDL's and IDL's may be one or two dimensional. CDL's may have only one dimension. Every data list has a label which gives its name, and every element of a data list may have a label.

Furthermore, each user created directory and data list has an identifier (ID) which is a list of integer numbers. The ID's provide another way to locate and organize data in the data base.

Directories are collections of data lists and other directories. Directories are said to "own" DL's and other directories which belong to it. Thus a hierarchy of directories is formed in which each directory is owned by another directory and which may own still other directories. This hierarchy is a tree since no directory is owned by more than one directory.

Figure II-1 is a schematic of the MOPADS data base which shows the tree structure of the hierarchy. Various data lists may be owned by each directory. For example, the operator state vectors of the operators are represented as data lists that are owned by the system module directories. In Figure II-1, suppose that there is a reference system module for the AN/TSQ-73. It's directory would be owned by

the "REFERENCE-ADSM" directory. The directory for the AN/TSQ-73 will own the operator state vector DL's for each operator (i.e., the Tactical Director (TD) and the Tactical Director Assistant (TDA)). The contents of these DL's will be the baseline or default conditions for these operators.

When a simulation data set is constructed, the AN/TSQ-73 directory (and all of the DL's it owns) will be copied to its appropriate position in the command and control structure (which descends from the "COMMAND-AND-CONTROL" directory). The operator state vector DL's in these copied AN/TSQ-73's will represent the operators during a simulation, and their contents can be individually edited with the user interface to simulate operators with characteristics that differ from those of the default case.

The MOPADS User Interface is used to accomplish all of this; therefore, it is obvious that the MOPADS user must be familiar with the structure of the data base. Table II-1 contains a description of the contents of each directory shown in Figure II-1. Directories are specified by their "directory codes" which are integer numbers between zero and thirteen. Each type of directory has a unique code. The directory codes are shown in Figure II-1 above the upper right corner of each directory rectangle.

3-0 SCENARIOS AND REFERENCE SYSTEM MODULES

3-1. Air Scenarios.

Air Scenarios contain a record of the movements of aircraft. Air Scenarios are created by MOPADS modelers using the procedures in Polito (1983a). Each Air Scenario is defined within a coordinate system and attacks a set of protected or critical assets. The MOPADS modeler specifies the coordinate system and the location of each critical asset. This information is stored in a "CRITICAL-ASSET-CONFIGURATION" directory.

The MOPADS modeler may then develop one or more Air Scenarios within a Critical Asset Configuration. Each Air Scenario is a directory which contains data lists representing aircraft movements. The coordinates of the aircraft movements are specified with respect to the coordinate system of the Critical Asset Configuration. Furthermore, the flight paths of the aircraft in the Air Scenario represent an attack scenario against the critical assets of the Critical Asset Configuration. As will be seen, when performing a MOPADS simulation, the user must specify which Critical-Asset-Configuration directory and which Air Scenario directory to use. Section VI below describes the contents of Air Scenarios and Critical-Asset-Configuration directories in more detail.

Table II-1. MOPADS Data Base Directories.

DIRECTORY CODE	LABEL	DESCRIPTION
0	(MASTER-DIRECTORY)	The master directory is the single entry point to the MOPADS DB.
1	SIMULATION-DATA-SET	This directory owns all of the information requird to perform a MOPADS simulation. It owns a command and control configuration, specification of the scenario, and output statistics. More than one SIMULATION-DATA-SET directory may exist.
2	REFERENCE-ADSM	This directory owns all of the reference system modules in the DB. Only one code 2 directory exists.
3	RUN-n*	This directory contains statistics for one run of the simulation (run n).
4	SCENARIOS	This directory owns all of the scenarios in the DB. It is analogous to a code 2 directory. Only one will exist.
5	STATISTICS	This directory owns output statistics from a MOPADS simulation resulting from execution of a code 3 directory specification. More than one of these directories may exist.
6	TACTICAL-SCENARIO	This directory owns all information necessary to run a MOPADS simulation except the command and control configuration. A simulation data set may contain more than one code 6 directory.
7	COMMAND-AND-CONTROL	This directory owns an entire command and control configuration. All of the system modules in the configuation descend from this directory. Each code 1 directory contains only one code 7 directory.

Table II-1. (continued)

DIRECTORY CODE	LABEL	DESCRIPTION
8	air scenario*	These directories contain records of aircraft movements in air battles. As many of these as are needed may exist in each code 13 directory.
9	MESSAGES	This directory contains messages sent among system modules during simulations. Each code 7 directory will own one of these.
10	ACKNOWLEDGEMENT-MESSAGES	Each code 12 directory owns one of these. The directory contains messages that must be acknowledged.
11	reference system module*	Each reference system module is a type 11 directory.
12	system module*	When reference system modules are copied to their positions in a command and control configuration, they become code 12 directories. Their labels are internally assigned.
13	CRITICAL-ASSET-CONFIGURATION	This directory contains the physical layout of the battle area. It defines the coordinate syste and location of critical assets.

* Labels or portions of labels in lower case letters imply that the label may vary from instance to instance of these directories.

3-2. Reference Air Defense System Modules.

MOPADS modelers develop MSAINT models of air defense systems. The collection of such models represents a "menu" of models that the MOPADS user can choose from in creating MOPADS models of air defense configurations. The models that make up this menu are called Reference System Modules, because they contain the baseline or default values of operator and system parameters.

All Reference System Modules belong to the "REFERENCE-ADSM" directory in the data base (see Figure II-1). The Reference System Modules contain data lists with the following information:

1. Operator State Vectors
2. Environmental State Vectors
3. Operator Labels
4. MSAINT Task Node Data
5. Resource Data

All of this information can be edited with the user interface when the Reference System Modules are copied into a Command-and-Control structure to become System Modules for simulation. Section V describes the contents of System Modules in more detail.

4-0 SIMULATION DATA SETS

When a MOPADS user begins to create a MOPADS model of an air defense system, he/she first needs to ensure that a MOPADS modeler has created a Critical Asset Configuration and at least one Air Scenario for the area to be simulated. Also, Reference System Modules of each type of air defense unit to be included must be present in the data base. The user then creates a Simulation Data Set in the data base (using commands in the User Interface). The Simulation-Data-Set directory will contain a Command-and-Control directory. The user will copy the Reference System Modules into correct positions descending from the Command-and-Control directory. An example of an air defense configuration (showing only the System Module directories) is shown in Figure II-2. This figure shows a configuration with a Group AN/TSQ-73 that has two subordinate Battalions. One of the Battalions has two IHAWK batteries and the other has only one IHAWK.

The analogy of the data base structure and the command-and-control structure is obvious: a unit is subordinate to another unit if its directory is owned by the directory of the superior unit. Thus by using the appropriate User Interface commands, the user can construct an air defense configuration that represents the command hierarchy under study.

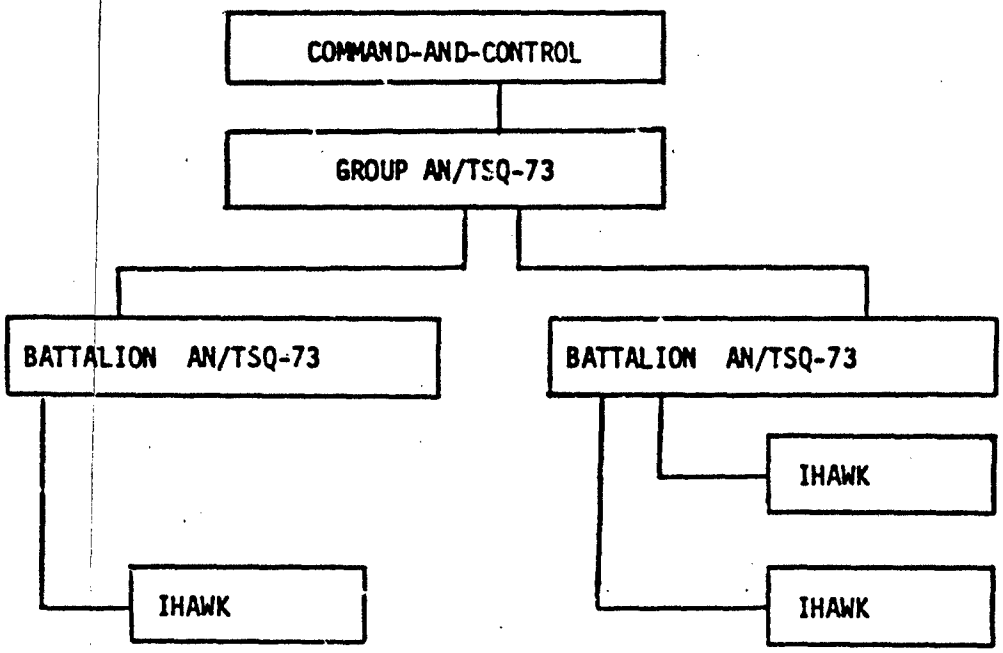


Figure II-2. Example Air Defense Configuration.

In addition to the air defense configuration, the Simulation-Data-Set also contains a Tactical-Scenario directory that owns information concerning the simulation to be performed. For example, the Tactical Scenario contains a specification of which Critical Asset Configuration and which Air Scenarios are to be used. Also, output statistics are saved in the Tactical Scenarios during a simulation. Sections VII, VIII, and IX contain detailed discussions of these aspects of the data base.

III. GETTING STARTED WITH MOPADS

1-0 DATA CARDS

The MOPADS software reads one or more data cards whenever it is executed. This is true even when a User Interface session is run. This section describes these initial data requirements.

MOPADS has two execution modes: "User-Interface" and "Run." The User-Interface mode initiates an interactive user interface session. This allows the user to create, edit, and examine entries in the data base. MOPADS must be executed in the Run mode, however, to perform a simulation. The usual sequence of events for the user is as follows:

1. Execute one or more User Interface sessions to create a complete Simulation Data Set that contains all data necessary to perform a simulation.

2. Execute MOPADS in the Run mode to perform the simulation. In this mode, MOPADS can be executed as a batch job because no interactive input or output is required.

3. Execute a User Interface session to examine the outputs from the simulation performed above.

The data cards discussed below must be placed on the "MOPADS Batch Input File" (Polito, 1983c). The data cards are read with FORTRAN 77 list directed input, which means that the column spacing in which the data appears is not significant. Table II-2 shows the contents of the data cards.

For a User Interface session, only one data card is required. It must have the characters 'USER-INTERFACE' typed on it. The apostrophes must be included. This is true of all of the data cards shown in Table III-1.

To perform a MOPADS simulation, the first data card must be 'RUN' (data card 1 in Table II-2). This must be followed by additional data cards to specify the data for the simulation. Data card 2 specifies the name of the data base file. The user assigns integer numbers to Simulation Data Sets and to Tactical Scenarios when they are created with the User Interface. Data cards 3 and 4 require the user to specify which of these are to be used in the simulation since more than one of each may exist.

Table III-1. MOPADS Data Cards.

USER INTERFACE MODE: *		
CARD	LABEL FIELD	DATA FIELD
1	'USER-INTERFACE'	(not used)

RUN MODE: *		
CARD	LABEL FIELD	DATA FIELD
1	'RUN'	(not used)
2	'DATA BASE FILE NAME'	'file name'
3	'SIMULATION DATA SET'	integer number
4	'TACTICAL SCENARIO'	integer number
5	'N', 'P', or 'C'	'file name'
Repeat card 5 as needed		

* Apostrophes must be typed wherever shown in the data cards.

Finally, when a Reference System Module is created by a MOPADS modeler, it is necessary to create an MSAINT network data file that contains MSAINT data cards that define the network model. One such file is created for each Reference System Module, and the MOPADS software must have access to all of them. Therefore, a type 5 data card must be included for each Reference System Module known to MOPADS. At this writing, there are four Reference System Modules incorporated into MOPADS. A type 5 data card must be included for each of them, even if not all types are represented in the Simulation Data Set to be simulated, e.g., a data card must be included for the Group AN/TSQ-73 even if the Simulation Data Set contains only Battalion AN/TSQ-73's and IHAWK's.

The label field for type 5 data cards is the echo option for that system module type. The meanings of the options are as follows:

'P' The MSAINT network data cards are echoed to the MOPADS line printer output file as they are read. Also, input error messages are printed. 'P' stands for "Partial."

'C' In addition to 'P' above, a formatted summary of the MSAINT network is printed. 'C' stands for "Complete."

'N' No input data is echoed. Input errors are printed, however. 'N' stands for "None."

Examples of MOPADS data cards are given below. Example file names are hypothetical, but they are valid file names for Digital Equipment Corporation's VAX computers with the VMS operating system on which MOPADS was developed. Other computer systems may use different file name conventions. Consider the following:

```
'RUN'
'DATA BASE FILE NAME', '[PROJ.MOPADS]MOPADS.DBF'
'SIMULATION DATA SET', 3
'TACTICAL SCENARIO', 2
'N', 'CNTRL.NET'
'N', 'GROUP73.NET'
'N', 'BATT73.NET'
'P', 'IHAWK.NET'
```

A MOPADS execution in the Run mode is to be performed. The MOPADS data base is on file [PROJ.MOPADS]MOPADS.DBF. Simulation Data Set number 3 is to be used with its Tactical Scenario numbered 2.

MOPADS modelers must assign a number to Reference System Modules, and the type 5 data cards must be ordered to agree with this numbering. For the present implementation, the order shown above is mandatory. The first system module type is the Control System module which is a default MSAINT model that is

automatically included in every simulation. Its function is to control the simulated flights of aircraft in the Air Scenario. The Control System Module is not present in the "REFERENCE-ADSM" directory and need not be explicitly copied into the Simulation Data Set by the user. The network data files for the Group AN/TSQ-73, Battalion AN/TSQ-73, and the IHAWK must follow in that order.

In the example above, the Control System Module network is to be found on file CNTROL.NET, and no echo of the data cards are to be echoed. Similarly, the Group and Battalion AN/TSQ-73 and the IHAWK network data are on files GROUP73.NET, BATT73.NET, and IHAWK.NET, respectively. The data cards for the IHAWK are to be echoed. Finally, the commas that separate the label and data fields in the example can be replaced by one or more blanks.

2-0 OTHER FILES

In the User Interface mode, the MOPADS software will access several other files. It must have access to any MOPADS data base files which will be accessed during the session. The software opens these files with FORTRAN 77 OPEN statements and internally assigns a unit number to them. Input from and output to the User's terminal is performed with the MOPADS interactive input and interactive output files. These files must be associated with the User's terminal by job control language, Polito (1983c). Furthermore, some options in the User Interface will write information to the MOPADS line printer output file. This file must also be assigned with job control language.

In the Run mode, the interactive input and output files are not used. However, several others are. MOPADS reads the system module network files and writes a composit network file on the MSAINT Network Input File. The DEC VAX name assigned to this file is MOPADS.NET and it is assigned by the software. Similarly, a scratch file is created as a temporary file which will be deleted when execution terminates. Finally, if a trace is produced, it will be written to a file named TRACE.DAT.

IV. USING THE USER INTERFACE

1-0 WORKING AND REFERENCE DATA BASES

With the User Interface it is possible to have two MOPADS data base files open simultaneously. They are referred to as the "working" and "reference" data bases. The working data base file is the one which is affected by editing commands issued by the user. The reference data base is used only to store and retrieve back-up or reference material.

For example, when a MOPADS modeler creates a new reference air defense system module, the data base directory for this model will contain parameters specific to the operators and environment for the air defense system represented by the model. The MOPADS modeler should create this model in a data base file that contains no other information. Then this data base file should be archived to insure that it is not inadvertently lost or damaged.

When a MOPADS user desires to use this new system module, he should obtain a copy of the data base information in his own personal MOPADS data base file. This can be accomplished with the User Interface by designating the archive file as the reference data base and the User's data base as the working data base. Commands are available to copy the system model information from the REFERENCE-ADSM directory of the reference data base to the REFERENCE-ADSM directory of the working data base. Procedures for performing these operations are described in Polito (1983a,b)

Normally, only a working data base is opened by the user, since saving and retrieving data base information is needed only to preserve or access reference type data.

2-0 CURRENT DIRECTORIES AND DATA LISTS

Look now at Figure II-1. The User Interface is like a telescope which can be focused on one and only one directory in the data base at a time. This is called the "current directory," In other words, the user can examine the contents of only one directory at a time. Naturally, commands are available which allow the current directory to be changed.

Similarly, the user may examine and/or change the contents of individual data lists which belong to the directories. To do this, the data list must be selected as current also. Therefore, when using the User Interface, there is usually a current directory, and there may be a current data list.

3-0 ID'S, DIRECTORY POSITION, AND LABELS

The User Interface is capable of printing a Directory Report which lists the characteristics and contents of the current directory. Figure IV-1 shows an example Directory Report. The information at the top of the report pertains to the current directory.

The User Message, "MOPADS CURRENT DIRECTORY," simply indicates that this Directory Report is being produced in response to the DIRECTORY command which will be discussed shortly. Next, the type of data base is printed. This field will be either "Working" or "Reference." The name of the data base file is printed next, so the user can always determine which files are open as the reference and working data bases.

Every directory and data list has a label consisting of up to 40 characters (25 for data list) and an ID which is a list of integer numbers. The label and ID of the current directory are printed next. ID's are assigned in a systematic way which is explained in Tables IV-1 and IV-2.

```
                    D I R E C T O R Y   R E P O R T

    USER MESSAGE:          MOPADS CURRENT DIRECTORY
    DATA BASE:             WORKING
    DATA BASE FILE:        TEST.DBF
    DIRECTORY LABEL:       SIMULATION-DATA-SET
    DIRECTORY ID:                1     1
    RANKING CODE(OWNED DIRECTORIES):   INCREASING ON ID(   1)
    RANKING CODE(OWNED DATA LISTS):    INCREASING ON ID(   1)

OWNED DIRECTORIES:
------------------
DIRECTORY POSITION:          1
LABEL:                       TACTICAL-SCENARIO
ID: (  1-  2)=           6       1

DIRECTORY POSITION:          2
LABEL:                       COMMAND-AND-CONTROL
ID: (  1-  2)=           7       1


OWNED DATA LISTS :
------------------
DIRECTORY POSITION:          3
LABEL:                       COPY-COUNTER
ID: (  1-  2)=           0       0
```

Figure IV-1. Example Directory Report.

The ranking codes for owned directories and data lists are given next. For the example in Figure IV-1, owned directories are arranged in increasing order of the first word of their ID's. Ranking can be done decreasing instead of increasing, and if the ranking is done on "ID(0)", then no ranking is performed. The ranking codes for the directories are normally not of interest to the MOPADS user.

The contents of the directory are listed next. The label and ID of each owned DR and DL is given. For example, the directory "TACTICAL-SCENARIO" has a two word ID which is ID(1)=6 and ID(2)=1. In addition, a number called the "DIRECTORY POSITION" is given for each DL and DR. The directory position is a unique integer number assigned to each owned DR and DL. The directory position of "TACTICAL SCENARIO" is "1."

The directory position has physical meaning, but its use to the MOPADS user is in specifying a new current directory. For example, if we desired to make the "TACTICAL SCENARIO" the new current directory, we could do it in one of three ways:

1. Give the label (with no typing errors),
2. Give the entire ID, or
3. Give the dirctory position.

Usually, it is easiest to specify the directory position.

Tables IV-1 and IV-2 contain the conventions used in assigning ID's to DR's and DL's. For example, the first element of the ID of a DR is the directory code (see Table II-1). Through frequent use and reference to Tables IV-1 and IV-2, the user will become familiar with the meanings of the ID's.

4-0 SUBPROCESSES

Figure IV-2 is a schematic of the organization of the User Interface. The User Interface contains five subprocesses:

1. CREATE/EDIT SIMULATION DATA SET
2. SET UP SIMULATION RUN DATA
3. EXAMINE STATISTICS
4. CREATE/EDIT SCENARIO DATA
5. CREATE/EDIT REFERENCE SYSTEM MODULE

Each subprocess has its own set of commands. Subprocesses 1, 2, and 3 are described in Sections VII, VIII, and IX of this report.

Subprocess 4 and 5 are used mainly by MOPADS modelers. They are described briefly in Sections VI and V and more completely in Polito (1983a,b).

Finally, a set of basic data base commands is available in all subprocesses. These commands are discussed in this section.

Table IV-1. Directory ID's

DIRECTORY CODE	ID ELEMENT	VALUE
1	1 2	1 sequentially numbered
2	1 2	2 1
3	1 2	3 simulation run number
4	1 2	4 0
5	1 2	5 1
6	1 2	6 sequentially numbered
7	1 2	7 1
8	1 2	8 user assigned air scenario number
9	1 2 3 4	9 0 0 0
10	1 2 3 4	10 0 0 ADSM component number (ACN)
11	1 2 3 4	11 unique value assigned to the ADSM character code (ACC) 0 basic ACN (e.g., 1000, 2000, etc.)

Table IV-1 (continued)

12	1 2 3 4	12 same as code 11 above sequential assigned ACN
13	1 2	13 user assigned

See MOPADS Volume 5.17 (Polito 1983d) for complete descriptions of the directory ID's and contents.

Table IV-2. ID's of Selected Data Lists.

DATA LIST	OWNER DIRECTORY CODES	ID ELEMENT	VALUE
Run Data	6	1 2	1 0
TD-TASK-DATA	11,12	1 2 3 4	unique number assigned to the ADSM character code (ACC) 2004 1 0
OP-OPERATOR-STATE-VECTOR	11,12	1 2 3 4	same as TD-TASK-DATA 1516 operator type 0
EN-ENVIRONMENTAL-STATE-VECTOR	11,12	1 2 3 4	same as TD-TASK-DATA 514 1 0
FIELD-OF-VIEW FV	12	1 2 3 4	same as TD-TASK-DATA 622 1 ADSM Component Number

Complete descriptions of all Data Lists are contained in MOPADS Volume 5.17 (Polito 1983d).

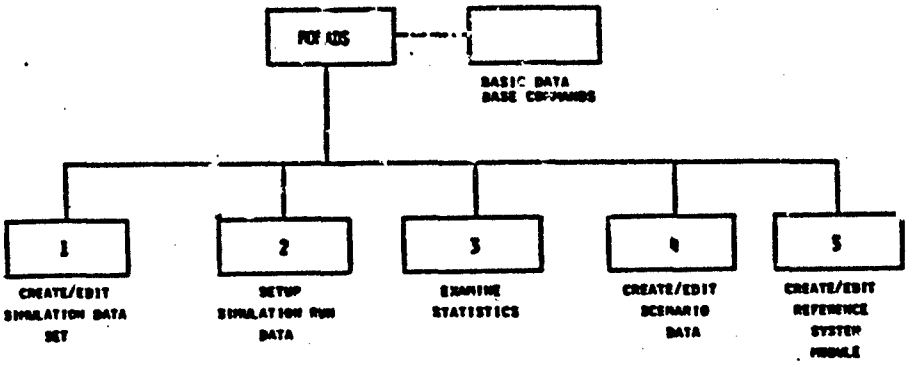


Figure IV-2. Organization of the User Interface.

5-0 USER INTERFACE COMMAND SYNTAX

The User Interface is primarily a command driven processor that waits for the user to issue instructions. It does, however, have aspects of menu driven systems in that some commands result in menus being presented to the user. Also, the command processor (FFSP described in Goodin & Polito(1983c))permits menu-like presentations of commands.

The regular mode for entering commands is shown below.

command,prompt1=response1/prompt2=response2/...

The commands and prompts are keywords recognized by MOPADS. The responses are particular values for the prompts. For example, consider this.

OPEN,FILE=MOPADS.DBF/STATUS=OLD

OPEN is the command. FILE and STATUS are prompts recognized by MOPADS, and MOPADS.DBF and OLD are values for the prompts.

Certain prompts for a command may have default values that will be used if the prompt is not entered. In the example above, another prompt, DB, specifies which data base is to be associated with the file. Its default is WORKING, so by not entering it on the command line, WORKING is automatically selected. If the default value is not desired, then the prompt must be explicitly entered on the command line.

If the user fails to enter responses to all required prompts, MOPADS will interactively prompt for them. For example,

```
OPEN,STATUS=OLD

FILE[NO DEFAULT] = MOPADS.DBF
```

After processing the OPEN command, MOPADS found that the required prompt, FILE, was not entered. It printed "FILE[NO DEFAULT]=" to prompt the user for a response. If the last non-blank character on a command line is a dash (-), MOPADS will interactively prompt for all unentered prompts, even those with defaults. For example,

```
OPEN,STATUS=OLD-

DB[WORKING] = REFERENCE

FILE[NO DEFAULT] = MOPADS.DBF
```

The dash caused "DB[WORKING]=" to be printed. The value between the brackets is the default for the prompt. The default can be selected by typing "DEF" as the response. DEF can also be entered on the command line; e.g.

```
OPEN,DB=DEF/STATUS=OLD/FILE=MOPADS.DBF
```

The above demonstrates that the prompt-response groups can be entered in any order.

Also, a command can be cancelled at any time by typing "CANC" as a response or a prompt. For example,

```
OPEN,CANC

OPEN,FILE=CANC
```

Note that DEF and CANC are essentially reserved words. The user interface treats commas, blanks, and equal signs as interchangeable separators. Also, multiple separators are treated as a single separator. This means that the commas in the previous examples could be replaced by any combination of one or more blanks and commas. The same is true of the equal signs, but their use is recommended to make the command lines easy to read. The slashes are required separators between prompt-response groups, but they can be preceded or followed by blanks or commas.

A response may include separators (i.e., commas, blanks, equal signs, and slashes) if it is enclosed in quote marks ("). For example, on some computers file names contain embedded blanks, e.g.,

```
OPEN  FILE="MOPADS DBF"
```

Without the quote marks above, MOPADS will consider MOPADS DBF as two responses when only one is desired. (NOTE: A single prompt may have more than one response if the programmer specified it that way. In such a case, each response would be separated by a blank or comma. In the case above, however, where a single response is required, the quote marks must be used to embed the blank in the response.)

Any response may be enclosed in quotes, althought there is no advantage in doing so unless a separator is to be embedded. Blank responses can be entered with " " where at least one blank appears between the quotes.

A generalization of entering only some of the prompts is to enter only the command name:

```
OPEN

DB[WORKING]=DEF

FILE[NO DEFAULT]=MOPADS.DBF

STATUS[OLD]=DEF
```

The User Interface will prompt for all responses. This method can be selected if the user does not remember the prompts.

For commands which the user issues frequently, a concise mode can be selected by preceding the command with "C-". In this case, the prompt= part of the syntax may be omitted. For example,

Responses must be entered in the same order as they are prompted in the command-name-only form. No response may be skipped, except that if all remaining responses have defaults and the defaults are desired, then the command line may be terminated (e.g., the STATUS response was omitted above since OLD was desired). The dash works in the concise mode in the same way as in other modes.

The following rules will formalize the previous discussion of how syntax is processed by FFSP.

The **command-name-only** form of a command may be used at any time by typing only the command name.

Blank responses and responses containing separators may be entered by enclosing them in quotes. To enter a blank response type " " (including the quotes). At least one blank must be entered between the quote marks.

A command may be **cancelled** at any time by typing CANC for any prompt or response. You can not abbreviate CANC.

The user may elect to use the **default** value(s) by typing DEF for any response in a response list up to one field past the last response in the list.

Slashes (/) must be used to separate one prompt-response group from another. **Blanks or commas** may be used to separate all other fields. The equal sign should be used to separate prompts from their responses; however, it is not required.

Command and prompt names may be **abbreviated** to any non-ambiguous string of characters. For example, if there are two commands, DESIGN and DESCRIBE, they can be abbreviated DESI and DESC respectively. The commands may be abbreviated in longer forms. For example, the user may enter DESC, DESCR, DESCRI, DESCRIB, or DESCRIBE for the command DESCRIBE.

If a command line in regular or concise mode is ended with **more than one dash**, the last dash will signify to the system to prompt the user for all the unentered responses. Other dashes will then be considered as part of a response.

Any multiple combination of commas and blanks is treated as a single separator. For example,

NAME = BILL WOLF and NAME = BILL , WOLF

are equivalent (here the response is a list of two character strings).

If the user enteres an incorrect response or misuses the syntax, FFSP will explain the error and prompt interactively for all remaining responses.

Concise mode is signified by preceding the command name with "C-" (without the quotes).

6-0 BASIC DATA BASE COMMANDS

The following commands are available in all five subprocesses.

1. CLOSE
2. CURRENT
3. DEPOSIT
4. DIRECTORY
5. EXAMINE
6. HELP
7. MENU
8. OPEN
9. PLINK
10. QUIT
11. SELECT
12. TERMINATE

The prompt and responses for these commands are shown in Table IV-3. A brief description of each command follows.

6-1. CLOSE. The CLOSE command (Table IV-3a) will close either the working or reference data base. It can be used to switch to a new data base file.

6-2. CURRENT. The CURRENT command (Table IV-3b) will display label, ID or both of the current directory and/or data list on either data base.

6-3. DEPOSIT. DEPOSIT (Table IV-3c) is a low level editing command that allows any element of the current data list to be changed. DEPOSIT interactively requests element numbers and new values.

6-4. DIRECTORY. (Table IV-3d) shows the contents (all owned directories and/or data lists) of the current directory on either data base. It shows the labels, ID's, and directory positions of the contents. This information is useful for the SELECT command.

Table IV-3. Basic Data Base Command Prompts.

COMMAND	PROMPTS	RESPONSES	DEFAULTS
(a) CLOSE	DB	WORKING REFERENCE	WORKING
	STATUS	KEEP DELETE	KEEP
(b) CURRENT	DB	WORKING REFERENCE	WORKING
	DISPLAY	LABEL ID ALL	LABEL
	TYPE	DIRECTORY DATA-LIST ALL	DIRECTORY
(c) DEPOSIT	DB	WORKING REFERENCE	WORKING
(d) DIRECTORY	DB	WORKING REFERENCE	WORKING
	TYPE	DIRECTORY DATA-LIST ALL	DIRECTORY
(e) EXAMINE	DB	WORKING REFERENCE	WORKING
	HEADER	NO YES	NO
	DEVICE	TERMINAL PRINTER	TERMINAL

Table IV-3 (continued)

(f) HELP	COMMAND	a command name	--
(g) MENU	(none)	--	--
(h) OPEN	DB	WORKING REFERENCE	WORKING
	FILE	file name	--
	STATUS	OLD NEW	OLD
	MAXRECORD	non-negative integer	0
(i) PLINK	DB	WORKING REFERENCE	WORKING
(j) QUIT	(none)	--	--
(k) SELECT	DB	WORKING REFERENCE	WORKING
	TYPE	DIRECTORY DATA-LIST	DIRECTORY
	POSITION	integer	0
	LABEL	string	blank
	ID	list of integers	0
(l) TERMINATE	(none)	--	--

6-5. EXAMINE. EXAMINE (Table IV-3e) will display selected contents of the current data list to the terminal or to the MOPADS line printer output file. If the latter is selected, the data list label and other information will also be printed.

6-6. HELP. HELP (Table IV-3f) will print the prompts and options for the prompts for the specified command.

6-7. MENU. MENU has no prompts. It will print all commands available in the current subprocess.

6-8. OPEN. OPEN (Table IV-3h) will open a data base file as either the working or reference DB. OPEN will not automatically close the current DB. CLOSE must be used explicitly before OPEN to switch DB files. MAXRECORD is the maximum number of records allowed in a data base file. It may not be needed for your computer. Zero implies no limit on the file size. The (MASTER-DIRECTORY) is current after the OPEN command.

6-9. PLINK. PLINK (Table IV-3i) will change the current directory to the owner of the directory which was current when PLINK was issued.

6-10. QUIT. QUIT has no prompts. It causes the current subprocess to be exited.

6-11. SELECT. SELECT (Table IV-3k) changes the current directory or data list to one that is owned by the directory that is current when SELECT is issued. The desired DR or DL is selected by specifying one (and only one) of the following: 1-its directory position, 2-its label, or 3-its ID. This information is obtained with the DIRECTORY command.

6-12. TERMINATE. TERMINATE has no prompts. It will close all open data bases and terminate execution. This is the normal way to end a User Interface session.

6-13. An Annotated Example. Figure IV-3 is an example using all the Basic Data Base Commands. The annotations are explained below.

(1) - The CLOSE commands disassociates the current data base file from the MOPADS software. After close, MOPADS will know no working DB. If STATUS is specified as DELETE, the data base file will be destroyed permanently.

(2) - OPEN associates the data base file VOL46.DBF to the software as the working DB. STATUS=OLD implies that the DBF was previously created as a MOPADS data base file. If STATUS=NEW is specified, MOPADS will create a new MOPADS DB on VOL46.DBF, and any information previously contained on VOL46.DBF will be lost.

(3) - The MENU command is always available to list the commands currently available.

(4) - HELP prints information about a particular command.

(5) - The user has asked for information on the prompts DISPLAY and TYPE. For DISPLAY, MOPADS shows that it expects a response of one character string that must be one of the enumerated values LABEL (display the label), ID (display the ID), or ALL (display both label and ID). The default is LABEL. HELP is terminated by entering "*."

(6) - When OPEN is issued, the (MASTER-DIRECTORY) becomes current. The CURRENT command will not display a directory which has no owner, which of course, the MASTER-DIRECTORY does not.

(7) - The DIRECTORY command will always work, however, even on the (MASTER-DIRECTORY). The contents of the (MASTER-DIRECTORY) are shown.

(8) - The SELECT command makes the REFERENCE-ADSM directory current by specifying its directory position in the (MASTER-DIRECTORY).

(9) - The DIRECTORY command confirms that REFERENCE-ADSM is current and shows the owned reference ADSM's (in this case only one, E-EXAMPLE).

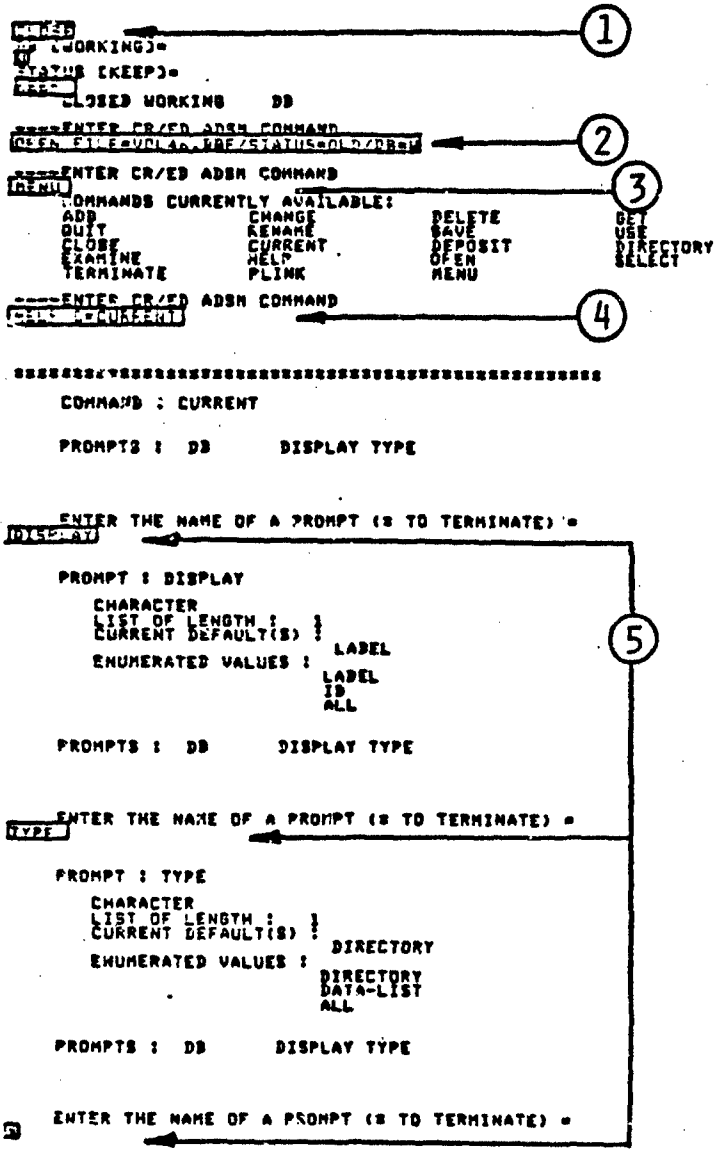


Figure IV-3. Example Using Basic Commands.

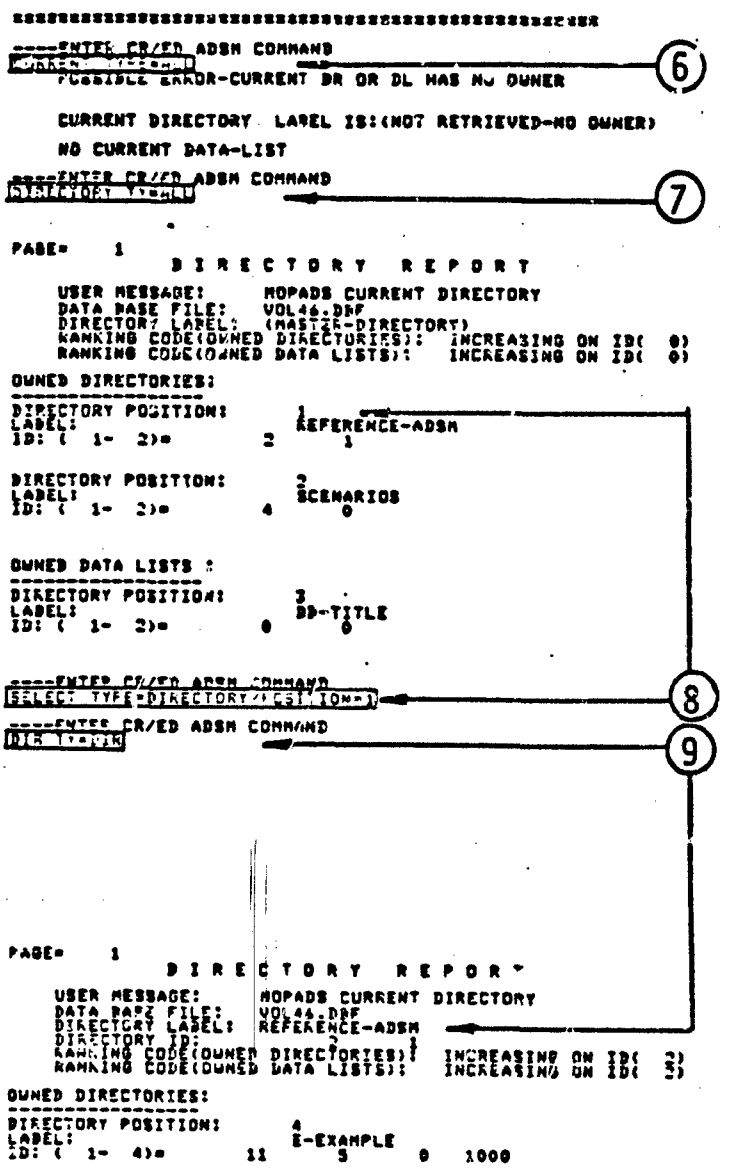


Figure IV-3. (continued)

```
---ENTER CR/ED ADSM COMMAND
REPORT TYPE=DIRECTORY
---ENTER CR/ED ADSM COMMAND
DIR TYPE=DIRECTORY
```

(10)

(11)

```
PAGE=    1
                    D I R E C T O R Y   R E P O R T

    USER MESSAGE:      MOPADS CURRENT DIRECTORY
    DATA BASE FILE:    VOL44.DBF
    DIRECTORY LABEL:   E-EXAMPLE
    DIRECTORY ID:          11          3        0    1000
    RANKING CODE(OWNED DIRECTORIES):   INCREASING ON ID(  0)
    RANKING CODE(OWNED DATA LISTS):    INCREASING ON ID(  2)

OWNED DATA LISTS :
------------------
DIRECTORY POSITION:      1
LABEL:                   R-ADB-RESOURCES
ID: (  1-  3)=       S     18        1

DIRECTORY POSITION:      2
LABEL:                   EM-ENVIRONMENT-STATE-VEC
ID: (  1-  3)=       S    514        1

DIRECTORY POSITION:      3
LABEL:                   OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=       S   1516        1

DIRECTORY POSITION:      4
LABEL:                   OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=       S   1516        2

DIRECTORY POSITION:      5
LABEL:                   DT-OPERATOR-TYPE
ID: (  1-  3)=       S   1520        1

DIRECTORY POSITION:      6
LABEL:                   RL-RESOURCE-LABELS
ID: (  1-  3)=       S   1012        1

DIRECTORY POSITION:      7
LABEL:                   YD-TASK-DATA
ID: (  1-  3)=       S   2004        1

---ENTER CR/ED ADSM COMMAND
```

(12)

Figure IV-3. (continued).

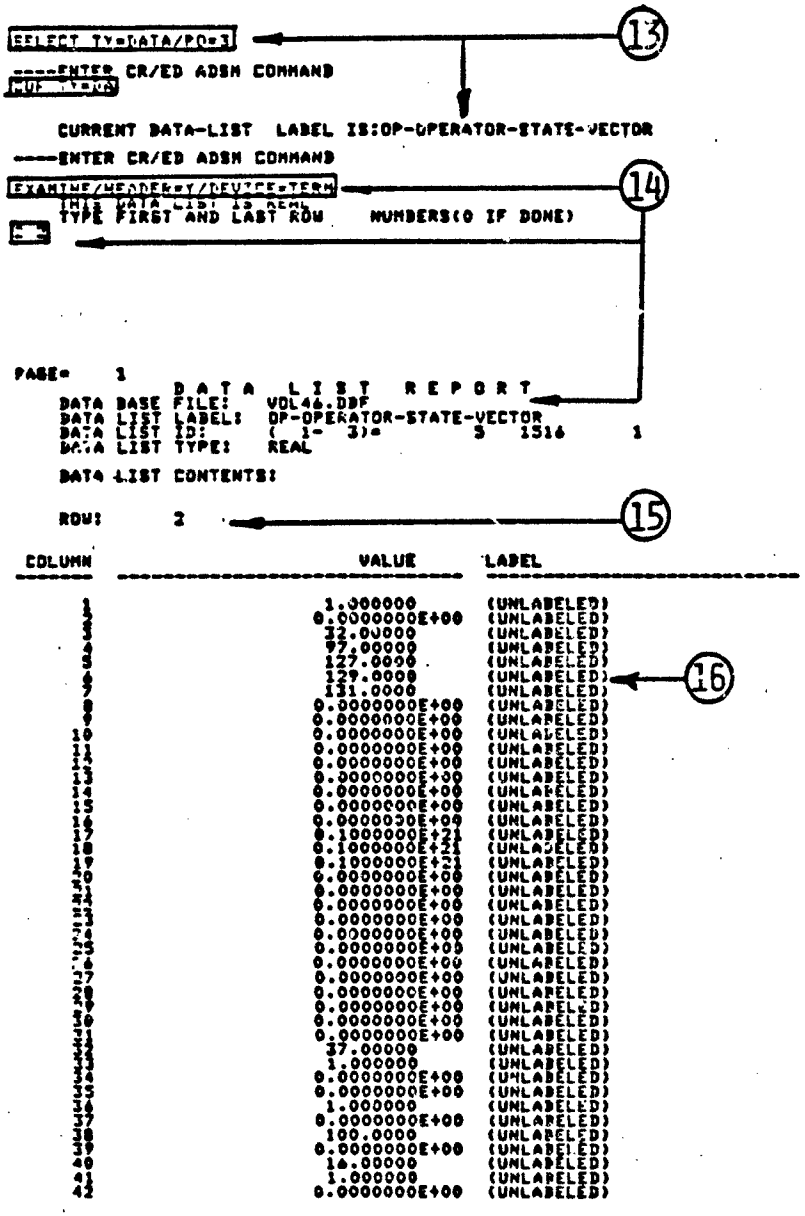


Figure IV-3. (continued)

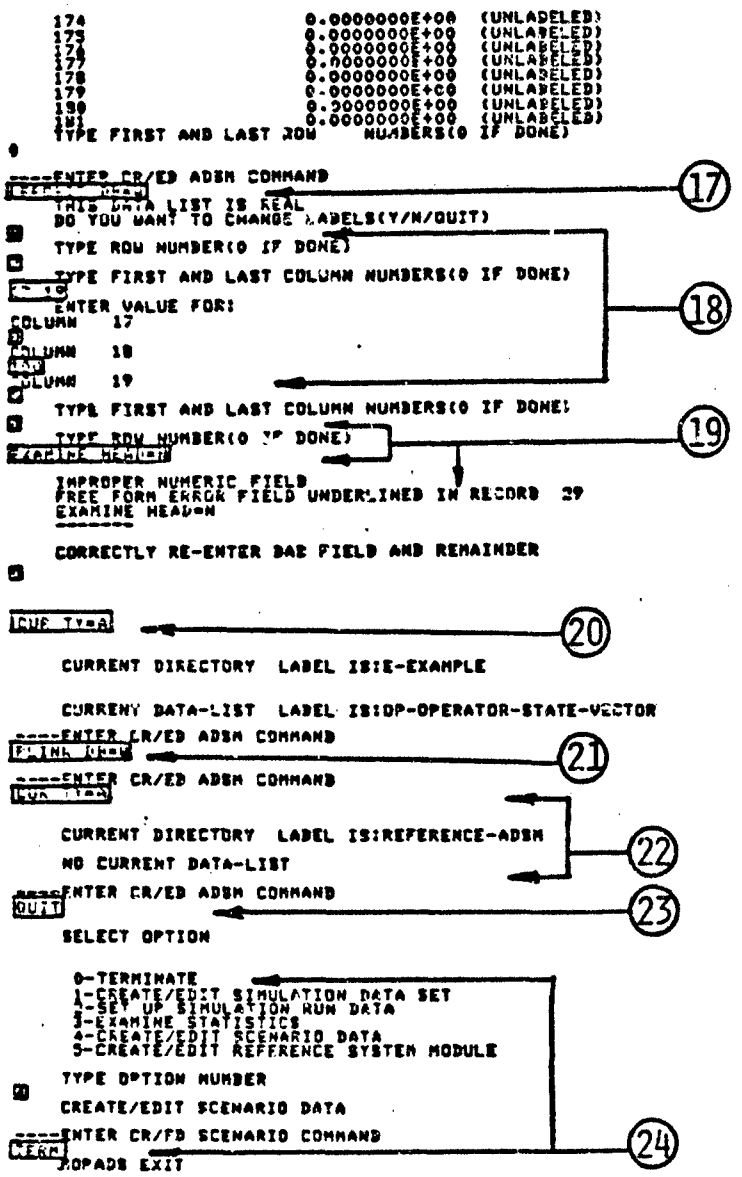


Figure IV-3. (continued)

(10) - Select E-EXAMPLE to be current.

(11) - Look at the data list directory of E-EXAMPLE.

(12) - This operator state vector will be edited.

(13) - The SELECT command makes the above operator state vector the current data list. This is confirmed with the following CURRENT command.

(14) - EXAMINE specifies that the data list is to be displayed at the terminal. The user wants row 2. HEADER=Y causes the "DATA LIST REPORT" and the next 6 lines to be printed.

(15) & (16) - This display demonstrates that the basic data base commands are low level commands; they know nothing about the structure or contents of MOPADS data bases. The operator state vectors have two rows. The second row contains the reference values for each column. Prior to a simulation, the second row is copied to the first row, which is dynamically accessed during the simulation. In this way, the reference or starting values are not lost. MOPADS stores column labels only for the first row, however, since it would double the storage requirement to duplicate the labels for each row. The "CREATE/EDIT REFERENCE SYSTEM MODULE" subprocess knows all of this, so when a listing is obtained with the CHANGE command in that subprocess, the labels appear with the reference values.

The EXAMINE command, however, simply operates on a data list without knowledge of the meaning of its contents. Here, all of the elements of row 2 appear to be (UNLABELED). Also, EXAMINE prints the entire row. Only a portion of the listing is shown here for brevity.

(17) - Use the DEPOSIT command to change elements. It too is a low level command, so (row, column) addressing must be used.

(18) - Elements 17, 18, and 19 of row 2 are changed (these are operator trace parameters).

(19) - This demonstrates MOPADS response to input errors. The DEPOSIT command had not yet terminated when the user entered the next EXAMINE. MOPADS is expecting a numeric value for a row number. The free-format input processor examines every input string for

validity before processing it. Since "EXAMINE" is not a valid number, it prints the error message and requests correct input. Without this feature, the program would terminate abnormally from simple typing errors and perhaps damage the data base. MOPADS will protect the user from most such errors.

(20) - CURRENT shows that E-EXAMPLE is the current directory.

(21) - PLINK makes the current directory equal to the owner of the directory that was current when PLINK was issued.

(22) - The CURRENT command confirms that the REFERENCE-ADSM directory is now current. Note that when the current directory is changed, the current data list becomes undefined.

(23) - QUIT exits the current subprocess and brings up the subprocess option menu again.

(24) - The TERM command terminates execution. Selecting the zero option on the subprocess selection menu accomplishes the same thing.

V. CONTENTS OF THE REFERENCE AIR DEFENSE SYSTEM MODULES

1-0 REFERENCE ADSM DATA LISTS

Reference Air Defense System Modules (ADSM) are code 11 directories. They contain the base line values of the various parameters which define the modules. The User Interface contains editors for changing the information in these directories. The same editors are available for modifying working ADSM's (code 12 directories). Code 12 directories are simply copies of the code 11 directories.

The Reference ADSM directories contain the following data lists:

1.	Operator State Vectors	These DL's contain data that is unique to a single operator.
2.	Environmental State Vector	This DL contains data which affects all operators in a system module.
3.	Task Data	This DL may contain information for each MSAINT task node in the MSAINT model of the system.
4.	System Resources	This DL contains information about any hardware resources which may have been defined by the MOPADS modeler.
5.	Operator Type and Resource Labels	These DL's are used internally by the software and are of no concern to the user.

The contents of these DL's is discussed briefly below. The values for all of the parameters discussed in the following sections can be edited by the user with the User Interface. Values for the parameters are not shown in this document since they will vary from system module to system module. For example, the values for the IHAWK are given in Goodin & Polito (1983b).

2-0 OPERATOR STATE VECTOR

Each operator in an ADSM is represented by a separate operator state vector. These vectors have several parts: human factors

parameters, goal parameters, objective function parameters, and display data.

The human factors parameters are shown in Table V-1. Discussion of these parameters is contained in Laughery 1983 and Polito (1983d). Goal parameters are shown in Table V-2.

Each goal that the operator has will be represented by fifteen parameters as shown in Table V-2. The parameters have the meanings shown in Figure V-1 where

$\underline{GS}_i$ is GOAL-STATE-LOW-i

$\underline{GP}_i$ is PRIORITY-LOW-i

$\overline{GS}_i$ is GOAL-STATE-HIGH-i

$\overline{GP}_i$ is PRIORITY-HIGH-i

The values for m, M, a, A, b, and B are computed once when the goal parameters are initially entered by the MOPADS modeler. If the MOPADS user desires to change these values, they must be computed manually using the procedures in Polito(1983e) and then entered with the User Interface editor.

Objective Function data is shown in Table V-3. These parameters specify whether the operator minimizes the average goal priority (signified by a value of 1.0) or maximizes the average goal priority reduction per unit time (signified by a value of 2.0) The number of goals that the operator considers is also specified.

Finally, the labels and values for Display Parameters are stored in the Operator State Vectors. Up to 10 display parameters can be defined. The definitions of display parameters is entirely up to the MOPADS modeler except that parameters 7, 8, and 9 must be as shown in Table V-4. A typical set of display parameters is shown in the table. Values must be specified for whatever parameters are defined for the ADSM.

3-0 ENVIRONMENTAL STATE VECTORS

Each Code 11 directory has an Environmental State Vector. The parameters in this data list apply to everyone in the system module. There are two types of data: system parameters and human factors parameters.

Table V-1. Human Factors Parameters in the Operator State Vectors.

1	CORE-TEMPERATURE
2	CIO-VALUE
3	TIME-ON-TASK
4	DAYS-OF-DUTY
5	SEARCH-DIMENSIONS
6	NUMBER-FIRE-UNITS
7	PERCENTAGE-RECOVERY
8	PREVIOUS-WORK
9	PREVIOUS-REST
10	FLASH-INTENSITY
11	TARGET-SPEED
12	TARGET-TYPE
13	TARGET-SIZE
14	TARGET-COLOR
15	SEARCH-AREA
16	BINOCULAR-USAGE
17	SLANT-RANGE-TO-TARGET
18	TARGET-TRAJECTORY
19	TARGET-BAKGRND-COMPLEXITY
20	NUM-BACKGROUND-CHARACTERS
21	MESSAGE-BACKLOG
22	SIGNALS-PER-MINUTE
23	HOURS-WORKED-PER-WEEK
24	DAYS-WITHOUT-SLEEP
25	DAYS-OF-NIGHT-DUTY
26	SIMULTANEOUS-TASKS
27	CONTRAST-RATIO
28	AVE-HOURS-SLEEP
29	OBJECTIVE-FUNCTION
30	GOALS-CONSIDERED
31	TARGET-BRIGHTNESS
32	NIGHTS
33	SKY-GROUND-RATIO
34	AIRCRAFT-ALTITUDE
35	METEOROLOGICAL-RANGE
36	THRESHOLD-CONTRAST
37	TARGET-HEIGHT
38	TARGET-WIDTH
39	TARGET-DEPTH
40	HORIZONTAL-RANGE
41	NUM-OF-RESOLUTION-ELEM
42	NUM-OF-SUSPECT-AREAS
43	AIRCRAFT-SPEED
44	FIELD-OF-VIEW
45	OBSERVER-OFFSET
46	UNUSED
47	DISPLAY-TARGET-LOCATION
48	TARGET-LOCATION
49	DISPLAY-RESOLUTION
50	DISPLAY-BACKGROUND-HEIGHT
51	DISPLAY-BACKGROUND-WIDTH
52	DISPLAY-BACKGROUND-DEPTH
53	DISTANCE-TO-DISPLAY
54	DISPLAY-HEIGHT
55	DISPLAY-WIDTH
56	TARGET-NOISE-LEVEL
57	TARGET-DURATION
58	EXPERIENCE
59	SIGNAL-PROBABILITY
60	REST-PERIODS
61	TASK-ERROR-FACTOR
62	TASK-ELEMENT-ERROR-FACTOR
63	DAYS-SINCE-PRACTICE

Table V-1. (continued)

```
64   SENSE-OF-DIRECTION
65   SKIN-TEMPERATURE
66   TIME-IN-TEMPERATURE
67   PREVIOUS-SKIN-TEMPERATURE
68   X-SCREEN-CENTER
69   Y-SCREEN-CENTER
70   SCREEN-RANGE

TYPE ELEMENT TO CHANGE( 0 FOR NONE)
```

Table V-2. Goal Parameters in the Operator State Vectors.

```
 1   GOAL
 2   LITTLE-M
 3   BIG-M
 4   LITTLE-A
 5   LITTLE-B
 6   BIG-A
 7   BIG-B
 8   GOAL-STATE-LOW-1
 9   PRIORITY-LOW-1
10   GOAL-STATE-LOW-2
11   PRIORITY-LOW-2
12   GOAL-STATE-HIGH-1
13   PRIORITY-HIGH-1
14   GOAL-STATE-HIGH-2
15   PRIORITY-HIGH-2
```

Table V-3. Objective Function Parameters in the Operator State Vectors.

```
1   OBJECTIVE-FUNCTION
2   NUMBER-OF-GOALS
```

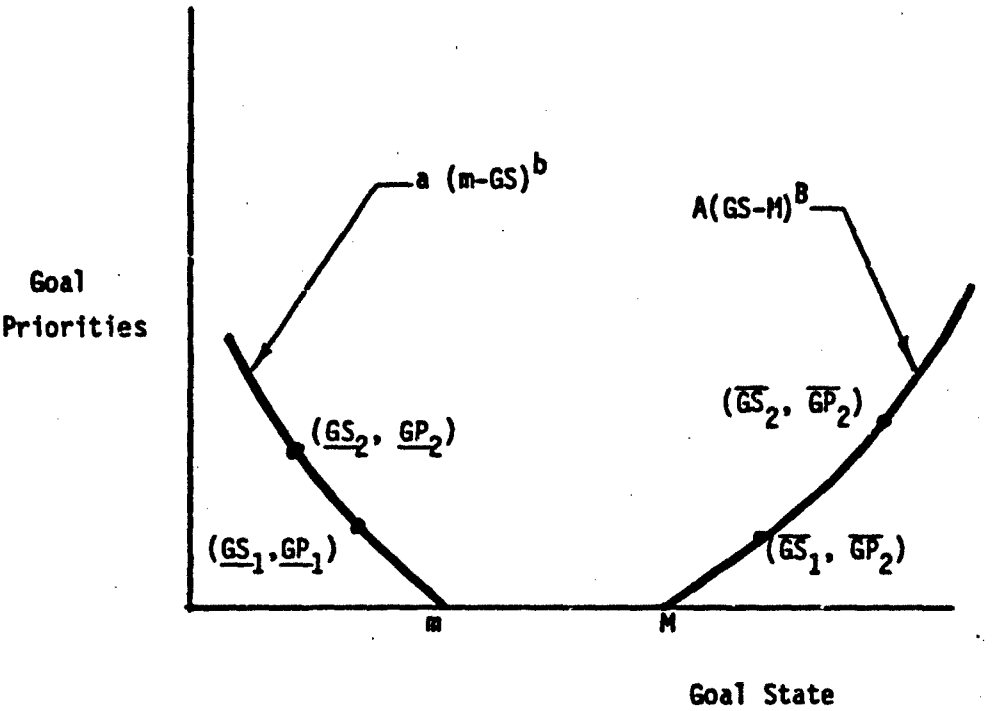


Figure V-1. Goal Parameters.

Table V-4. Typical Display Parameters in the Operator State Vector.

1	HOOKED-ITEM
2	VECTORS
3	SCREEN-ALPHA
4	ENGAGEMENT-MARKERS
5	PAIRING-LINES
6	MAP
7	SCREEN-RANGE
8	SCREEN-X
9	SCREEN-Y
10	UNUSED

Table V-5 shows the system parameters. At the present time only the following parameters are used:

7 and 8	Initial numbers of hot and cold missiles at each fire section for the IHAWK
18,19, and 20	The location of the unit.
21	The sampling option for task times. Acceptable values are: 1 - unmoderated,deterministic task time (i.e.,use the mean times specified for the task time distribution) 2 - unmoderated, stochastic (i.e., random sampled times without human factors moderators) 3 - moderated, deterministic (i.e., moderate the mean with human factors influences) 4 - moderated, stochastic (i.e., moderate the mean by human factors influences and then select a random sample from this new distribution)

Table V-6 shows the human factors parameters in the Environmental State Vector. As with the Operator State Vectors, a more detailed discussion of these parameters is contained in Laughery 1983 and Polito (1983d).

4-0 TASK DATA

Certain parameters apply to the task rather than to the individual performing the task. Examples are the skills required to perform the task and the energy output required. Associated with each task is the vector of task data shown in Table V-7.

Distribution types accepted by MOPADS are as follows:

1. Constant
2. Normal
3. Uniform
4. Erlang-1 (exponential)
5. Lognormal
6. (not used)
7. Beta
8. Gamma

Table V-5. System Parameters in the Environmental State Vectors.

1	SYSTEM-MODE
2	OPERATOR-MODE
3	UNUSED
4	METHOD-OF-CONTROL
5	WEAPONS-CONTOL-STATUS
6	UNUSED
7	INITIAL-AMMUNITION-HOT
8	INITIAL-AMMUNITION-COLD
9	UNUSED
10	UNUSED
11	UNUSED
12	UNUSED
13	UNUSED
14	UNUSED
15	UNUSED
16	UNUSED
17	UNUSED
18	X-POSITION
19	Y-POSITION
20	Z-POSITION
21	SAMPLING-OPTION

Table V-6. Human Factors Parameters in the Environmental State Vectors.

TYPE	ELEMENT TO CHANGE(0 FOR NONE)
1	DRY-BULB-TEMPERATURE
2	RELATIVE-HUMIDITY
3	AIR-MOVEMENT-RATE
4	NOISE-LEVEL
5	WORKING-AREA-ILLUMINATION
6	NUMBER-ON-DUTY
7	VIBRATION
8	AMBIENT-VAPOR-PRESSURE
9	NOISE-PREDICTABILITY

Table V-7. Task Specific Data.

DISTRIBUTION-TYPE
MEAN
STANDARD-DEVIATION
KILOCALORIES/MIN
NUMBER-OF-BRANCHES-OUT
STIMULUS-MODE-1
STIMULUS-MODE-2
RESPONSE-MODE
OBSRVR-TARGET-POSITION
CONTROL-DISTANCE
CONTROL-WIDTH
NUMBER-OF-DISPLAYS
NUMBER-OF-ALTERNATIVES
NUM-STM-ITEMS
SKILL-INDEX
SKILL-WEIGHT

One or more skills may be required to perform the task. If so, the SKILL-INDEX and SKILL-WEIGHT parameters are repeated for each skill. The weights are specified as fractions or percents. Further discussions are found in Polito & Laughery 1983, Laughery 1983, and Laughery & Gawron 1983.

The other human factor parameters are discussed in Laughery 1983 & Polito (1983d). They are used in MOPADS in the same way as the human factors parameters of the Operator State Vectors.

Finally, the Task Data may contain system resource needs. For example, certain tasks may require system resources that are subject to breakdown or which may become unavailable for some other reason. System resources are specified in a similar fashion to skills. In other words, a resource index and a parameter are specified. The current MOPADS models do not represent such system resources, so they are not included in Table V-7. However, the data base and User Interface support this feature, so a MOPADS modeler can build or modify MOPADS models to include system resources. (NOTE: The system resource feature described here is entirely independent of the resource modeling capability built in to the MSAINT language.)

The System Resources data list contained in Reference ADSM directories contains data on system resources if any are defined. It is used in conjunction with the resources specifications in the Task Data. This data list is not used by the present MOPADS modeler. See Polito(1983d) for details.

VI. CONTENTS OF THE SCENARIOS DIRECTORY

1-0 DISCUSSION OF SCENARIOS

Figure VI-1 shows an expanded view of the scenarios directory and its descendants. A scenario consists of two parts: a specification of the coordinate system and the locations of critical assets or protected sites that the air defense system seeks to protect, and a specification of the aircraft which will fly through the area during the simulation.

The CRITICAL-ASSET-CONFIGURATION (Code 13) directories contain a complete scenario specification as discussed above. More than one CRITICAL-ASSET-CONFIGURATION can be contained in a MOPADS data base as indicated in Figure IV-1.

Each code 13 directory contains a COORDINATE-AND-ASSET-DATA data list that contains the definition of the coordinate system and the locations of all critical assets. It will also contain one or more air scenario (code 8) directories that specify aircraft movements. Since a set of critical assets can be attacked in a multitude of ways, MOPADS allows the modeler to develop more than one air scenario and store them in the data base. The particular air scenario to be simulated is selected when a simulation is set up. Each air scenario (code 8) contains separate data lists for hostile, friendly, and "other" aircraft tracks. "Other" tracks are those that cannot be classified as friendly.

2-0 CRITICAL ASSET CONFIGURATION

The ID's of Code 13 directories consist of two parts:

ID(1) = 13

ID(2) = user specified integer

The CRITICAL-ASSET-CONFIGURATIONS are selected by specifying ID(2) when setting up a simulation.

The contents of the COORDINATE-AND-ASSET-DATA data list are shown in Table VI-1. The reference point (elements 1, 2, and 3) specify the origin of the coordinate system. A rectangular coordinate system for a flat earth is used by MOPADS. The reference point allows a particular point on the earth to be specified by latitude and longitude to aid in building a scenario for a particular area, and taking the coordinates of critical assets, tracks, radars, and air defense units from a terrain map of the region.

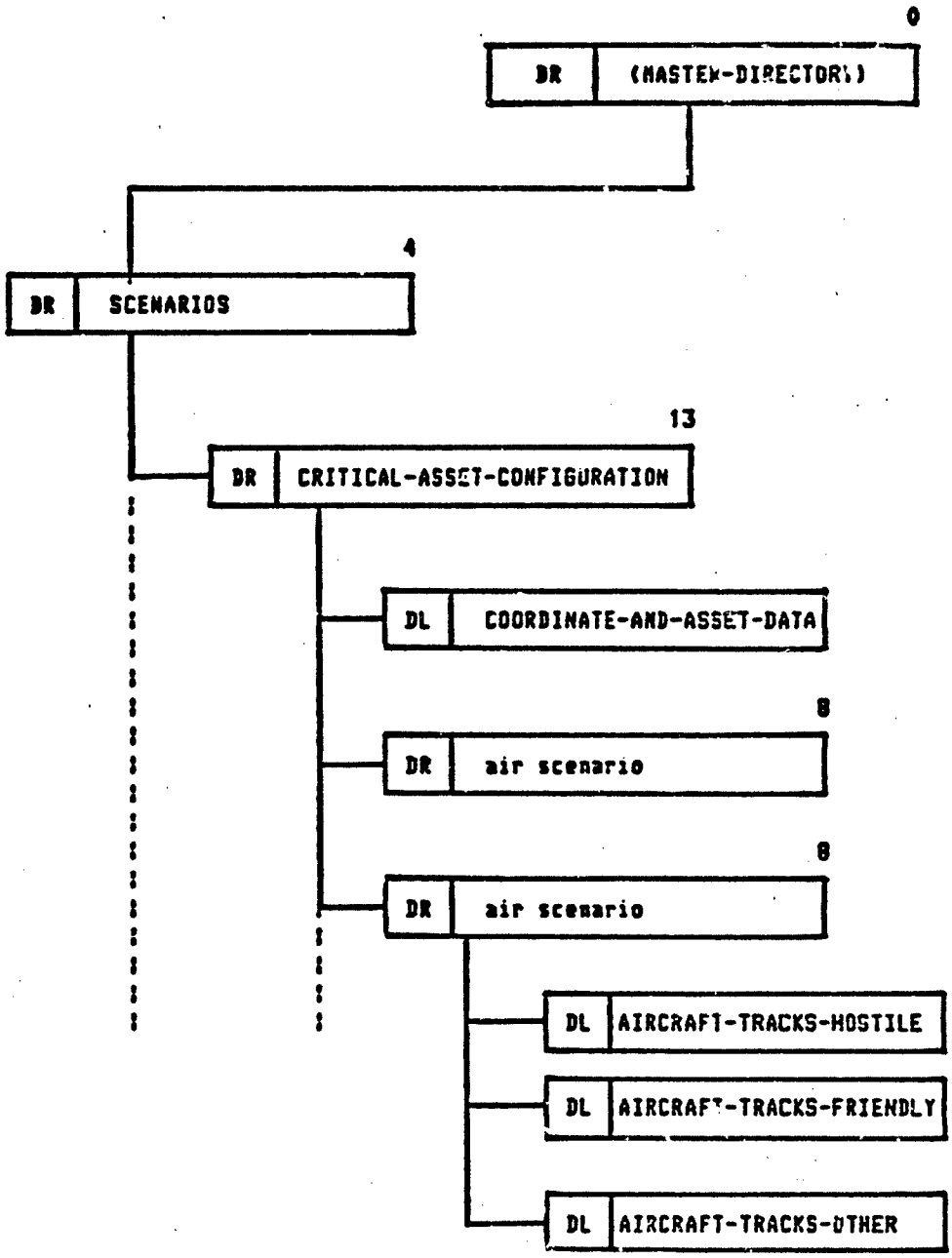


Figure VI-1. Scenarios Portion of the MOPADS Data Base.

Table VI-1. Contents of the COORDINATE-AND-ASSET-DATA Data List.

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1	REFERENCE-LATITUDE	decimal degrees (negative for south latitude)
2	REFERENCE-LONGITUDE	decimal degrees (negative for east longitude)
3	REFERENCE-ALTITUDE	feet above MSL
4	NUMBER-OF-CRITICAL-ASSETS	The number of critical asset data sets which follow.
5	X-COORDINATE	(n.mi.) for first critical asset (+ is east)
6	Y-COORDINATE	(n.mi.) for first critical asset (+ is north)
7	Z-COORDINATE	(ft.) for first critical asset (+ is up)
8	SITE-TYPE	A code value (currently always zero)
9	asset label (5 char.max)	Zero
10	(Not used)	Zero
		repeat (5) - (10) for each critical asset

If the scenario is not related to a particular region, the coordinates of the reference point may be set to zero.

All other coordinates are specified in relation to the reference point. For example, critical assets are located by specifying x, y, and z coordinates using the reference point as the origin. The x and y coordinates are in units of nautical miles with positive x being east and positive y being north. The z coordinate is in units of feet above or below the reference point. Positive z is up.

Each critical asset is specified by its coordinates and a label. Growth potential is allowed in the data base for differentiating among critical assets by type and possibly by priority.

3-0 AIR SCENARIOS

Air scenario (code 8) directories contain three data lists; one for each type of track. The data lists are identically structured in that they contain the same types of information in the same format. Two types of records are stored as shown in Table VI-2.

Aircraft tracks are represented as piecewise linear segments. Flight direction and velocity do not change along a segment but may vary from segment to segment. There is no limit to the number of segments, so curivilinear motions can be approximated if needed. Furthermore, the aircraft motions are completely specified by the information in these data lists. Therefore, the current models do not represent evasive actions or contingency redirection by hostile aircraft.

The first type of record is a track initiation record. This record specifies the initial coordinates of the track, the time in minutes from the beginning of the simulation when the track appears, the aircraft type, multiplicity, and a Track ID number. Aircraft type codes are specified in Polito (1983d), but they are currently not used by the models.

Each track segment is specified by a track segment record with the following information: coordinates of the end point, and speed in knots. Additionally, for hostile tracks, this record contains an indication of whether or not the end point of the segment is a target (critical asset or air defense unit), the probability that the target is destroyed, and whether or not the aircraft is jamming along the segment. Currently, the jamming information is not used. All of this data is specified by the MOPADS modeler at the time the air scenario is created. See Polito (1983b) for more details.

Finally, air scenario directories have labels assigned by the MOPADS modeler at the time they are created, and there ID's consist of two parts:

ID(1) = 8

ID(2) = user specified integer

The particular air scenario to be simulated is selected by the MOPADS user by specifying ID(2) when setting up a simulation.

Table VI-2. Track Data.

TRACK INITIATION RECORD	
	Track ID number
	Initiation time (minutes from start of simulation)
	Multiplicity
	Aircraft type
	x-position (n.mi.)
	y-position (n.mi.)
	z-position (ft.)
TRACK SEGMENT RECORD	
	x-position of end of segment (n.mi.)
	y-position of end of segment (n.mi.)
	z-position of end of segment (n.mi.)
	speed (knots)
	end point a target (0-no, 1-yes)
	jamming (0-no, 1-yes)
	probability of destroying target

VII. CREATING AND EDITING SIMULATION DATA SETS

1-0 CREATE/EDIT SIMULATION DATA SET COMMANDS

The following commands are available in the CREATE/EDIT SIMULATION DATA SET subprocess.

1. ADD
2. CHANGE
3. DELETE
4. INSERT
5. REMOVE

The prompts and responses for these commands are shown in Table VII-1.

The commands in this section are used to create SIMULATION-DATA-SET (code 1) directories (see Figure II-1), COMMAND-AND-CONTROL (code 7) directories and all directories that descend from the code 7 directories. Each of the commands is discussed in detail below, and this section ends with an annotated example.

Table VII-1 CREATE/EDIT SIMULATION DATA SET Command Prompts.

COMMAND	PROMPTS	RESPONSES	DEFAULTS
(a) ADD	ID NUMBER	positive integer	(none)
(b) CHANGE	DATA-LIST	TD R OP EN FV	(none)
(c) DELETE	ID NUMBER	positive integer	(none)
(d) INSERT	ADSM	Label of a Reference Air Defense System Module	(none)
(e) REMOVE	ADSM	Label of a Working Air Defense System Module	(none)

1-1. ADD. The ADD command (Table VII-1a) will create a new SIMULATION-DATA-SET directory. In fact it also creates the COMMAND-AND-CONTROL (code 7) and MESSAGES (code 9) directories. Thus, when this command is executed, an "empty" simulation data set is created. It is empty in that it contains no copies of reference ADSM's and no scenario data.

The ADD command can be issued as often as needed with the restriction only that the user assign a different ID NUMBER to each simulation data set. In this way, models of more than one command and control configurations can be represented in the same data base.

1-2. CHANGE. The CHANGE command (Table VII-1b) is used to edit data lists in working ADSM (code 12) directories. For example, if the command and control structure in Figure II-2 were created, each of the three IHAWK models could be individually edited with the CHANGE command. The CHANGE command operates on the current directory. Therefore, to edit the structure in Figure II-2, it would be necessary to use the SELECT and PLINK commands to make the desired IHAWK directory current.

The data which can be edited is specified by the prompt DATA-LIST as follows:

TD	-	Task Data
R	-	System Resources
OP	-	Operator State Vectors
EN	-	Environmental State Vector
FV	-	Field of View

All of these were discussed in Section V except Field of View. An air defense unit's field of view is determined by the areas in which its radars can detect aircraft as explained in Section 1-3 below. The CHANGE command can be used to edit the field of view data.

1-3. Characteristics of Viewers. Each air defense unit may "own" one or more "viewers." Viewers are usually radars (and in the current implementation, they are always radars), but in the case of REDEYE or VULCAN, for example, the viewer might be a human observer.

Each viewer has the following characteristics.

1. maximum range
2. minimum and maximum altitude
3. probability of detection
4. barriers to view
5. a sector of interest

The characteristics above serve to restrict a viewer's ability to detect aircraft. The maximum range and altitude restrictions are self explanatory. The probability of detection is the probability that the viewer will detect an aircraft that is otherwise in its field of view. MOPADS assumes that once an aircraft is detected it remains detected so long as it is in the viewer's field of view.

The MOPADS user may specify barriers-to-view that block out part of a viewer's ability to detect aircraft. The barriers approximate terrain and other limitations that preclude a radar or observer from seeing everything within range. Figure VII-1 shows how barriers may be specified. Two types of barriers may be specified: line barriers (type 1) and wedges (type 2).

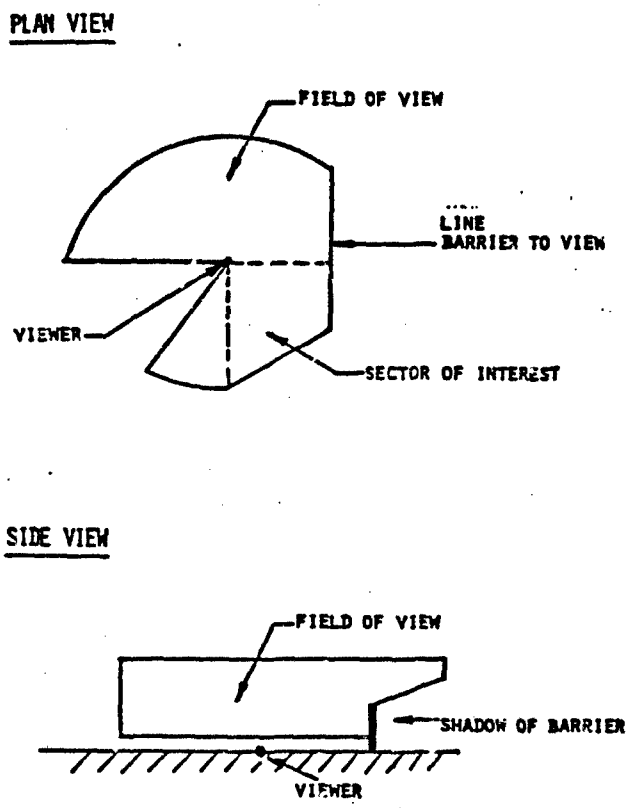


Figure VII-1. Viewers and Barriers-to-View.

In the plan view of Figure VII-1, two line barriers are shown to the east and southeast of the viewer. Everything farther away from the viewer than the line barrier is hidden from view if it is in the shadow of the barrier. The side view in Figure VII-1 demonstrates this. The area to the east of the barrier and below the shadow is hidden from the viewer (Note: the side view does not show a complete detail of the viewer in the plan view).

Line barriers may be positioned anywhere in the maximum range of the viewer and the end points may have different altitudes. MOPADS assumes a linear variation of altitude from one end of the barrier to the other. Using multiple line barriers, it is possible to create complex viewing areas that approximate actual radar viewing patterns.

Wedge barriers are simply pie shaped sections in which nothing is visible. An example is shown in Figure VII-1 in the plan view to the west of the viewer. The user specifies the start and end compass headings of the wedge.

The system of restricting the field of view described above is not a perfect representation of radar vision limits, but it permits a reasonable approximation for modeling purposes.

The IHAWK permits the operators to specify a "sector of interest" which is a pie shaped segment of its viewing area. The IHAWK computer will automatically process tracks in this sector. The sector of interest has no effect on the radar's ability to acquire aircraft. It serves only to delineate a high interest area to the computer. MOPADS has a facility to specify a sector of interest for each viewer. In the current implementation, the sector of interest is used only by IHAWK.

1-4. DELETE. The DELETE command (Table VII-1c) is used to destroy a SIMULATION-DATA-SET directory and all of the information which descends from it. No information from a deleted simulation data set can be retrieved.

1-5. INSERT. The INSERT command (Table VII-1d) copies a Reference ADSM (code 11) directory to the current directory. It is used to construct a command and control configuration in a SIMULATION-DATA-SET. For example, in order to create the configuration shown in Figure II-2, the COMMAND-AND-CONTROL directory would have been made current then an INSERT command such as

INSERT ADSM="G-GROUP-AN/TSQ-73"

would have been issued. This would have copied the Group AN/TSQ-73 reference system module to the position shown (i.e., "owned by" the COMMAND-AND-CONTROL directory). After INSERT was complete, the

COMMAND-AND-CONTROL directory will still be current. The SELECT command would be used to make the Group working ADSM current, and then INSERT would be issued twice to create the two Battalion AN/TSQ-73 ADSM's which descend from the Group ADSM. Similar procedures would be used to create the working copies of the IHAWK.

INSERT assigns directory labels and values to ID(3) and ID(4) in a systematic manner to code 12 directories that it creates. Recall from Table IV-1 that ID(4) of reference ADSM (code 11) directories is the Basic ACN. This is a multiple of 1000 and is unique to the system module type. For example, the IHAWK Basic ACN is 4000. As code 12 directories are created, the Basic ACN is incremented, so the working IHAWK directories in Figure II-2 would have values of ID(4) of 4001, 4002, and 4003. Which copy would have which value depends on the order in which they were created. In this way, every working ADSM in a simulation data set has a different value for its ID(4).

The value for ID(3) is sequentially assigned within a directory. What this means is that the two IHAWKs that belong to the right hand side battalion in Figure II-2 would have values of one and two for ID(3). ID(3) for the IHAWK of the left battalion would have a value of one. Likewise, the two battalions would have ID(3) values of one and two. Their ID(4) values would be 3001 and 3002 because 3000 is the Basic ACN for the Battalion AN/TSQ-73.

Labels are created using the system module one character ACC and the value of ID(3). The ACC's for the current MOPADS models are:

G	-	Group AN/TSQ-73
Q	-	Battalion AN/TSQ-73
W	-	IHAWK

Thus, the group ADSM in Figure II-2 has label G1. The two battalions have labels G1Q1 and G1Q2, and the IHAWK labels might be G1Q1H1, G1Q2H1, and G1Q2H2. In other words, the ACC and the value of ID(3) are appended as a suffix to the label of its superior unit. In this way, each working ADSM has a unique label.

1-6. REMOVE. REMOVE (Table VII-1e) will delete a working ADSM. The ADSM to be deleted is specified by giving its label, and it must belong to the current directory.

2-0 ANNOTATED EXAMPLES

2-1. The ADD, DELETE, INSERT, and REMOVE Commands. Figure VII-2 is an example of a terminal session. The annotations are explained below.

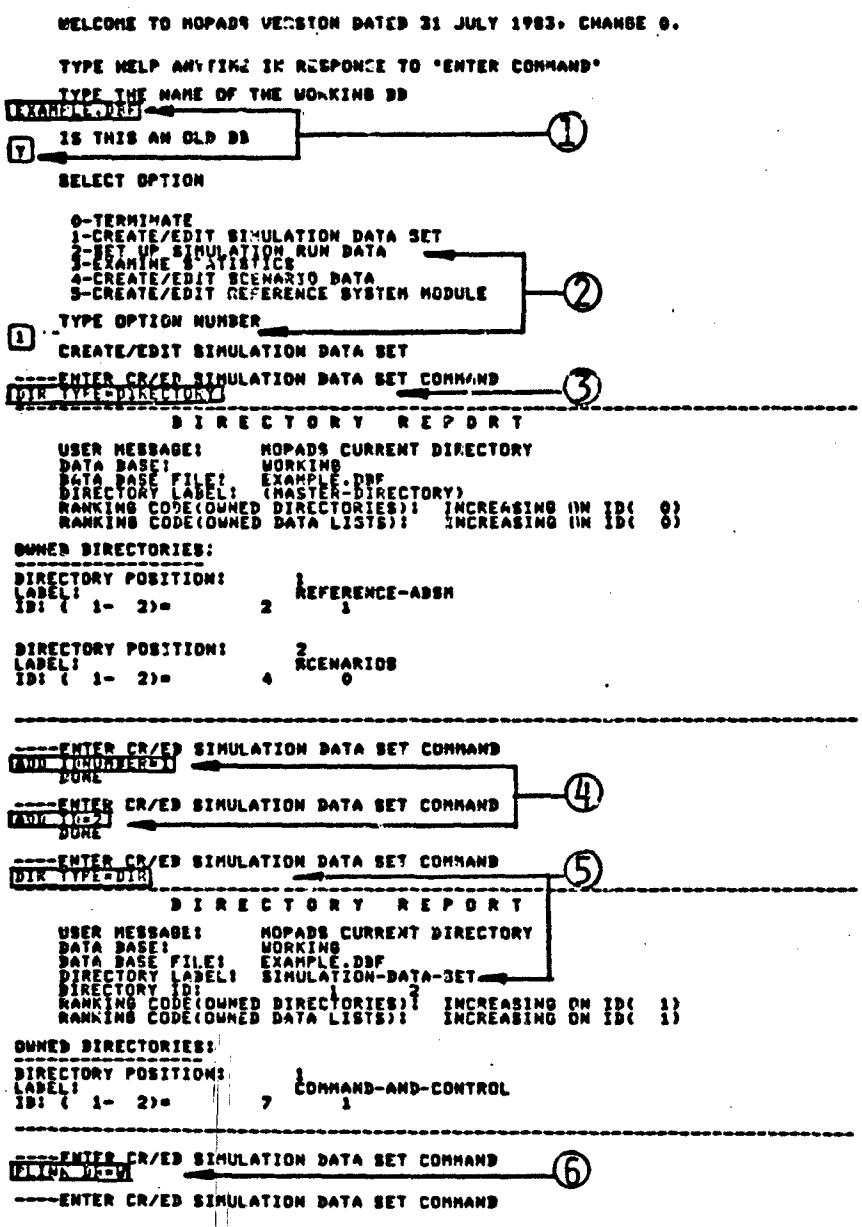


Figure VII-2. Examples of CREATE/EDIT SIMULATION-DATA-SET Commands.

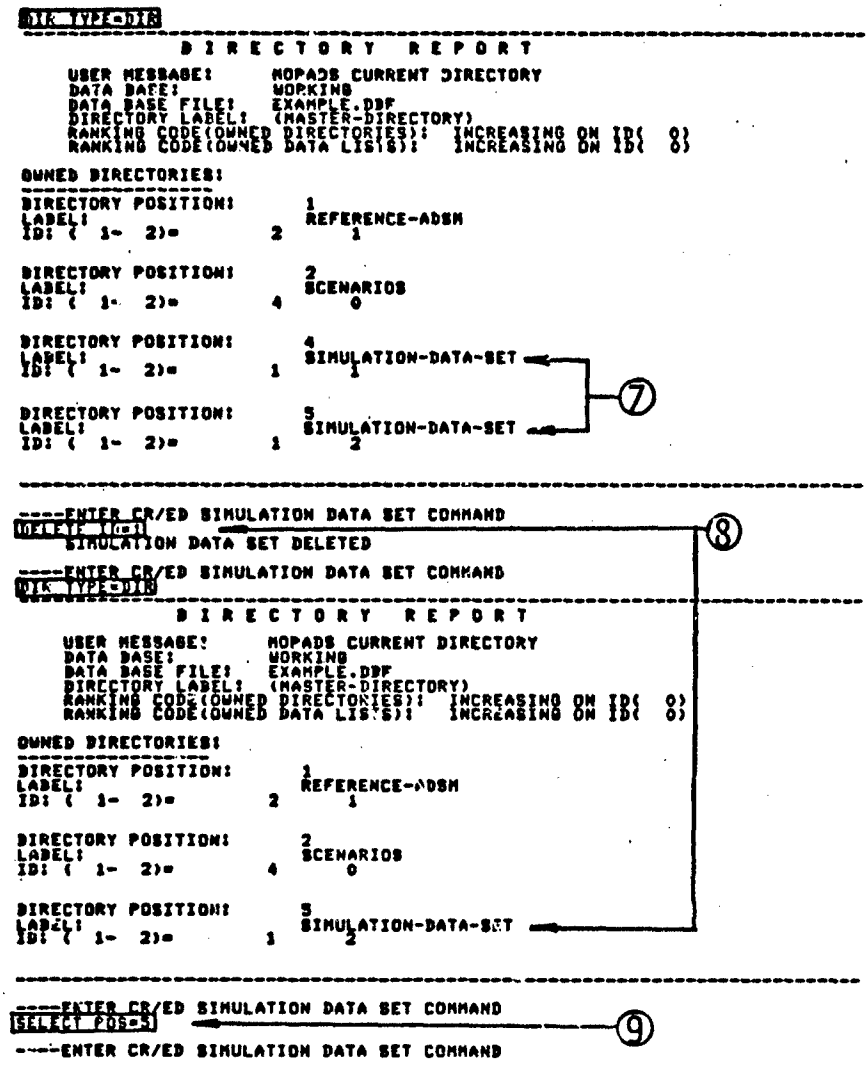


Figure VII-2. (Continued)

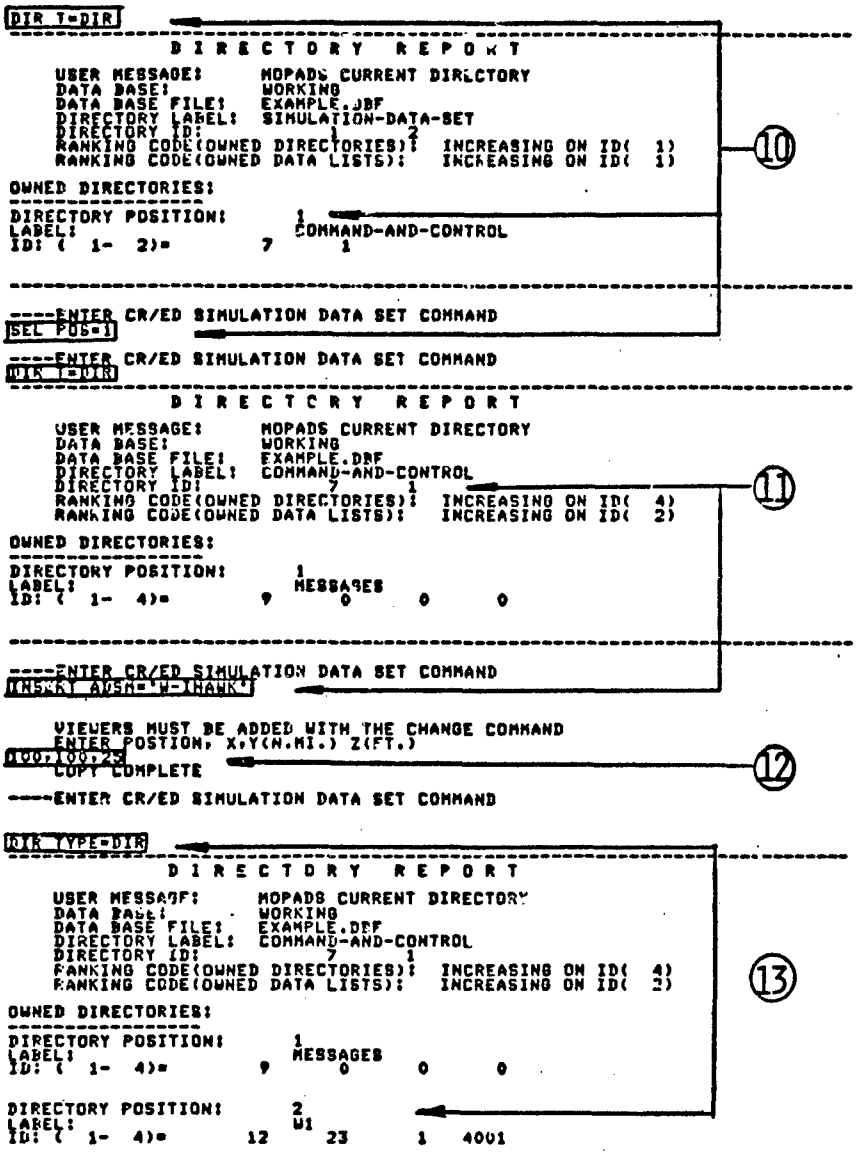


Figure VII-2. (Continued)

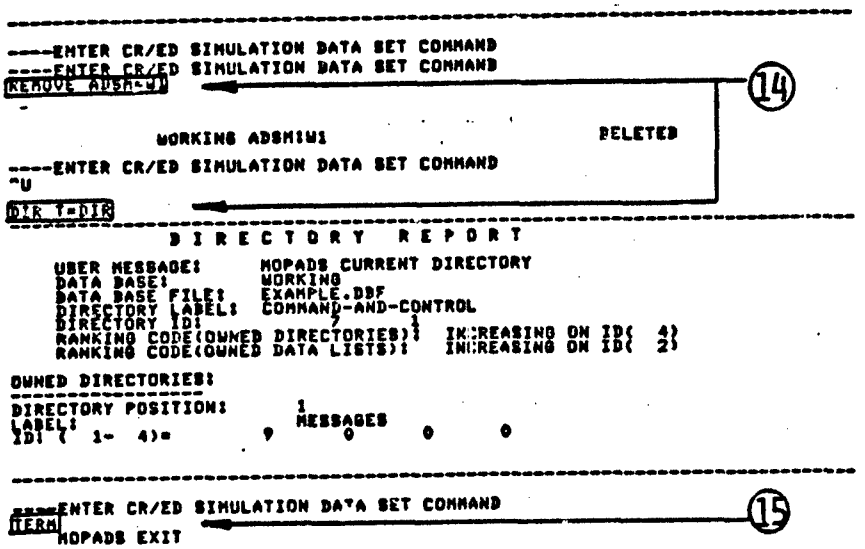


Figure VII-2. (Continued)

(1) As part of the log-on protocol, MOPADS requests the name of the file to be the working data base. If it is not an old (previously created) data base, the User Interface creates an empty MOPADS data base on the file.

(2) The main menu requires the user to select a subprocess. In this case, we select process one.

(3) The (MASTER-DIRECTORY) is current and it contains no simulation data sets.

(4) The ADD command is issued twice - Note the abbreviation of IDNUMBER in the second ADD command.

(5) The DIRECTORY command reveals that Simulation Data Set number two (see ID(2)) is current. It was the last one created.

(6) Use PLINK to make the (MASTER-DIRECTORY) current again.

⑦ DIRECTORY confirms that two simulation data sets have been created. They are numbered one and two as can be seen from their ID(2) values.

⑧ DELETE eliminates simulation data set one. This is confirmed by DIRECTORY.

⑨ Make simulation data set two current.

⑩ The simulation data set contains only the COMMAND-AND-CONTROL directory. Make it current with SELECT.

⑪ Copy the IHAWK ADSM to the COMMAND-AND-CONTROL directory with INSERT. This may take several minutes depending upon the system module being copied and the speed of the computer.

⑫ The position of the working ADSM must be specified with respect to the reference point. Therefore, the user must already have in mind the scenarios that this simulation data set will be used with.

⑬ DIRECTORY shows that COMMAND-AND-CONTROL is still current, and that the IHAWK has been copied with label W1. Note ID(3)=1 and ID(4)=4001.

⑭ REMOVE deletes the working ADSM labeled W1 and DIRECTORY confirms that it has been removed.

⑮ The User Interface session is ended with the TERMINATE command.

2-2. The CHANGE Command. The CHANGE command has subeditors to change each of the data lists in an ADSM. Figure VII-3 is an interactive session to edit this data list. The annotations are explained below.

① The CHANGE command specifies that the "TD" (task data) data list is to be edited.

② Data is entered by specifying a code character for the operation to be performed. The code characters are:

- S - show parameters for a node
- M - print the menu of code characters
- E - enter data for a new node. "E" may be used only once for any node to create it. This feature is used also in the CREATE/EDIT REFERENCE

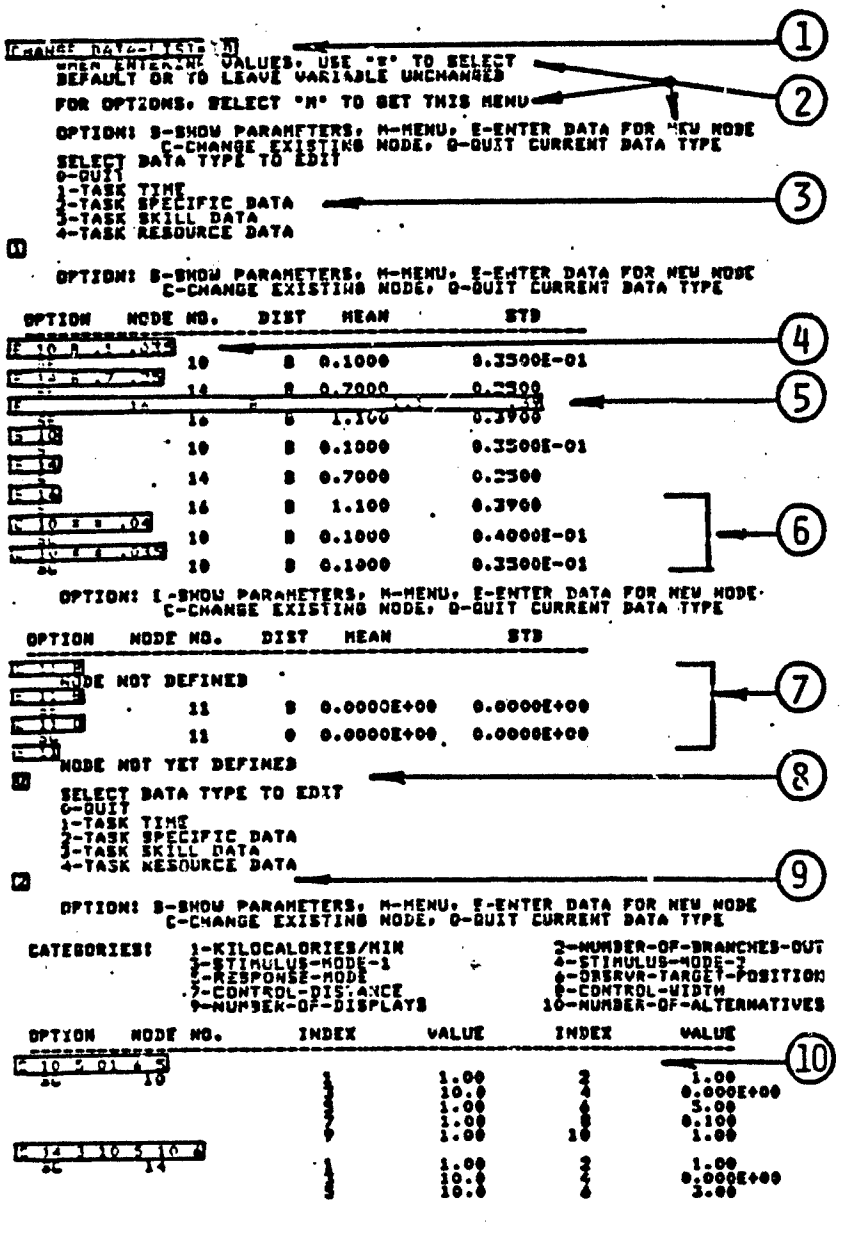


Figure VII-3. CHANGE Task Data Example.

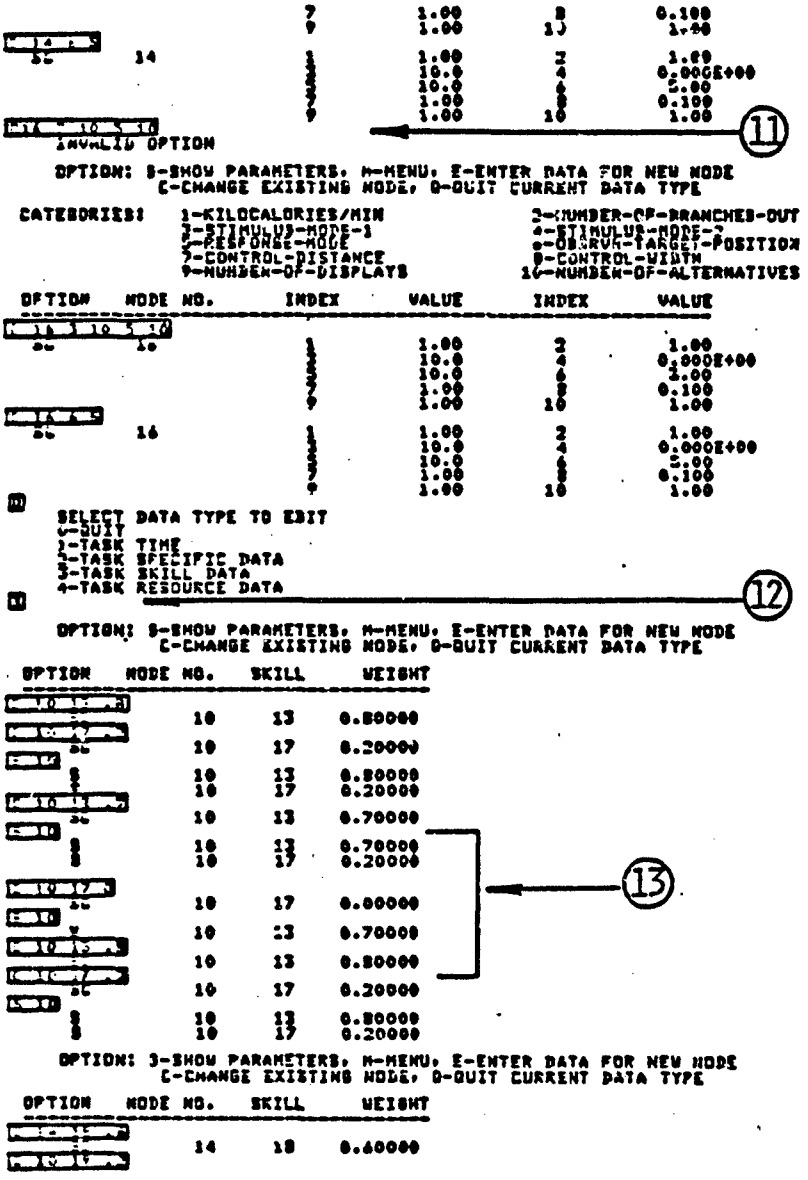


Figure VII-3. (Continued)

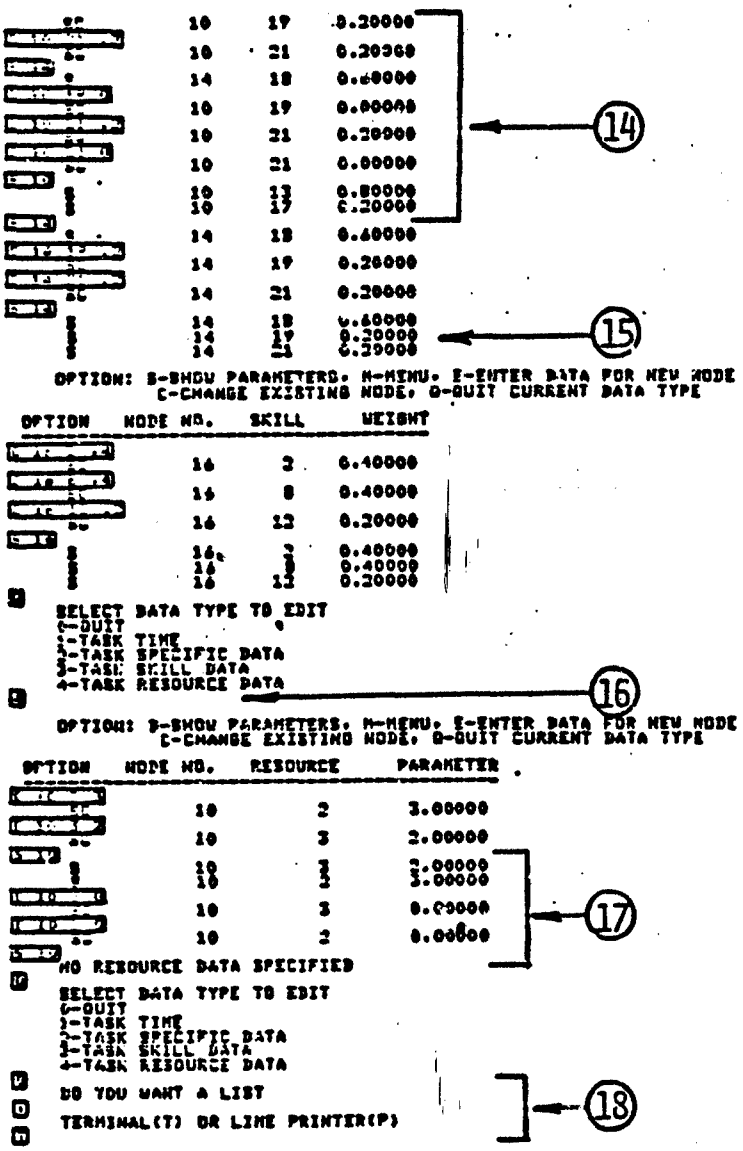


Figure VII-3. (Continued)

SYSTEM MODULE subprocess. This subeditor is common across the two subprocesses.

C - change parameters for an existing node. "C" must be used for all edits of a node after it is created with "E".

Q - quit editing the current type of task data

When defaults are appropriate or an old value is to be left unchanged, type an asterisk (*) for the value.

(3) There are four types of data that may be specified for each node, and they are edited separately. The user has chosen to begin with task time data.

(4) The data to be entered for task data are node number, distribution type, mean, and standard deviation. The input line creates node 10 with distribution type 10, mean = 0.1 and standard deviation = 0.035. The values entered must be separated by one or more blanks or commas. They need not line up under the labeled columns. MOPADS echoes the command data on the next line. The option echoed is the user option preceded by "S" (for Show). If an error is made that is echoed, the "S" will be replaced by an "X" (e.g., "XE").

(5) This line demonstrates that the spacing between input data elements is not significant.

(6) When using the "S" option (e.g., S 10), only the node number is required.

As a demonstration, the standard deviation of node 10 is changed to 0.04 and then back to 0.035. Note that the asterisks caused the distribution type and the mean to be unchanged.

The option menu and column headings have been printed again. They are printed every 15 lines so they will always be visible on most scrolling screen terminals.

(7) The "C" option is invalid for node 11 because it does not exist. The "E 11 8" line creates node 11 with distribution 8 and default mean and standard deviation.

NOTE: DELETING NODES

If the 'C' option sets the distribution type to zero, the node is deleted. This is the only way a node can be deleted. The "E" option can be used to re-create a node, but previous parameter values (e.g., mean) will be lost.

This capability is used to delete node 11.

⑧ The "Q" terminates editing of task time data and re-prints the data type menu.

⑨ Begin editing task specific human factors parameters. The 10 categories with their associated index values (1 through 10) are printed.

⑩ This "C" command sets parameters 5 to one and parameter 6 to 5. Then MOPADS echoes the parameter values for node 10. When a node is created it automatically has default values for each task specific parameter. All 10 current values are printed each time the node is referenced. Note that in the echo, parameter 5 has value one and parameter 6 has value 5.

⑪ If an invalid option ("C16") is specified, MCPADS prints the option menu.

⑫ Begin editing skill data. One skill may be specified on each command line. No skills are specified for a task by default. All skill data must be specified explicitly with the CHANGE command.

⑬ Skills 13 and 17 were specified for node 10. The command "C 10 17 0" deletes skill 17 for node 10. This is reflected by the subsequent "S 10" command. Also, existing skills may have their weights changed by issuing a "C" command (e.g., "C 10 13 .8").

⑭ Incorrect skills were entered for node 10, and the above capability was used to remove the skills from node 10 and add them to node 14.

⑮ Examine skills for node 14 and be sure they are correct before moving on to node 16.

⑯ Begin to edit ADSM resource data.

⑰ No ADSM resource data was specified for this example, but for completeness resource data is entered for node 10 and then removed. This removal is accomplished by issuing a "C" command with resource parameters equal to zero.

⑱ Before exiting from the CHANGE command, a listing of task parameters may be obtained. It can be printed at the terminal or on the MOPADS file for non-interactive output. These listings should be scrutinized to ensure correct data entery.

Figure VII-4 shows the CHANGE command to edit Operator State Vectors. The annotations on the figure are explained below.

① Three types of data are contained in the operator state vectors that can be edited with CHANGE.

② Begin editing human factors data. There are 70 human factors data elements. The user can display any of them. Here elements 20 to 25 are selected.

③ MOPADS prompts for elements to be edited one at a time. Both the label and value may be changed although normally there is little utility in changing the label since the moderator functions specify human factors data by element number. This however, does not preclude specialized procedures, written into the implementation of a particular system module which might make changing a label desirable.

④ Use the asterisk (*) to avoid changing existing values.

⑤ When done editing, type zero for the element number. This brings up the display option again.

⑥ New values for elements 24 and 25 are shown.

⑦ A listing may be obtained when done editing each data type.

⑧ Editing goal parameters is similar to editing human factors parameters except the labels may not be changed. Also, care must be taken in changing this data. See Polito (1983e) for a discussion of goal data.

```
CHANGE DATA-LIST-OP
    ENTER OPERATOR NUMBER TO EDIT(ZERO TO QUIT)
:
SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-HUMAN FACTORS PARAMETERS                                    (1)
    2-GOAL PARAMETERS
    3-OBJECTIVE FUNCTION PARAMETERS
1
    THERE ARE  29 ELEMENTS
    WHICH ELEMENTS WOULD YOU LIKE TO SEE                          (2)
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
20 25
    20   NUM-BACKGROUND-CHARACTERS          0.0000000E+00
    21   MESSAGE-BACKLOG                    0.0000000E+00
    22   SIGNALS-PER-MINUTE                  1.000000
    23   HOURS-WORKED-PER-WEEK               40.00000
    24   DAYS-WITHOUT-SLEEP                 0.0000000E+00
    25   DAYS-OF-NIGHT-DUTY                 0.0000000E+00
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)                           (3)
25
    TYPE NEW LABEL AND VALUE('S' FOR NO CHANGE)
S 1
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)                           (4)
24
    TYPE NEW LABEL AND VALUE('S' FOR NO CHANGE)
S 1
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE                          (5)
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
20 25
    20   NUM-BACKGROUND-CHARACTERS          0.0000000E+00
    21   MESSAGE-BACKLOG                    0.0000000E+00
    22   SIGNALS-PER-MINUTE                  1.000000
    23   HOURS-WORKED-PER-WEEK               40.00000
    24   DAYS-WITHOUT-SLEEP                  1.000000          (6)
    25   DAYS-OF-NIGHT-DUTY                  1.000000
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)                      (7)
0
    DO YOU WANT A LISTING?
N
SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-HUMAN FACTORS PARAMETERS
    2-GOAL PARAMETERS
    3-OBJECTIVE FUNCTION PARAMETERS
2
    THERE ARE  45 ELEMENTS
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 5
    1   GOAL                                1.000000
    2   LITTLE-M                           -0.1000000E+11
    3   BIG-M                               0.0000000E+00
    4   LITTLE-A                            0.0000000E+00
    5   LITTLE-B                            0.0000000E+00
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
16 30
   16   GOAL                                2.000000          (8)
   17   LITTLE-M                            25.00000
   18   BIG-M                               0.1000000E+11
   19   LITTLE-A                            0.3125343E-02
   20   LITTLE-B                            2.848437
   21   BIG-A                               0.0000000E+00
   22   BIG-B                               0.0000000E+00
   23   GOAL-STATE-LOW-1                    0.0000000E+00
   24   PRIORITY-LOW-1                      30.00000
   25   GOAL-STATE-LOW-2                    8.000000
   26   PRIORITY-LOW-2                      10.00000
   27   GOAL-STATE-HIGH-1                   0.0000000E+00
   28   PRIORITY-HIGH-1                     0.0000000E+00
   29   GOAL-STATE-HIGH-2                   0.0000000E+00
   30   PRIORITY-HIGH-2                     0.0000000E+00
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
    DO YOU WANT A LISTING?
N
SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-HUMAN FACTORS PARAMETERS
    2-GOAL PARAMETERS
    3-OBJECTIVE FUNCTION PARAMETERS
```

Figure VII-4. CHANGE Operator Data Example.

```
3
   THERE ARE   3 ELEMENTS
   WHICH ELEMENTS WOULD YOU LIKE TO SEE
   ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 2
      1   OBJECTIVE-FUNCTION                1.000000
      2   NUMBER-OF-GOALS                   2.000000   <-- (9)
   TYPE ELEMENT TO CHANGE( 0 FOR NONE)
1
   TYPE NEW LABEL AND VALUE('*' FOR NO CHANGE)
1 2
   TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
   WHICH ELEMENTS WOULD YOU LIKE TO SEE
   ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 2
      1   OBJECTIVE-FUNCTION                2.000000
      2   NUMBER-OF-GOALS                   2.000000
   TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
   WHICH ELEMENTS WOULD YOU LIKE TO SEE
   ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
   DO YOU WANT A LISTING?
Y
   TERMINAL(T) OR LINE PRINTER(P)             <-- (10)
T
```

```
PAGE=    1
                D A T A   L I S T   R E P O R T

    USER MESSAGE:      OPERATOR-1
    DATA BASE FILE:    VDL46.DBF
    DATA LIST LABEL:   OP-OPERATOR-STATE-VECTOR
    DATA LIST ID:      ( 1- 3)=          5    1516       1
    DATA LIST TYPE:    REAL

    DATA LIST CONTENTS:

    ROW:       1
```

COLUMN	VALUE	LABEL
1	1.000000	OPERATOR-TYPE
2	0.0000000E+00	OPERATOR-ID
3	32.00000	POINTER-TO-HF-TASK-DATA
4	97.00000	POINTER-TO-GOAL-PARAMETER
5	142.0000	POINTER-TO-OBJ-FUNCTION
6	144.0000	POINTER-TO-DISPLAY-INFO
7	147.0000	POINTER-TO-TASK-STACK
8	0.0000000E+00	UNUSED
9	0.0000000E+00	UNUSED
10	0.0000000E+00	UNUSED
11	0.0000000E+00	UNUSED
12	0.0000000E+00	UNUSED
13	0.0000000E+00	UNUSED
14	0.0000000E+00	UNUSED
15	0.0000000E+00	UNUSED
16	0.0000000E+00	UNUSED
17	0.1000000E+21	START-TRACE-TIME
18	0.1000000E+21	END-TRACE-TIME
19	0.1000000E+21	TRACE-INCREMENT
20	0.0000000E+00	UNUSED
21	0.0000000E+00	UNUSED
22	0.0000000E+00	UNUSED
23	0.0000000E+00	UNUSED
24	0.0000000E+00	UNUSED
25	0.0000000E+00	UNUSED
26	0.0000000E+00	UNUSED
27	0.0000000E+00	UNUSED
28	0.0000000E+00	UNUSED
29	0.0000000E+00	UNUSED
30	0.0000000E+00	UNUSED
31	0.0000000E+00	UNUSED
32	37.00000	CORE-TEMPERATURE
33	1.000000	CID-VALUE
34	0.0000000E+00	TIME-ON-TASK
35	0.0000000E+00	DAYS-OF-DUTY
36	1.000000	SEARCH-DIMENSIONS
37	0.0000000E+00	NUMBER-FIRE-UNITS

(11) <-- OPERATOR-TYPE

(12) <-- CORE-TEMPERATURE

Figure VII-4. (Continued)

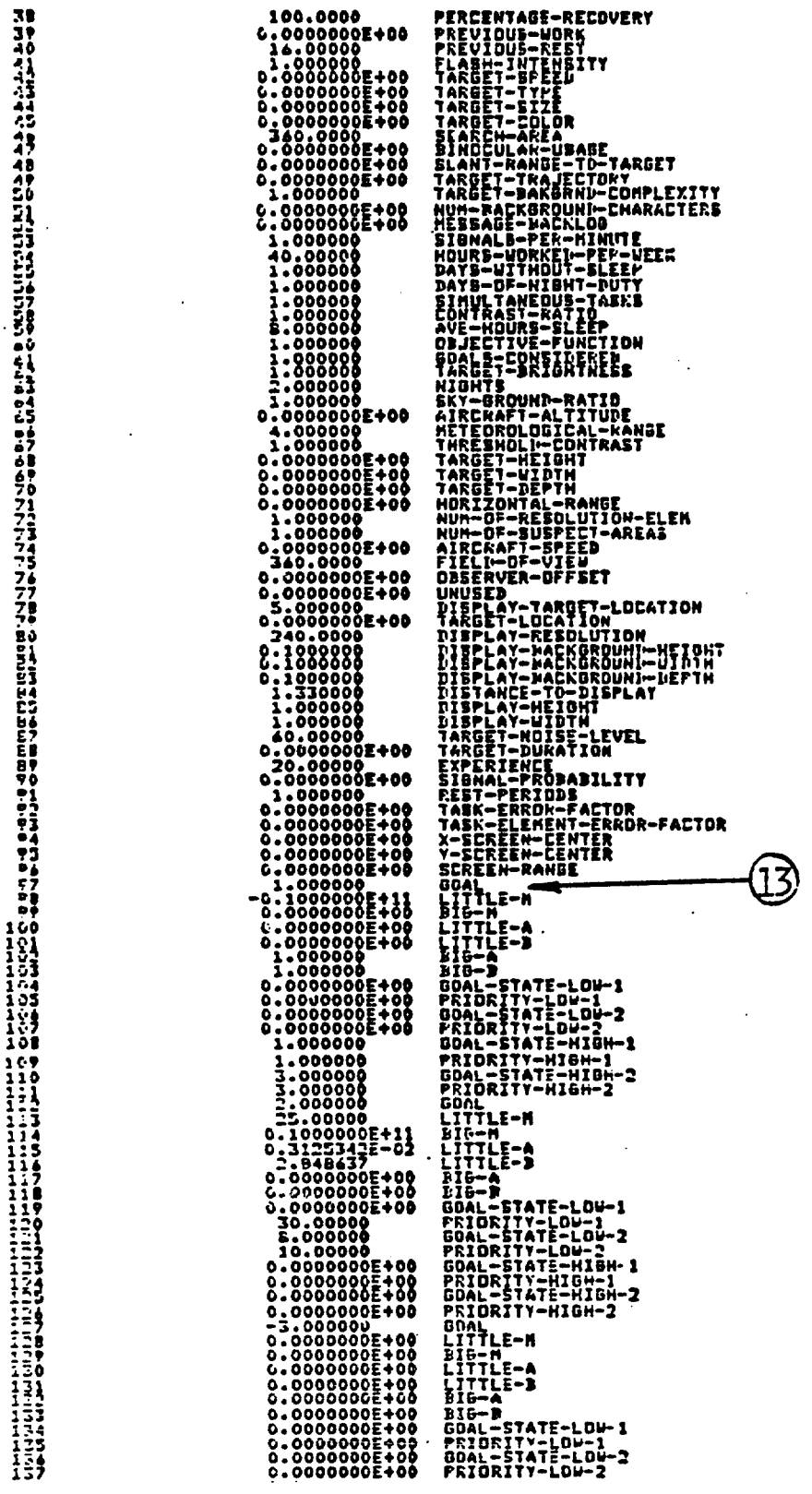


Figure VII-4. (Continued)

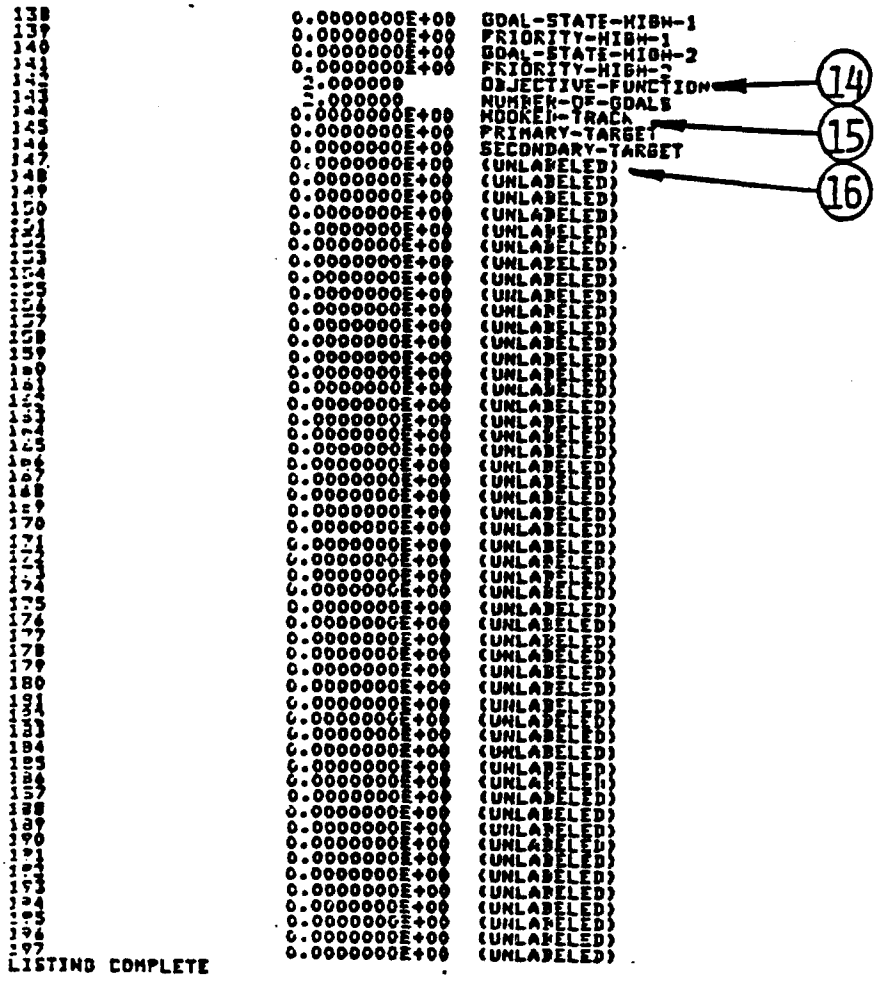


Figure VII-4. (Continued)

Each operator goal requires 15 goal elements (in the example there are 3 goals, hence 45 goal elements). In Figure VII-4, elements 23 to 30 are the points on the goal state vs. goal priority curve input by the user with the ADD command. Elements 17 to 22 are computed from these points to fit exponential curves to the points (see Polito (1983e)). No automatic computation of elements 17 to 22 is performed by CHANGE. In order to change an operator's goal curves, these elements must be computed manually and then edited with CHANGE.

(9) Two objective function parameters may be edited. See Polito (1983e) for a discussion of these parameters.

(10) The operator state vectors are stored with pointers to the various types of data (i.e., human factors, goal parameters, etc.). See (11). Thus, the human factors data begins at element 32 (12).

To find human factor parameter 6, for example, compute its index as follows: 32+6-1 = 37. Column 37 of the operator state vector is human factors parameter 6.

Similarly, the goal parameters start at column 97, (13), and the objective function data begins at column 142, (14).

Finally, the display and task stack information begins at columns 144, (15), and 147, (16), respectively. These data cannot be edited because they change dynamically during simulations.

Figure VII-5 is an example of editing the Environmental State Vector for the reference ADSM.

(1) & (2) The environmental state vector contains system and human factors data. Editing is performed in the same way as for human factor parameters in the operator state vectors.

(3) The listing is also in the same format as the operator state vectors. System parameters begin at element 2, (4), and human factors data starts at element 23, (5)

```
CHANGE DATA-LIST=FM
 SELECT DESIRED PARAMETERS TO EDIT
   0=QUIT
   1=SYSTEM PARAMETERS
   2=HUMAN FACTORS PARAMETERS <----------(1)
1  THERE ARE  21 ELEMENTS
   WHICH ELEMENTS WOULD YOU LIKE TO SEE
   ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 5
      1   SYSTEM-MODE                      1.000000
      2   OPERATOR-MODE                    0.000000E+00
      3   DEFCON                           0.000000E+00
      4   METHOD-OF-CONTROL                1.000000
      5   WEAPONS-CONTOL-STATUS            2.000000
   TYPE ELEMENT TO CHANGE( 0 FOR NONE)
3
   TYPE NEW LABEL AND VALUE("?" FOR NO CHANGE)
0 1
   TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
   WHICH ELEMENTS WOULD YOU LIKE TO SEE
   ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0  DO YOU WANT A LISTING?
```

Figure VII-5. CHANGE Environmental Data Example.

```
SELECT DESIRED PARAMETERS TO EDIT
 0-QUIT
 1-SYSTEM PARAMETERS
 2-HUMAN FACTORS PARAMETERS

 2

 THERE ARE   9 ELEMENTS
 WHICH ELEMENTS WOULD YOU LIKE TO SEE
 ENTER FIRST AND LAST ELEMENTS(0 TO STOP)

 1,9

    1   DRY-BULB-TEMPERATURE              72.00000
    2   RELATIVE-HUMIDITY                 50.00000
    3   AIR-MOVEMENT-RATE                  4.000000
    4   NOISE-LEVEL                       45.00000
    5   WORKING-AREA-ILLUMINATION         50.00000
    6   NUMBER-ON-DUTY                     1.000000
    7   VIBRATION                      0.0000000E+00
    8   AMBIENT-VAPOR-PRESSURE            10.00000

 TYPE ELEMENT TO CHANGE( 0 FOR NONE)

 5

 TYPE NEW LABEL AND VALUE('S' FOR NO CHANGE)

 S,72

 TYPE ELEMENT TO CHANGE( 0 FOR NONE)

 0

 WHICH ELEMENTS WOULD YOU LIKE TO SEE
 ENTER FIRST AND LAST ELEMENTS(0 TO STOP)

 0,0

 DO YOU WANT A LISTING?

 N
SELECT DESIRED PARAMETERS TO EDIT
 0-QUIT
 1-SYSTEM PARAMETERS
 2-HUMAN FACTORS PARAMETERS

 1

 THERE ARE  21 ELEMENTS
 WHICH ELEMENTS WOULD YOU LIKE TO SEE
 ENTER FIRST AND LAST ELEMENTS(0 TO STOP)

 0,0

 DO YOU WANT A LISTING?

 Y

 TERMINAL(T) OR LINE PRINTER(P)

 T

 ENTER MESSAGE TO PRINT WITH IT

DATA BASE FILE:   VOL1&.BRF
DATA LIST LABEL:  EN-ENVIRONMENT-STATE-VEC
DATA LIST ID:     1-  3/0        5    314      1
DATA LIST TYPE:   REAL

DATA LIST CONTENTS:

ROW:      1

COLUMN          VALUE          LABEL
------  ---------------------  ------------------------------
     1          23.00000       POINTER-TO-HF-DATA
     2          1.000000       SYSTEM-MODE
     3          1.000000       OPERATOR-MODE
     4     0.0000000E+00
     5          1.000000       METHOD-OF-CONTROL
     6          2.000000       WEAPONS-CONTOL-STATUS
     7          1.000000
     8     0.0000000E+00       AMMUNITION-HOT
     9     0.0000000E+00       AMMUNITION-COLD
    10     0.0000000E+00
    11     0.0000000E+00
    12     0.0000000E+00
    13     0.0000000E+00
    14     0.0000000E+00
    15     0.0000000E+00
    16     0.0000000E+00
    17     0.0000000E+00
    18     0.0000000E+00
    19     0.0000000E+00       X-POSITION
    20     0.0000000E+00       Y-POSITION
    21     0.0000000E+00       Z-POSITION
    22          3.000000       SAMPLING-OPTION
    23          72.00000       DRY-BULB-TEMPERATURE
    24          50.00000       RELATIVE-HUMIDITY
    25          4.000000       AIR-MOVEMENT-RATE
    26          45.00000       NOISE-LEVEL
    27          45.00000       WORKING-AREA-ILLUMINATION
    28          1.000000       NUMBER-ON-DUTY
    29     0.0000000E+00       VIBRATION
    30          10.00000       AMBIENT-VAPOR-PRESSURE

LISTING COMPLETE
```

CHANGE system resources were not defined for the example, but an example of defining such data is given in Figure VII-6. The data which may be specified for a resource is:

1. Resource Number
2. Number of Units Available
3. Mean Time Between Failures (MTBF) (hrs)
4. Mean Time to Repair (MTR) (hrs)
5. Use Code (0-non-exclusive use, 1-may be used by only one operator)
6. Status (1-green, 2-amber, 3-red)

Editing is performed in the same way as for Task data and the listing (not shown) is formatted in the same way.

```
CHANGE DATA-LIST=P
    WHEN ENTERING VALUES, USE 'S' TO SELECT
    DEFAULT OR LEAVE UNCHANGED
    FOR OPTIONS, SELECT 'H' TO GET THIS MENU

    OPTION: S-SHOW PARAMETERS, M-MENU, E-ENTER DATA FOR NEW RESOURCE
            C-CHANGE EXISTING RESOURCE, D-QUIT RESOURCE
 OPTION  RESOURCE NO.-UNITS     MTBF          MTR       CODE  STATUS
E 1 1 100. .8 1 1
  LABEL?
RADIO
  SC          1          1     100.0        0.8000        1       1
  LABEL=RADIO
C 1 2 75
  LABEL?
D
  SC          1          1      75.00       0.8000        1       1
  LABEL=RADIO
C 1 3 0
  SC          1          0  0.0000E+00   0.0000E+00       0       0
  LABEL=(UNUSED)
F 1
  RESOURCE NOT DEFINED
D
N
  DO YOU WANT A LISTING

----ENTER CR/ED ADBN COMMAND
```

Figure VII-6. CHANGE System Resource Data Example.

Figure VII-7 shows the CHANGE command to edit the field of view information. The annotations are explained below.

(1) The DIRECTORY AND SELECT commands are used to make the IHAWK directory W1 current.

(2) The CHANGE command indicates that the FIELD-OF-VIEW (FV) data list is to be edited.

(3) A menu of viwer editing options is given. Select "4" to create viewers.

(4) Viewers have a user specified number. In this case, "1".

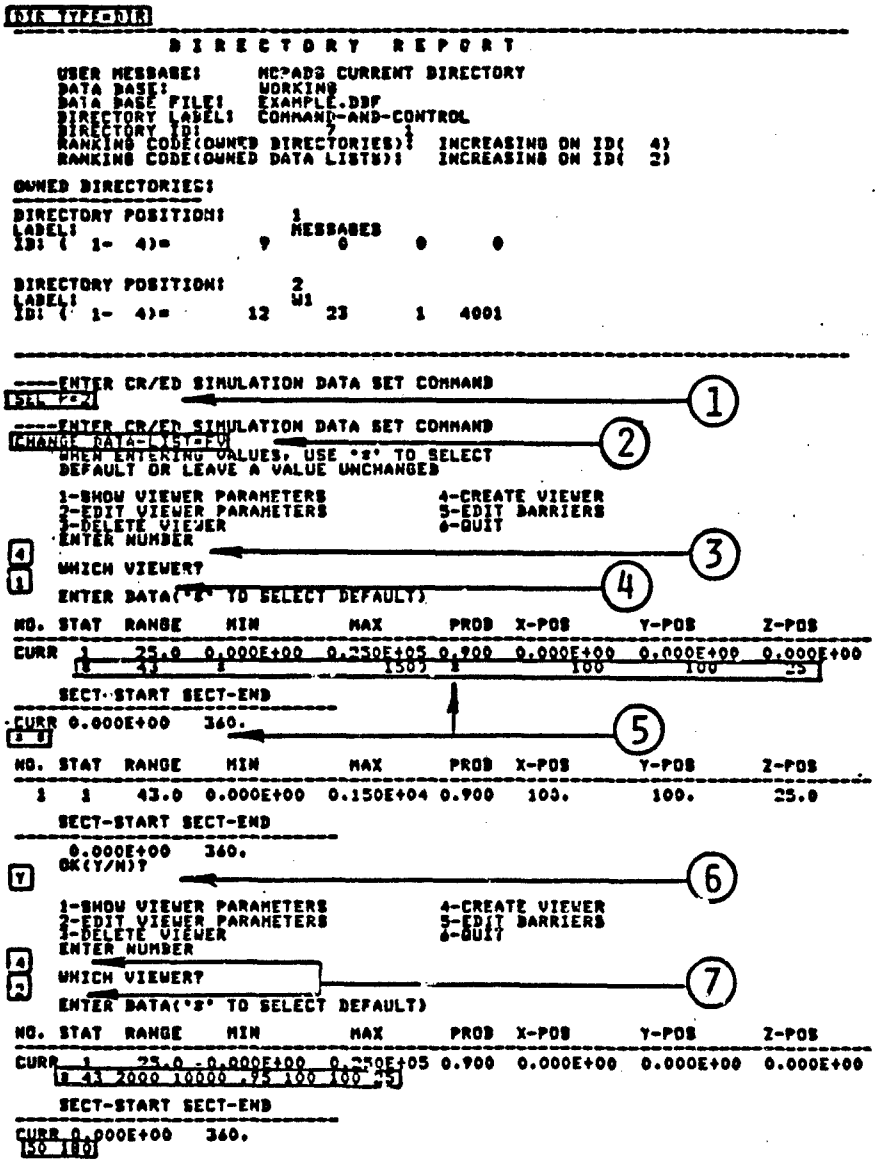


Figure VII-7. Annotated Example of the CHANGE Command Viewers.

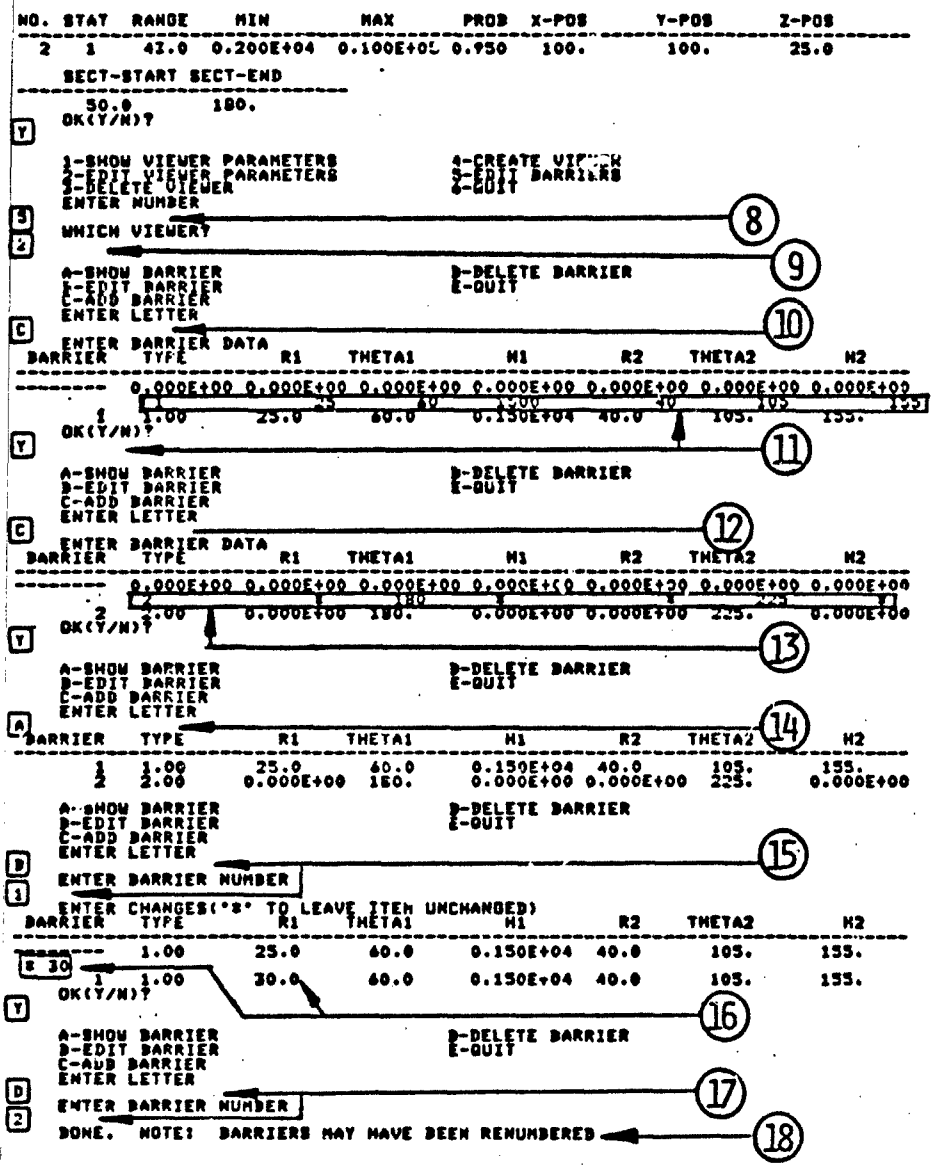


Figure VII-7. (Continued)

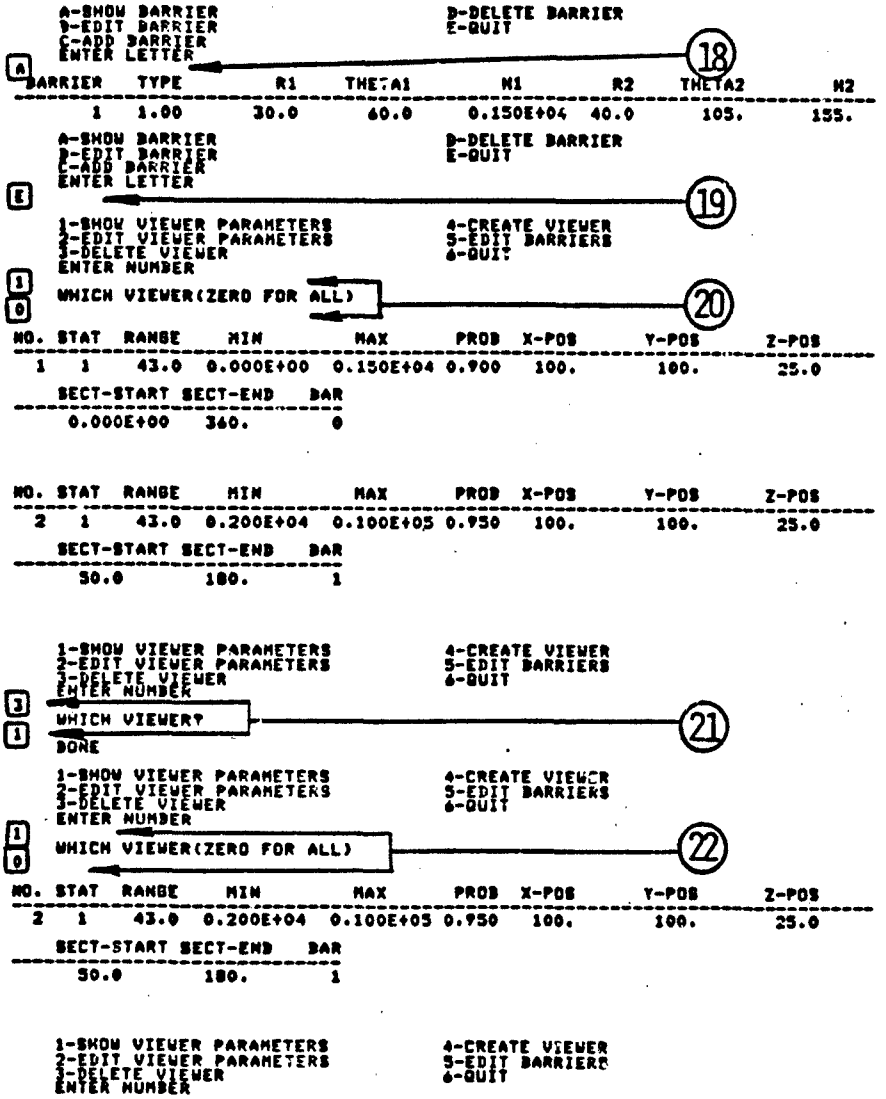


Figure VII-7. (Continued)

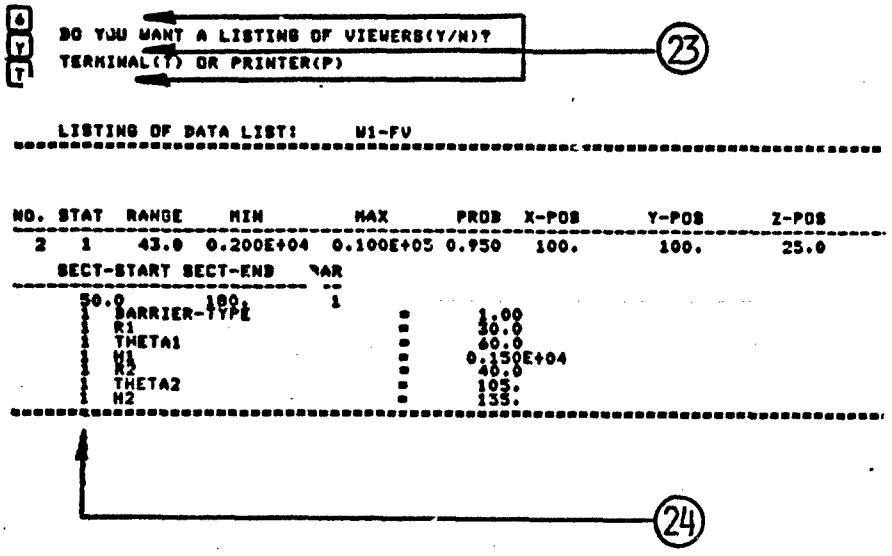


Figure VII-7. (Continued)

(5) CHANGE prints a default viewer;the parameters are:

STAT		1: viewing 2: not viewing
RANGE	-	range in nautical miles
MIN	-	minimum viewing altitude in feet above or below the viewer altitude (see Z-POS)
MAX	-	maximum viewing altitude above or below the viewer altitude
PROB	-	probability the viewer sees on aircraft
X-POS,Y-POS Z-POS	-	coordinates of the viewer position. X and Y are in nautical miles, Z is feet
SECT-START, SECT-END	-	start and end true compass azimuth of the sector of interest in decimal degrees

(6) After data is entered, MOPADS prints the new values and asks if they are ok.

⑦ A second viewer is created. Note that the data need not be typed exactly under the headings printed by MOPADS. The only requirement is that the data be typed in the correct order and the data fields be separated by blanks or commas.

⑧ Existing viewers can be edited.

⑨ Edit the second viewer.

⑩ MOPADS prints a viewer editing menu and asks for a selection. "C" indicates that a barrier will be defined.

⑪ Data entry for barriers is similar to that for viewers. The data are:

TYPE	-	barrier type 1- line barrier 2- wedge barrier
R1	-	distance of one end of a line barrier from the viewer (in nautical miles)
THETA1	-	true compass azimuth from the viewer to the end of the barrier (in decimal degrees)
H1	-	altitude above or below the viewer altitude of the end of the barrier (in feet)
R2,THETA2, H2	-	analogous to R1,THETA1, and H1 for the other end of a line barrier

See ⑬ for wedge barriers

⑫ Add a wedge barrier now.

⑬ For a wedge barrier, only THETA1 and THETA2 are significant. The barrier is from THETA1 clockwise to THETA2.

⑭ "A" shows all defined barriers. Note that each has been assigned a number by MOPADS.

⑮ Barriers are edited in the same way as viewers.

⑯ Asterisks need not be entered explicitly if remaining fields on a line are to be defaulted.

⑰ Delete the second barrier.

⑱ MOPADS may renumber the barriers after deletion. For example, if there are five barriers, and the third one is deleted, MOPADS will renumber the remaining barriers one through four.

⑲ "E" returns to the viewer menu.

⑳ "1" prints selected viewers. Note the new field on the second line, "BAR". This is the number of barriers for the viewer.

㉑ Delete viewer number 1.

㉒ Note that the viewers are not renumbered. Viewer number two remains.

㉓ A listing of all viewers and barriers can be printed at the terminal or on the MOPADS line printer output file.

㉔ The column of "1"'s next to the barrier information indicates that this information is for barrier number on e.

VIII. SETTING UP SIMULATION RUN DATA

1-0 DATA REQUIREMENTS

1-1. Run Data.

There are nine data elements that must be specified prior to a MOPADS simulation. The first two are the critical asset configuration and the air scenario. Recall that a simulation data set is constructed with a particular critical asset configuration in mind.

The critical asset configuration is specified with the second element of the ID, ID(2), of the particular CRITICAL-ASSET-CONFIGURATION directory to be used. See Figure II-1 and Table IV-2. Similarly, the air scenario is specified by the second element, ID(2), of the ID of the appropriate directory. See Figure II-1 and Table IV-2 again. Naturally, the air scenario must belong to the specified critical asset configuration directory.

The number of simulation runs must also be specified. Each run is a replication of the entire simulation. Multiple runs are performed to obtain aggregate statistics over several replications. Replications of a MOPADS simulation should not be done unless some of the system modules have stochastic sampling options (see Section V, 3-0). If only deterministic sampling options are used, then only one run should be performed.

The start and end times of the simulation must also be specified. These are twenty four hour military times from 0000 to 2359.

MSAINT will produce an event trace if requested. The start and end runs must be specified for the trace. Zero run numbers will suppress the trace. Event traces are frequently interesting, but they produce a very great deal of output, so the user should not select a trace unless it is needed for some purpose.

MOPADS uses a separate random number stream for each working system module so the task times generated for each module are independent. In addition, it employs an initialization scheme to ensure that Task times are independent from run to run. This is accomplished by employing a special initialization stream from which the other random number streams are initialized. MOPADS has a default seed for this initialization stream, but the user may specify a seed for this stream if desired.

Finally, the track update interval can be specified. This is the maximum time, in seconds, between re-computation of the aircraft

coordinates. The default is 15 seconds. Note that this is the maximum time. As a practical matter, MSAINT will usually compute aircraft coordinates every second or so. The update interval is provided as a back-up to ensure that, in unusual cases, the tracks are updated frequently.

1-2. Fire Unit to Critical Asset Assignments.

Normally, fire units are positioned in order to protect one or more critical assets, and the critical assets are "assigned" to the fire unit. The critical assets are defined in the CRITICAL-ASSET-CONFIGURATION directory. They must be assigned to fire units before a simulation. Critical assets which are not assigned to fire units are not protected. Normally only one site is assigned to an IHAWK battery.

2-0 SET UP SIMULATION RUN DATA COMMANDS

The commands for this subprocess are shown in Table VIII-1. They are described briefly below.

2-1. ADD. The ADD command is used to create a new "TACTICAL-SCENARIO" directory. These directories are numbered by the user. The response to the prompt TSNUMBER becomes the number of the tactical scenario. A SIMULATION-DATA-SET directory must be current in order to issue this command.

2-2. DELETE. The DELETE command will delete a "TACTICAL-SCENARIO" directory from the current simulation data set. It will also destroy all information owned by the tactical scenario.

2-3. EDIT. EDIT is used to add or change run data for the tactical scenario. The data which can be changed with EDIT was described in Section 1-0 above.

2-4. USE. USE provides an easy way to make a simulation data set current. The response to the prompt IDNUMBER is the simulation data set that will become the current directory.

3-0 AN ANNOTATED EXAMPLE

The annotations are Figure VIII-1 are explained below.

(1) Select the CREATE/EDIT SIMULATION RUN/DATA subprocess from the main menu.

(2) The USE command attempts to make simulation data set number one current. Since no such simulation data set exists, an error message is printed.

(3) Simulation data set number two is made current.

Table VIII-1. SET UP SIMULATION RUN DATA Commands.

COMMAND	PROMPTS	RESPONSES	DEFAULTS
(a) ADD	TSNUMBER	positive integer	--
(b) DELETE	TSNUMBER	positive integer	--
(c) EDIT	TSNUMBER	positive integer	--
(d) USE	IDNUMBER	positive integer	--

④ The ADD command creates Tactical Scenario number two in the current simulation data set. All parameters in the tactical scenario assume default or null values. The EDIT command must be used to specify other values.

⑤ EDIT enters the editor.

⑥ Title information can be specified for each tactical scenario. Here only a single line of title information is entered. Type "/QUIT" when done entering the title.

⑦ Note the repeat of the question.

DO YOU WANT TO EDIT THE TITLE (Y/N)?

This editor will always permit the user to go back and re-edit data just edited. This is true of all data editing features in this editor.

The title editor permits lines of text to be replaced. It does not support character editing within a line.

⑧ When done editing the title, the editor moves on to run data. It prints the current values of the run parameters and prompts for changes.

⑨ Pairs of numbers represent element numbers and new values. For example, "4 0915" changes the start time to 0915. Note the re-prompt for editing run data again. If the response to this question was YES, the new values of the run parameters would be printed.

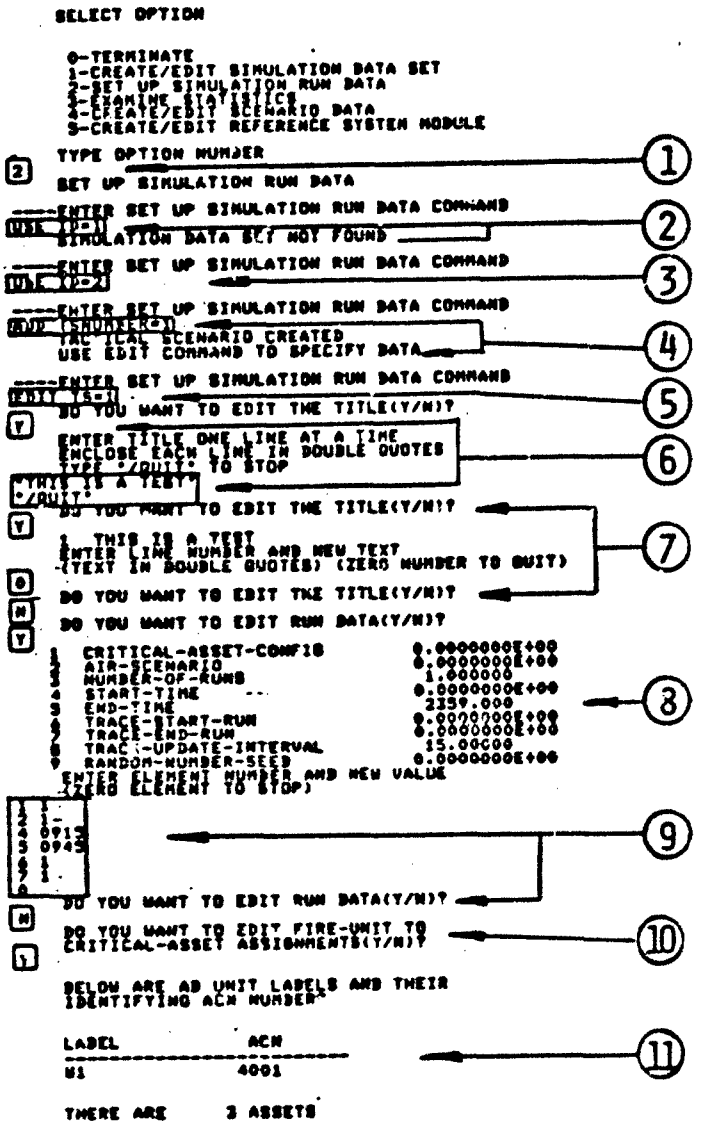


Figure VIII-1. Examples of SET UP SIMULATION RUN DATA Commands.

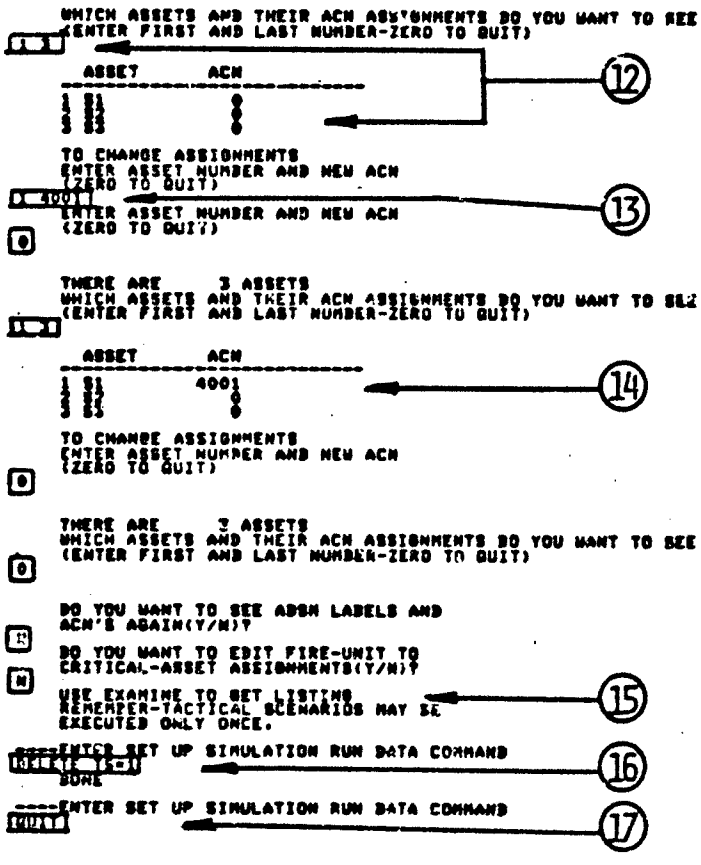


Figure VIII-1. (Continued)

(10) Begin to edit fire unit to critical asset assignments.

(11) The editor prints the ACN's of all air defense units in the simulation data set.

(12) Next the editor asks which asset assignments the user wants to see. The asset number, asset label, and the ACN it is assigned to are printed. Since this tactical scenario has not been edited before, there are no assignments.

(13) Assign asset one to be protected by the unit with ACN 4001.

(14) Print the assignments again to verify the new assignment.

⑮ Tactical scenarios contain the output statistics produced by MOPADS when they are simulated. In order to ensure that such information is not inadvertently destroyed, MOPADS will allow a Tactical Scenario to be used only once.

⑯ The DELETE command destroys the specified tactical scenario.

⑰ QUIT causes a return to the main menu.

IX. EXAMINING STATISTICS

1-0 MOPADS STATISTICS

1-1. Run Statistics.

MOPADS can collect and report two broad categories of statistics: Run and Summary. Run statistics are data collected for one run (or iteration) only. An example might be the average number of active tracks. If more than one run is performed, a value for this statistic would be reported for each run, and each of the values would be independent. In other words, the average number of aircraft in run two has no effect on the average in run three.

Summary statistics, on the other hand, are collected over runs. An example is the percent down time of an IHAWK launcher. Each run produces one observation of this statistic. The observations are averaged, and the average down time is reported once at the end of all runs. Summary statistics, then, obtain one observation for each run and report the mean, standard deviation, etc. of those observations. Summary statistics are discussed in Section 1-2 below. Run statistics are discussed in this section.

1-1.a Task Node Run Statistics and Histograms.

MSAINT task nodes can be designated as statistics nodes. If a node is so specified, MSAINT will automatically collect and report various statistics. The report has headings as shown below.

TASK NODE	LABEL	STAT TYPE	COLL TIME	AVERAGE	STD DEV	MINIMUM	MAXIMUM	NO. OBS

TASK NODE - The number of the task node.

LABEL - The label of the task node.

STAT TYPE - One of the following:

- FIR - record the time that the task is performed for the first time in a run.
- ALL - record the time that the task is performed for each time it is performed in a run.

BET - record the time interval between performance of the task.

INT - record the time interval to the time the task is performed from a previous time. This option requires that the previous time be set in an operator's attributes by another node.

NUM - record the number of times the task node is performed.

COLL TIME - The time that the observation is collected. It is one of the following:

REL - when the node is release.

STA - when the task node is started.

COM - when the task node is completed.

CLR - when the task node is cleared.

As an example, if STAT TIME is BET and COLL TIM is COM,then the node would record the time between completions of the task node.

AVERAGE, STD DEV - The sample average and standard deviation of all the observations taken during a run.

MINIMUM, MAXIMUM - The smallest and largest observation.

NO.OBS - The number of observations.

1-1.b Run Statistics Based on Observations (User).

MSAINT allows the modeler to collect any statistics of his choosing by providing subprograms that are called from modeler written FORTRAN code. An example might be the number of missiles fired per salvo. A value is collected each time a salvo is fired, and these observations are averaged. The report headings are shown below.

LABEL	MEAN	STD DEV	MINIMUM	MAXIMUM	NO.OBS

Each such statistic is assigned a label which is printed under the LABEL heading. The other headings are the same as explained above.

1-1.c Time Persistent Statistics (User).

Time persistent statistics are similar to statistics based on observation except that they are averaged over time. An example is the average number of active tracks. The number of active tracks persists until another track enters the simulation or until a track completes or is destoryed. The average is computed by weighting each observation by the length of time that is persists. The headings are shown below.

LABEL	MEAN	STD DEV	MINIMUM	MAXIMUM	INTERVAL

All of the headings are the same as described earlier except INTERVAL. INTERVAL is the length of time in minutes over which the statistics were collected. Normally this is the length of time that the simulation was run.

1-1.d Histograms (User).

User histograms are similar to task node histograms, except that the modeler must record observations by making subprogram calls to the appropriate statistics collection programs.

1-1.e Count Statistics.

Count statistics are simply a record of the number of times that a particular event occurred during a run. An example is shown below.

	COUNT
NUMBER OF SITES DESTROYED	1
NUMBER OF SITES ATTACKED	1
NUM HOST TRACKS NOT DEST	0
TOTAL NUMB HOSTILE TRACKS	3
TOTAL NUMB FRIEND TRACKS	1
TOTAL NUMB OTHER TRACKS	1
NUMB HOSTILE A/C DEST	2
NUMB FRIENDLY A/C DEST	0
NUMB OTHER A/C DEST	1
NUMB HOST TRACKS ATTACKED	2
NUMB FRND TRACKS ATTACKED	0
NUMB OTHR TRACKS ATTACKED	1

****

1-1.f Operator Task Fraction Statistics.

MOPADS reports the fraction (in percent) of the time that each operator spends in each operator task. (Note that operator tasks are not the same as task nodes; consult Goodin & Polito (1983a,b) for the AN/TSQ-73 and the IHAWK task lists. An example for an IHAWK Fire Control Operator (FCO) is shown below.

OPERATOR TASK FRACTION STATISTICS :RUN 1
AIR DEFENSE UNIT :W1
OPERATOR :FCO-A

TASK	TIME PERCENT
7	48.251
8	7.588
9	7.773
11	5.717
12	2.121
13	3.337
14	25.214

The MOPADS document Goodin & Polito (1983b) must be consulted to identify tasks 7, 8, etc. Tasks which are not performed are not reported.

1-2. Summary Statistics.

1-2.a SAINT Resource Utilization Statistics.

If a system module uses SAINT resources, then their utilizations will be available as summary statistics. The report headings are shown below.

RES NUM	LABEL	AVERAGE PCT BUSY	STD DEV	MINIMUM PCT BUSY	MAXIMUM PCT BUSY

RES NUM - The number assigned to the resource.

LABEL - The resource label.

AVERAGE PCT BUSY - If more than one run is performed, this is the average of the percent busy for each run.

STD DEV - The standard deviation of the percent busy.

MAXIMUM PCT BUSY, MINIMUM PCT BUSY - The largest and smallest percent busy observed.

1-2.b Task Node Summary Statistics.

MSAINT also automatically provides summary statistics of the TASK NODE RUN statistics. It does this with the following procedure. At the end of each run, the average value is computed for each statistic discussed in Section 1-1.a above. If more than one run is performed, the computed averages are themselves averaged to obtain a sample mean of the sample means. This data is reported as shown below.

TASK NODE	LABEL	STAT TYPE	COLL TIME	AVERAGE OF AVERAGE	STD DEV OF AVERAGES	MINIMUM AVERAGE	MAXIMUM AVERAGE	NO. OBS

The headings have the same meanings as in Section 1-1.a except as shown below.

AVERAGE OF AVERAGE - An estimate of the mean of the sample mean.

STD DEV OF AVERAGE - An estimate of the standard deviation of the sample mean.

MINIMUM AVERAGE, MAXIMUM AVERAGE - The larges and smallest sample mean observed.

NO.OBS - The number of runs.

2-0 EXAMINE STATISTICS COMMANDS

The commands for the subprocess are shown in Table IX-1. They are described briefly below.

2-1. DISPLAY. DISPLAY prints the labels of all system modules and/or operators in the simulation data set. If ADSM's is specified as YES, the labels of all system modules is printed.

If OPERATORS is specified as an ACC, the labels of the operators for that system module type are printed. If OPERATORS is "*", then all operator labels are printed. Finally, a blank response signifies no operator labels are to be printed.

Table IX-1. Examine Statistics Commands.

COMMAND	PROMPTS	RESPONSES	DEFAULTS
a) DISPLAY	ADSM's	No Yes	No
	OPERATORS	blank,*,or an ACC	blank
b) PRINT	none	none	none
c) SHOW	none	none	none
d) USE	SIMULATION-DATA-SET	positive integer	The current simulation data set
	TACTICAL-SCENARIO	positive integer	The current tactical scenario
	RUN	positive integer or "SUMMARY"	SUMMARY

2-2. PRINT. Print displays statitics. It has no prompts. Use of the PRINT command is discussed in Section 3-0 below.

2-3. SHOW. The SHOW command prints the current simulation data set, tactical scenario and run. See USE below.

2-4. USE. The MOPADS User must specify which set of statistics he/she desires to examine. This is done by specifying a simulation data set, a tactical scenario, and a run (or SUMMARY). The USE command accomplishes this. The prompts for USE are updated each time they are given. This means that they remember their last value. Thus, the User can change runs by specifying only the new run number, e.g., USE, RUN=2.

3-0 ORGANIZATION OF THE PRINT COMMAND

3-1. The Main Category Menu.

The PRINT command provides a flexible scheme for selectively printing statistics. For example, all statistics for a particular system module can be printed, or the Operator Task Fraction statistics for all IHAWK Tactical Control Officers (TCO's) can be selected. This is accomplished by selecting a main and a secondary statistics category. The main category menu is shown below.

```
MAIN CATEGORY MENU                          :
   -  - EXIT PRINT COMMAND
   1  - AIR DEFENSE UNIT
   2  - OPERATORS
   3  - CONTROL SYSTEM
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   F  - MOPADS TRACE
   *  - TOGGLE DISPLAY DEVICE(T)
```

The option "*" determines where statistics are printed. Another menu appears that permits the user to select from T-terminal, P-printer, or B-both. Option 3, CONTROL SYSTEM, refers to the control system module that is implicitly present in every simulation data set. A small MSAINT network forms this module which manages the flight paths of aircraft.

3-2. Secondary Category Menus.

When a main category is selected, a secondary category menu is printed. As an example, if main category 1 - AIR DEFENSE UNIT is selected, the following secondary menu appears.

```
MAIN CATEGORY SELECTION:AIR DEFENSE UNIT
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   *  - TOGGLE DISPLAY DEVICE(B)

ENTER SELECTION( ? FOR MENU)
D

ENTER LABEL OF AIR DEFENSE UNIT(? FOR MENU)
 E.G. G1Q2H3 OR Q (FOR ALL TYPE Q'S)
W1
```

The user selected D - MOPADS COUNT statistics. He is now prompted for an air defense unit specification. The label of one air defense unit can be given (e.g., G1Q2H3) in which case the count statistics for that unit will be printed. If an ACC is given (e.g., Q), then the count statistics for all type "Q" system modules in the simulation data set will be printed.

After printing statistics, the secondary menu will be printed again for another selection. The user can return to the main menu with option "0" or exit the PRINT command with option "-".

4-0 THE MOPADS TRACE

No trace has been implemented in the User Interface. A main category selection is available for TRACE, and if it is selected a secondary menu will be printed. If a secondary selection is made, a message will be printed indicating that this feature is not available yet. It has been included for future growth.

An event trace is available, however. The runs to be traced are specified in the Tactical Scenario by the User, and a trace is produced on the MOPADS trace file. (TRACE.DAT for DEC VAX implementations.) An exerpt of the trace is shown below.

```
*****DETAILED ITERATION OUTPUT FOR RUN NUMBER    1 *****

OPERATOR TYPE  MODULE ID  MODULE TYPE  TASK NO.  TASK LABEL  TASK POINT  SIMULATION TIME
-------------  ---------  -----------  --------  ----------  ----------  --------------
           1       3001            3        31  CREATETD    RELEASE     0.000000E+00
           2       3001            3        32  CREATTDA    RELEASE     0.000000E+00
           3       4001            4        45  SOURCTCO    RELEASE     0.000000E+00
           4       4001            4        46  SOURCTCA    RELEASE     0.000000E+00
           5       4001            4        47  SOURCASO    RELEASE     0.000000E+00
           6       4001            4        48  SOURCFCA    RELEASE     0.000000E+00
           7       4001            4        49  SOURCFCB    RELEASE     0.000000E+00
           0          0            1         1  STARTTR     RELEASE     0.000000E+00
           0          0            1         5  CNTRLTIM    RELEASE     0.000000E+00
           0          0            1         1  STARTTR     COMPLETE    0.000000E+00
           0          0            1         2  INITIATE    RELEASE     0.000000E+00
           7       4001            4        49  SOURCFCB    COMPLETE    0.000000E+00
           7       4001            4        40  TASKSEQW    RELEASE     0.000000E+00
           6       4001            4        48  SOURCFCA    COMPLETE    0.000000E+00
           6       4001            4        40  TASKSEQW    RELEASE     0.000000E+00
           5       4001            4        47  SOURCASO    COMPLETE    0.000000E+00
           5       4001            4        40  TASKSEQW    RELEASE     0.000000E+00
           4       4001            4        46  SOURCTCA    COMPLETE    0.000000E+00
           4       4001            4        40  TASKSEQW    RELEASE     0.000000E+00
           3       4001            4        45  SOURCTCO    COMPLETE    0.000000E+00
           3       4001            4        40  TASKSEQW    RELEASE     0.000000E+00
           2       3001            3        32  CREATTDA    COMPLETE    0.000000E+00
           2       3001            3        30  TASKSEQ     RELEASE     0.000000E+00
           1       3001            3        31  CREATETD    COMPLETE    0.000000E+00
           1       3001            3        30  TASKSEQ     RELEASE     0.000000E+00
           1       3001            3        30  TASKSEQ     COMPLETE    0.000000E+00
           1       3001            3       191  TSEVTD      RELEASE     0.000000E+00
```

The headings have the following meanings.

OPERATOR TYPE - Operator type as assigned by the MOPADS Modules when the reference system modules were created. For the current implementation the values indicate:

0 - control system module event

1 - Battalion AN/TSQ-73 Tactical Director

2 - Battalion AN/TSQ-73 Tactical Director Assistant

3 - IHAWK Tactical Control Officer

4 - IHAWK Tactical Control Assistant

5 - IHAWK Azimuth Speed Operator

6 - IHAWK Fire Control Operator A

7 - IHAWK Fire Control Operator B

8 - Group AN/TSQ-73 Tactical Director 1

9 - Group AN/TSQ-73 Tactical Director 2

MODULE ID - The unique number assigned to a system module in the simulation data set. The ACN of the component. See Section VII, 2-5.

MODULE TYPE - The type of component:

1 - Control System Module

2 - Group AN/TSQ-73

3 - Battalion AN/TSQ-73

4 - IHAWK

TASK NO. - Task node number.

TASK LABEL - Label of the task node.

TASK POINT - One of RELEASE, START, COMPLETE, or CLEAR. If RELEASE and START time are the same, only RELEASE will be printed.

SIMULATION TIME - The simulated time in minutes. If in the tactical scenario, the user specified a start time of 0915, then the simulation time will begin at 555. (9x60+15).

A typical MOPADS run processes a great many events, so the trace can be very long. MOPADS will execute faster and consume less disk file space if a trace is not produced.

5-0 AN ANNOTATED EXAMPLE

The following is an annotated example of a terminal session with the EXAMINE STATISTICS subprocess.

(1) Select the EXAMINE STATISTICS subprocess from the User Interface main menu.

(2) The User Interface prints the current simulation data set, tactical scenario, or run to remind the user that they have not been specified.

(3) The USE command selects simulation data set 2, tactical scenario 1, and run 1. This is confirmed with the SHOW command.

(4) The display command will print system module and operator labels.

(5) The control system module is always present with label CONTROL. Two other system modules were also in this simulation: a Group AN/TSQ-73 with label Q1, and an IHAWK with label Q1W1.

(6) The labels of the operators in the system module types are printed.

(7) Select main category of Air Defense unit.

(8) The above selection results in this secondary category menu.

(9) Select Task Node Statistics.

(10) Here we select the system modules. Since W is specified, the Task Node Statistics for all IHAWK system modules will be printed. If only one system module is desired, give its unique label.

(11) There is one statistics node in this IHAWK network. It was activated 142 times.

(12) Request the histogram. The histogram is printed showing how the 142 observations are distributed. Histogram parameters are specified on SAINT network data cards.

(13) Select resource statistics. Since these statistics are summary statistics and we selected run 1 earlier ((3)), this is an inconsistent request. MOPADS prints a message to this effect.

(14) Return to the main category menu and select Operators.

(15) The secondary category menu contains only Task Fraction Statistics. Select all operators from all IHAWK's.

(16) Task Fraction Statistics are printed.

(17) Select SAINT User Statistics from the main menu.

(18) Select Air Defense Unit and specify the AN/TSQ-73. Note that CONTROL can be specified for the air defense unit label. The effect is the same as if CONTROL SYSTEM were selected from this menu. CONTROL can be specified any time the user is asked for an air defense system label.

(19) (20) (21) All three types of user statistics are printed. Histograms are not printed, however, unless specifically requested.

(22) Select Control System from the main menu and Count Statistics from the secondary menu.

(23) Count statistics are printed for the Control System Module.

(24) Toggle the display device so output is printed at both the terminal and on the MOPADS line printer file. The main category menu shows the current display option.

(25) Select summary statistics with the USE command. Confirm with SHOW.

(26) Re-enter PRINT and select resource statistics.

(27) Select IHAWK system module from the secondary menu.

(28) Resource statistics are printed.

(29) Select Task Node statistics from the main menu and Q1W1 from the secondary menu.

(30) The summary statistics for the one statistics node are printed. Note that this is the same node seen in (11). The average from the 142 observations at (11) was 0.9921. This value is recorded as an observation of summary statistics. Since only one run was performed, the summary statistics have only this one observation.

(31) The PRINT command can be exited from any menu.

6-0 CHANGING THE STATISTICS

Statistics may be added/deleted/or changed with relatively little difficulty. In fact, some statistics can be modified directly by a MOPADS user without help from a MOPADS modeler.

6-1. Task Node Statistics are the easiest to change. These statistics are specified by entries on the data cards for the system module networks. If the MOPADS user is familiar with how these data cards are developed, then a quick reference to the task networks in Goodin & Walker (1983a,b) will provide the information necessary.

Once the desired statistics are specified on the network data cards, they are automatically collected by MSAINT and copied to the User Interface at the end of each run.

6-2. Time Persistent Statistics, Statistics Based on Observation, and User Histograms are easily specified, but they require programming to collect observations. The statistics are specified with data cards as are task node statistics. However, MSAINT does not automatically collect observations. A MOPADS modeler must insert subroutine calls to the appropriate statistics programs in order to record observations, see Walker (1983)

6-3. Count Statistics have no data cards. They are specified by including their labels in a FORTRAN DATA statement in BLOCK DATA program BLOCKY. Observations are recorded by calling SUBROUTINE CSTATY at the appropriate times. See Goodin & Polito (1983d) for descriptions of the subprograms.

6-4. Operator Task Fraction Statistics are collected automatically by MOPADS. Additions to these statistics would occur only if new operator tasks were added to the models. This would be a substantial revision of the modules. Task Fraction Statistics for the new tasks will be collected automatically if the conventions in Goodin & Walker (1983a,b) are followed.

6-5. Resource Statistics are collected automatically for all SAINT resources that are defined. Resources with blank labels are not reported, however. These resources are specified on network data cards.

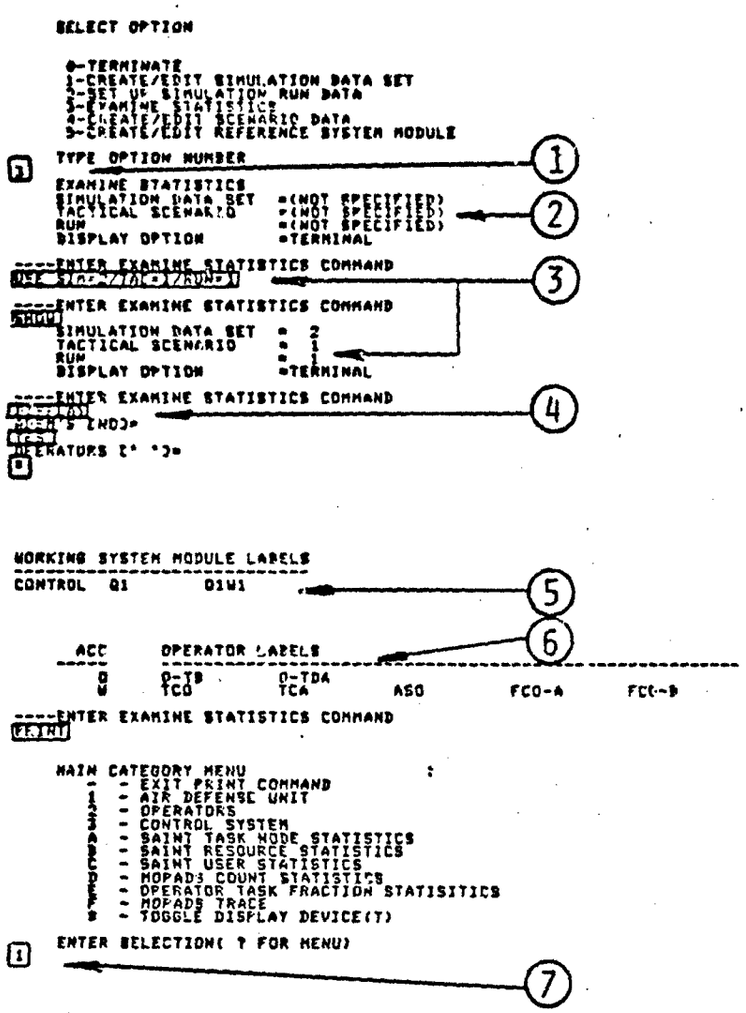


Figure IX-1. Example EXAMINE STATISTICS Commands.

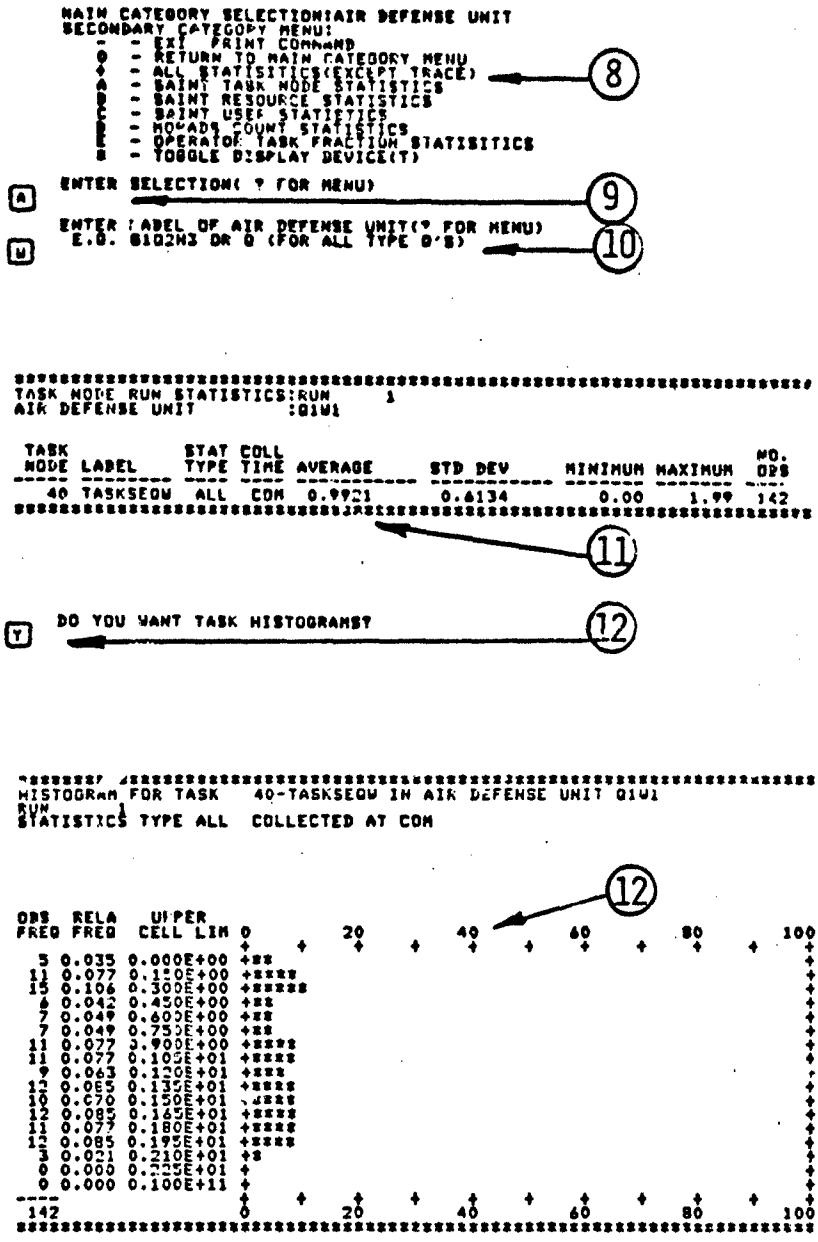


Figure IX-1. (Continued)

```
MAIN CATEGORY SELECTION:AIR DEFENSE UNIT
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISTICS(EXCEPT TRACE)
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   $  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
D

ENTER LABEL OF AIR DEFENSE UNIT(? FOR MENU)
 E.G. 8102H3 OR 0 (FOR ALL TYPE 0'S)
W
RESOURCE STATISTICS ARE SUMMARY ONLY

MAIN CATEGORY SELECTION:AIR DEFENSE UNIT
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS


   $  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
0

MAIN CATEGORY MENU
   -  - EXIT PRINT COMMAND
   1  - AIR DEFENSE UNIT
   2  - OPERATORS
   3  - CONTROL SYSTEM
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   F  - MOPADS TRACE
   $  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
2

MAIN CATEGORY SELECTION:OPERATORS
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   E  - OPERATOR TASK FRACTION STATISITICS
   $  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
E

ENTER LABEL OF AIR DEFENSE UNIT(? FOR MENU)
 E.G. 8102H3 OR 0 (FOR ALL TYPE 0'S)
W

ENTER OPERATOR LABEL(* FOR ALL, ? FOR MENU)
*
```

```
**********************************************************************
OPERATOR TASK FRACTION STATISTICS     :RUN     1
AIR DEFENSE UNIT                      :01W1
OPERATOR                              :TCO

     TASK                               TIME
                                      PERCENT
     -----                            -------
       21                             31.003
       22                              8.563
       23                              9.554
       24                             19.464
       27                              8.202
       28                              3.248
       29                             17.713
       31                              1.650
       32                              0.600
**********************************************************************
```

Figure IX-1. (Continued)

```
OPERATOR TASK FRACTION STATISTICS     !RUN     1
AIR DEFENSE UNIT                     !Q1W1
OPERATOR                             !FCO-B

     TASK                              TIME
                                     PERCENT
     -----                           -------
        7                             48.717
        8                              6.828
        9                              0.910
       11                              2.270
       12                              0.954
       13                              1.501
       14                             15.931
       16                              3.489
```

```
    MAIN CATEGORY SELECTION:OPERATORS
    SECONDARY CATEGORY MENU:
       -  - EXIT PRINT COMMAND
       0  - RETURN TO MAIN CATEGORY MENU
       E  - OPERATOR TASK FRACTION STATISITICS
       S  - TOGGLE DISPLAY DEVICE(T)

    ENTER SELECTION( ? FOR MENU)
0

    MAIN CATEGORY MENU                    :
       -  - EXIT PRINT COMMAND
       1  - AIR DEFENSE UNIT
       4  - OPERATORS
       5  - CONTROL SYSTEM
       A  - SAINT TASK NODE STATISTICS
       B  - SAINT RESOURCE STATISTICS
       C  - SAINT USER STATISTICS
       D  - MOPADS COUNT STATISTICS
       E  - OPERATOR TASK FRACTION STATISITICS
       F  - MOPADS TRACE
       S  - TOGGLE DISPLAY DEVICE(T)

    ENTER SELECTION( ? FOR MENU)                    (17)
C

    MAIN CATEGORY SELECTION:SAINT USER STATISITICS
    SECONDARY CATEGORY MENU:
       -  - EXIT PRINT COMMAND
       0  - RETURN TO MAIN CATEGORY MENU
       +  - ALL STATISITICS(EXCEPT TRACE)
       1  - AIR DEFENSE UNIT
       5  - CONTROL SYSTEM
       S  - TOGGLE DISPLAY DEVICE(T)

    ENTER SELECTION( ? FOR MENU)                    (18)
1

    ENTER LABEL OF AIR DEFENSE UNIT(? FOR MENU)
      E.G. G1Q2H3 OR 0 (FOR ALL TYPE 0'S)
Q1W1
```

```
OPERATOR TASK FRACTION STA"ISTICS     !RUN     1
AIR DEFENSE UNIT                      !Q1W1
OPERATOR                              !TCA

     TASK                              TIME
                                     PERCENT
     -----                           -------
       17                             59.893
       18                              9.037
       19                             19.961
       20                             11.108
```

Figure IX-1. (Continued)

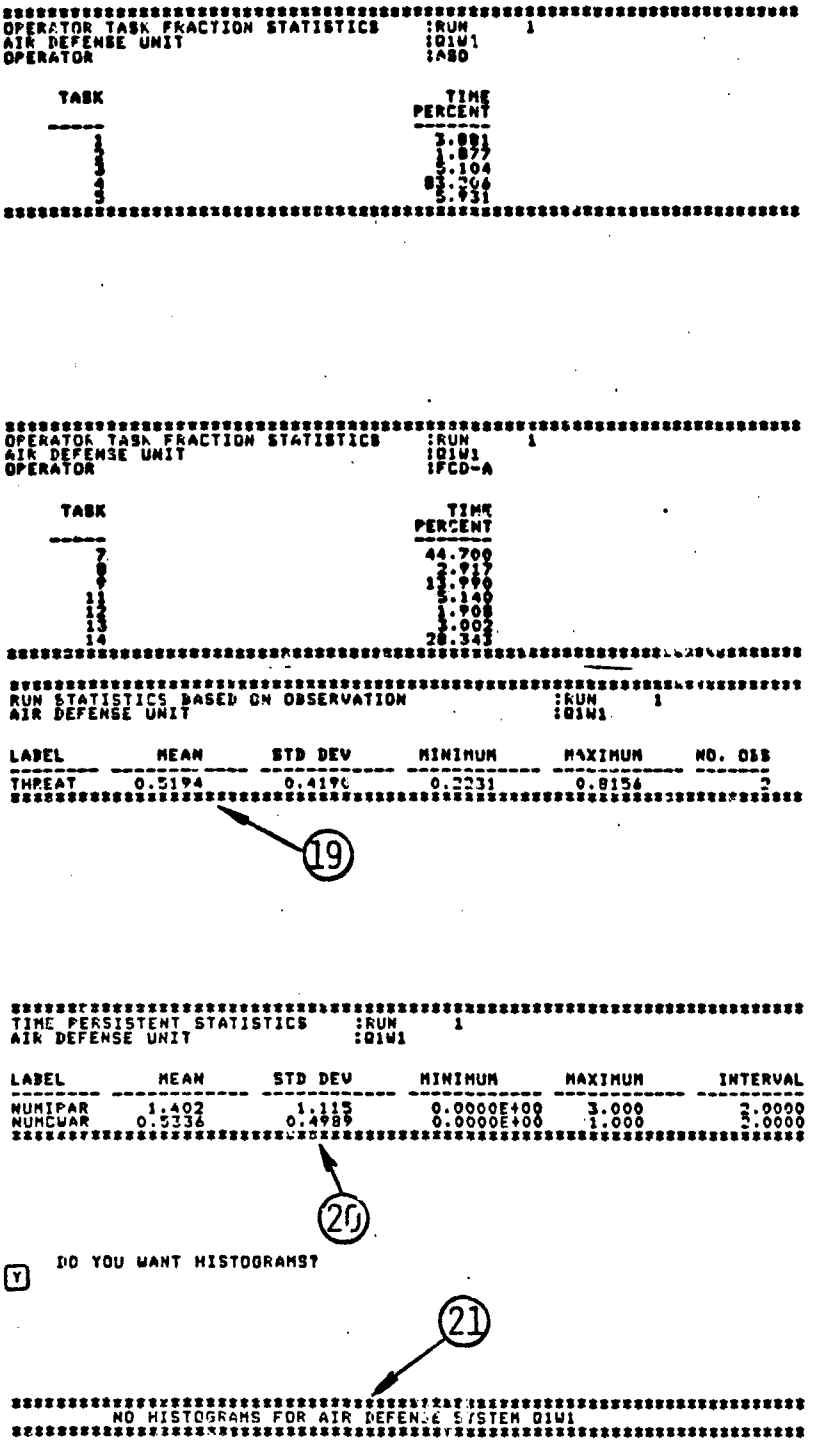


Figure IX-1. (Continued)

```
MAIN CATEGORY SELECTION:SAINT USER STATISITICS
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   1  - AIR DEFENSE UNIT
   3  - CONTROL SYSTEM
   F  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
```

0

```
MAIN CATEGORY MENU                    :
   -  - EXIT PRINT COMMAND
   1  - AIR DEFENSE UNIT
   2  - OPERATORS
   3  - CONTROL SYSTEM
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   F  - MOPADS TRACE
   $  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
```

3 ← (22)

```
MAIN CATEGORY SELECTION:CONTROL SYSTEM
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   $  - TOGGLE DISPLAY DEVICE(T)


ENTER SELECTION( ? FOR MENU)
```

D

```
**********************************************************************
COUNT STATISTICS     :RUN     1
AIR DEFENSE UNIT     :CONTROL

                                 COUNT
                                 --------
NUMBER OF SITES DESTROYED           1
NUMBER OF SITES ATTACKED            1
NUM HOST TRACKS NOT DEST            0   <- (23)
TOTAL NUMB HOSTILE TRACKS           1
TOTAL NUMB FRIEND TRACKS            1
TOTAL NUMB OTHER TRACKS             1
NUMB HOSTILE A/C DEST               1
NUMB FRIENDLY A/C DEST              0
NUMB OTHER A/C DEST                 1
NUMB HOST TRACKS ATTACKED           2
NUMB FRND TRACKS ATTACKED           0
NUMB OTHR TRACKS ATTACKED           1
**********************************************************************
```

```
MAIN CATEGORY SELECTION:CONTROL SYSTEM
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   $  - TOGGLE DISPLAY DEVICE(T)
```

Figure IX-1. (Continued)

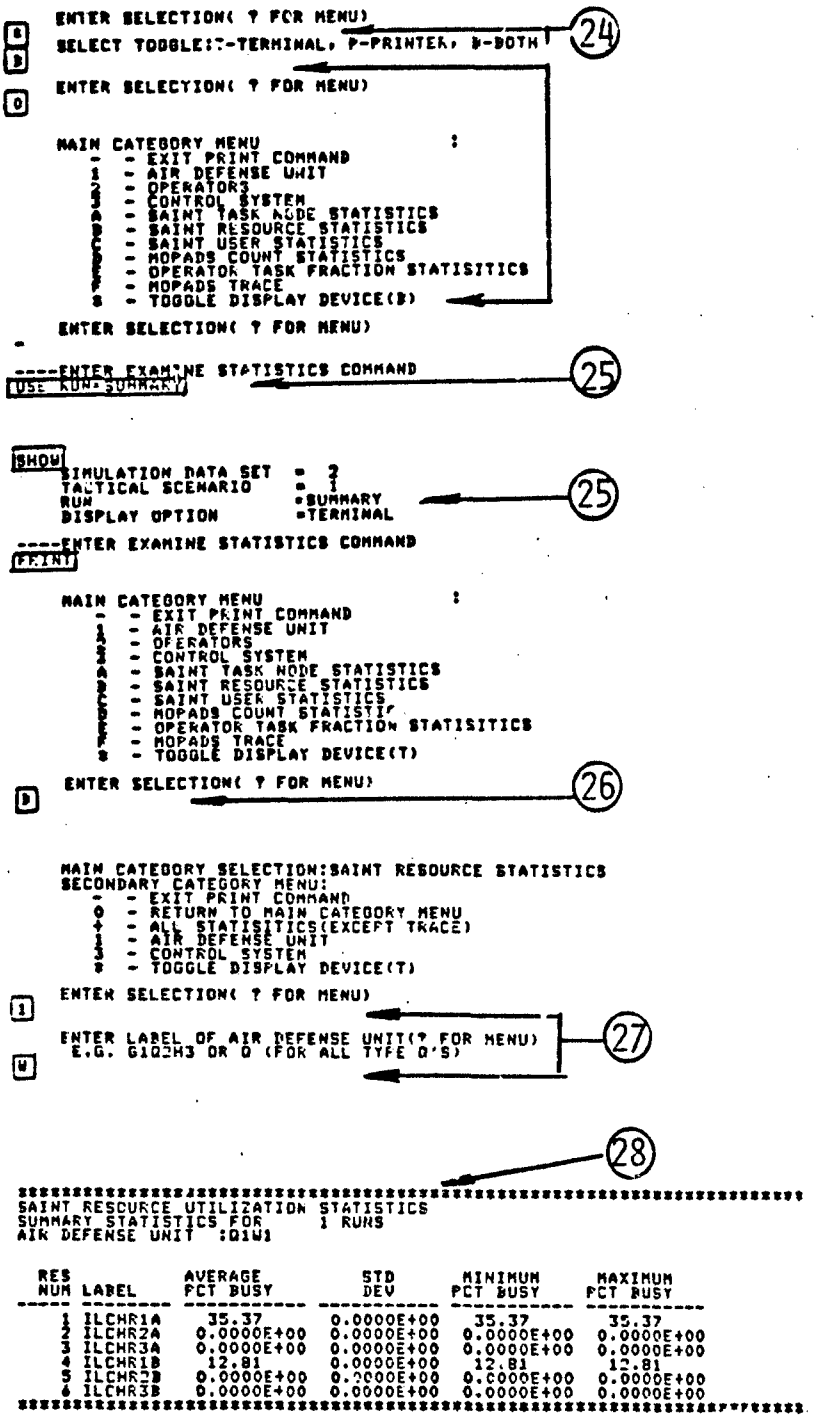


Figure IX-1. (Continued)

```
MAIN CATEGORY SELECTION:SAINT RESOURCE STATISTICS
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   1  - AIR DEFENSE UNIT
   3  - CONTROL SYSTEM
   8  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
0

MAIN CATEGORY MENU                    :
   -  - EXIT PRINT COMMAND
   1  - AIR DEFENSE UNIT
   2  - OPERATORS
   3  - CONTROL SYSTEM
   A  - SAINT TASK NODE STATISTICS
   B  - SAINT RESOURCE STATISTICS
   C  - SAINT USER STATISTICS
   D  - MOPADS COUNT STATISTICS
   E  - OPERATOR TASK FRACTION STATISITICS
   F  - MOPADS TRACE
   8  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
A                                                    29

MAIN CATEGORY SELECTION:SAINT TASK NODE STATISTICS



SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   1  - AIR DEFENSE UNIT
   3  - CONTROL SYSTEM
   8  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)
1

ENTER LABEL OF AIR DEFENSE UNIT(? FOR MENU)
E.G. G102H3 OR 0 (FOR ALL TYPE 0'S)
01W1
                                                     30
**********************************************************************
TASK NODE SUMMARY STATISTICS      :    1 RUNS
AIR DEFENSE UNIT          :01W1

 TASK                STAT COLL AVERAGE OF  STD DEV OF  MINIMUM MAXIMUM  NO.
 NODE LABEL          TYPE TIME AVERAGE     AVERAGES    AVERAGE AVERAGE  OBS
----- ------------- ---- ---- ----------- ------------ ------- ------- ----
   40 TASK:SEQW     ALL  COM  0.9921      0.0000E+00     0.99    0.99     1
**********************************************************************



MAIN CATEGORY SELECTION:SAINT TASK NODE STATISTICS
SECONDARY CATEGORY MENU:
   -  - EXIT PRINT COMMAND
   0  - RETURN TO MAIN CATEGORY MENU
   +  - ALL STATISITICS(EXCEPT TRACE)
   1  - AIR DEFENSE UNIT
   3  - CONTROL SYSTEM
   8  - TOGGLE DISPLAY DEVICE(T)

ENTER SELECTION( ? FOR MENU)                         31
-

----ENTER EXAMINE STATISTICS COMMAND
```

Figure IX-1. (Continued)

(31) The PRINT command can be exited from any menu.

6-0 CHANGING THE STATISTICS

Statistics may be added/deleted/or changed with relatively little difficulty. In fact, some statistics can be modified directly by a MOPADS user without help from a MOPADS modeler.

6-1. Task Node Statistics are the easiest to change. These statistics are specified by entries on the data cards for the system module networks. If the MOPADS user is familiar with how these data cards are developed, then a quick reference to the task networks in Goodin & Walker (1983a,b) will provide the information necessary.

Once the desired statistics are specified on the network data cards, they are automatically collected by MSAINT and copied to the User Interface at the end of each run.

6-2. Time Persistent Statistics, Statistics Based on Observation, and User Histograms are easily specified, but they require programming to collect observations. The statistics are specified with data cards as are task node statistics. However, MSAINT does not automatically collect observations. A MOPADS modeler must insert subroutine calls to the appropriate statistics programs in order to record observations, see Walker (1983)

6-3. Count Statistics have no data cards. They are specified by including their labels in a FORTRAN DATA statement in BLOCK DATA program BLOCKY. Observations are recorded by calling SUBROUTINE CSTATY at the appropriate times. See Goodin & Polito (1983d) for descriptions of the subprograms.

6-4. Operator Task Fraction Statistics are collected automatically by MOPADS. Additions to these statistics would occur only if new operator tasks were added to the models. This would be a substantial revision of the modules. Task Fraction Statistics for the new tasks will be collected automatically if the conventions in Goodin & Walker (1983a,b) are followed.

6-5. Resource Statistics are collected automatically for all SAINT resources that are defined. Resources with blank labels are not reported, however. These resources are specified on network data cards.

X. REFERENCES

Goodin II, J. R. & Polito, J. User guide for the AN/TSQ-73 system module (MOPADS Vol. 3.1). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (a)

Goodin II, J. R. & Polito, J. User's guide for the IHAWK system module (MOPADS Vol. 3.2). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (b)

Goodin II, J. R. & Polito, J. MOPADS free-format syntax processor (MOPADS FFSP) (MOPADS Vol. 5.11). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Goodin II, J. R. & Polito, J. Documentation manual for the MOPADS control module (MOPADS/CNTRL) and the MOPADS common system module programs (MOPADS/CSMP) (MOPADS Vol. 519). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (d)

Goodin II, J. R. & Walker, J. L. Documentation manual for the AN/TSQ-73 system module (MOPADS Vol. 5.15). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (a)

Goodin II, J. R. & Walker, J. L. MOPADS documentation manual for the IHAWK system module (MOPADS Vol. 5.16). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (b)

Laughery, K. R. HOMO establishment of performance criteria for non-decision making tasks (MOPADS Vol. 5.6). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, 1983.

Laughery, K. R. and Gawron, Calspan Corporation, 1983.

Polito, J. Creating reference air defense system modules (MOPADS Vol. 4.6). Pritsker & Associates, Inc., prepared for the U.S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983. (a)

REFERENCES
(continued)

Polito, J. Creating MOPADS air scenarios (MOPADS Vol. 4.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. MOPADS architecture manual (MOPADS Vol. 4.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c).

Polito, J. MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d).

Polito, J. MOPADS task sequencing structure (MOPADS Vol. 5.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (e).

Polito, J. & Laughery, K. R. MOPADS final report (MOPADS Vol. 1.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Walker, J. L. MSAINT user's guide: changes and additions to the SAINT user's manual (MOPADS Vol. 4.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

XI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ATC-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1F-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

XII. CHANGE NOTICES

APPENDIX D

MOPADS FINAL REPORT:

MOPADS ARCHITECTURE MANUAL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

1. INTRODUCTION

1-0 THE WORLD VIEW

Every modeling system must settle on a perspective or "world view" upon which to construct representations of the systems under study. The world view is important because it determines the modeling methodology and the degree of detail which will be represented.

As an example, consider building models of nuclear fission processes. At the atomic level, nuclear fission consists of collisions of subatomic particles which split atomic nuclei releasing energy and spawning other particles which will go on to collide with still other atoms. The rate of this reaction is determined by the density of appropriate atoms and the presence of moderating materials which absorb subatomic particles without spawning new ones.

This process could be modeled by representing individual nuclear and subnuclear particles with a knowledge of the statistical characterizations of appropriate variables. The world view represented by such a model would be as just described.

An alternative world view is to consider the fission process in the aggregate as a thermal process whose rates can be characterized by shape, density, and orientation of the components. The mathematical model for this case is usually represented with ordinary and/or partial differential equations. The implicit assumption in such models is that the individual events associated with a single subatomic particle are insignificant to the aggregate thermal performance.

There is no "correct" world view for a particular system. The world view must be chosen with an eye to the objectives of the model. In the case of MOPADS, the problem of choosing a world view is multifaceted because of the many aspects of the systems being represented. The focus of MOPADS is clearly an operator performance, but the operators are embedded in a system involving communication (command and control) and exogenous activities (the air battle). The way and degree to which each of these aspects of the overall system is represented are discussed in Section II.

2-0 SOFTWARE ARCHITECTURE

A second purpose of this document is to explain the architecture of the MOPADS software. The software structure is discussed in considerable detail because of the size and complexity of the software system. Anyone modifying the software will need such a detailed road map in order to confidently proceed.

MOPADS is functionally hierarchical in that functions are separated in a tree structure from the main program. Also, in many cases, the subprogram calling sequences and structure have been standardized so that new air defense system modules can be implemented by emulating (mutatis mutandis) one of the existing system modules. Section III describes the MOPADS software structure in detail.

II. THE MOPADS WORLD VIEW

1-0 OPERATORS

The most fundamental assumption of MOPADS is that operators have a finite capability to perform their duties. The workload, their individual characteristics, and the external environment all affect their ability to perform their duties. Capturing the effects of these variables on operators' performance is the essential goal of MOPADS.

In response to the external stimuli provided by the air battle, operators perform the tasks (sometimes called "critical tasks") prescribed in the system manuals for their equipment. Each of these tasks is usually composed of several actions called "task elements" in the MOPADS terminology. Task elements are elemental in the MOPADS context because they represent the lowest level of operator activity that is modeled. Figure II-1 is an example of an operator task for the AN/TSQ-73 (U. S. Army Technical Manual, TM9-1430-652-10-3).

Tasks consist of individual actions (shown in trapezoids, e.g., ①), simple decisions (shown in diamonds, e.g., ②), and perhaps more complex activities that are themselves complete tasks, e.g., ③. Task elements are the individual actions (i.e., trapezoids in Figure II-1). It is, of course, possible to consider the task element as complex physiological activities. For example, the activity labeled ① in Figure II-1 is to press a particular button on a console. This activity requires memory accessess to recall the general location of the button, muscular activity in the head and eyes to visually acquire the correct button, and muscle activity in the arm and hand to physically press the button.

MOPADS does not represent human activity to this level of detail because the literature on human factors does not provide such performance data as a function of parameters which are easily measured in the air defense setting. The approach taken with MOPADS has been to develop a taxonomy of skills which adequately represent the activities of air defense operators (Laughery (1981a,b). The task elements are then described as functions of the required skills. Sufficient data is available to describe how performance of the various skills is effected by parameters measurable in the air defense setting.

This approach has several advantages all of which stem from the following: the lowest level of modeling detail is also the lowest level of operator description in official system documentation. As a result, communication between MOPADS modelers and

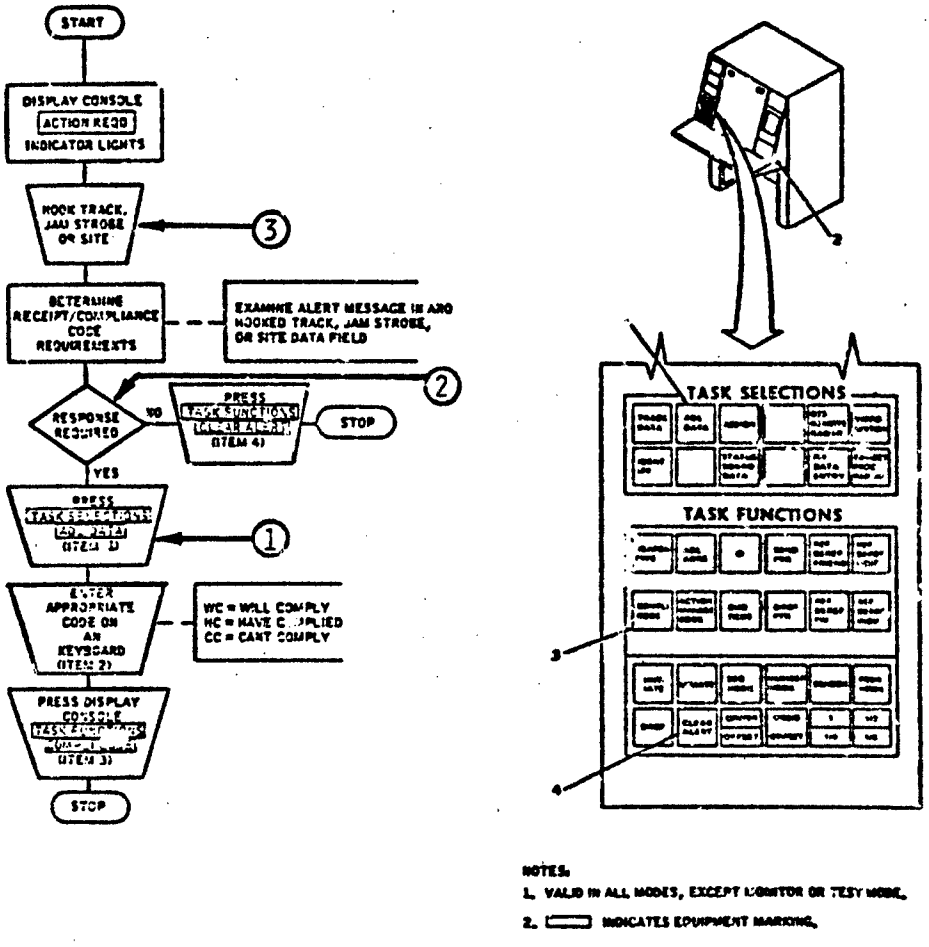


Figure II-1. Example Task for AN/TSQ-73.

subject matter experts is facilitated. Data collection of nominal task element times is reduced to collecting information that can be obtained directly by observing operators, and the lowest level of MOPADS output also corresponds to observable operator actions.

Before performing a task, the operator must select the task from among those that are available. In MOPADS terminology, this is the "task sequencing" activity. When the operator performs a task, he is essentially following a rule-based procedure because the activities that comprise the task are specified in official system documentation. The tasks are step by step procedures which the operators memorize. The selection of which task to perform, however, is knowledge-based behavior in which the operator calls upon his experience to select the task which will most directly lead to mission accomplishment. In most cases, some of the general knowledge base has been reduced to procedures in the form of Tactical Standard Operating Procedures.

Task sequencing is accomplished in MOPADS by representing the operators as goal seekers. Each operator has a set of goals whose levels of dissatisfaction can be evaluated and normalized to a fixed scale. The values of the goals on this fixed scale are used to rank the various goals according to their importance. The next task is selected based on estimates of how each available task will affect the goal dissatisfaction levels.

Various objective functions are avilable in MOPADS for the goal seekers. The simplest is a mini-max strategy in which the operator strives to achieve the greatest reduction in the most dissatisfied goal (i.e., "put out the biggest fire first"). The most complex is the case in which the operator attemps to obtain the largest reduction per unit time of the average of all goal dissatisfaction levels. See Polito (1983a) for more information.

2-0 AIR DEFENSE SYSTEMS

Air defense systems are components of air defense which include operators and equipment. Examples are the AN/TSQ-73 and the IHAWK. The operators of air defense systems are the foci of MOPADS models. In general, the operator tasks which are modeled as described in Section 1-0 above are taken from official documentation describing the air defense system. The models of the operator actions comprise most of the MOPADS representation of the air defense system, but there are aspects of the air defense system that are not directly related to the operator models. For example, there are crew actions associated with set-up, take-down, transportation, and maintenance that are not represented in MOPADS. MOPADS is concerned with operator performance in combat during an air battle, and only those operator tasks that are pertinent to this objective are modeled.

In addition, there are non-operator details of the air defense system that must be represented in order to construct a coherent model. These are discussed below.

2-1. The Environment.

Certain environmental variables affect human performance such as temperature, noise level, illumination, vibration, etc. Also, various system status parameters such as weapons control status and method of control may affect operator actions. These parameters are stored in an "environmental state vector" for each component of the air defense configuration. These environmental variables affect all of the operators serving in a particular air defense componant.

2-2. Field of View.

Knowledge of the air battle is obtained by the air defense configuration through radars and visual observations. Each component of an air defense configuration may have "viewers" which acquire information about the air battle. In most cases, these are radars, but in the case of systems such as Redeye, the view is the visual field of the operator.

Only aircraft that are in the field of view of an air defense component are known to the air defense configuration. If the ATDL is functioning, information on known aircraft is shared among components, however.

Zero or more viewers may be defined for each component. The field of view of each viewer can be defined in a reasonably flexible way that permits such things as ground obstructions and restricted sweep angles to be represented. A status for each viewer is maintained, so viewers may be modeled as becoming inoperative during the course of a simualtion.

2-3. Display Information.

Each air defense system will present different information to the operators which they must process and which will, in part, determine their performance. The MOPADS software provides mechanisms for maintaining operator display information. The MOPADS modeler must determine which subset of the information available to the operator is to be represented and select the appropriate definitions for MOPADS display parameters. In other words, the MOPADS data base software provides mechanisms to store and retrieve "display" information, but it does not "know" the definitions of the individual display variables. These definitions are determined by the MOPADS modeler when the air defense system module is developed and are manifested in the simulation through the SAINT user code written to implement the system module.

2-4. Hardware Resources.

The air defense system may have various subcomponents which are necessary for one or more of the operators to perform his tasks and which are subject to failure or in other ways become

unavailable. Such subcomponents represent "hardware resources" that the operators require in order to perform their functions. The MOPADS software contains mechanisms to define and maintain information about such resources. Hardware resources are similar to display information in that MOPADS simply provides a means to define, store, and access such information. The MOPADS modeler must implement the impact of such resources on the system when he developes an air defense system module. In other words, he must: 1) model the mechanisms by which the resources become available and unavailable, 2) model the contingent procedures the operators will use if a required resource is unavailable, and 3) maintain any desired statistical information resulting from the hardware resources.

3-0 COMMAND, CONTROL, AND COMMUNICATIONS (C^3)

An important aspect of an air defense configuration is the coordination that takes place between air defense components. Individual units do not normally work in isolation. The C^3 function must be represented in MOPADS in order to obtain a reasonable representation of an air defense configuration. C^3 is not, however, the primary focus of MOPADS, and no attempt has been made to develop a complex, high fidelity C^3 model.

In practice, the C^3 functions are implemented by communications sent from one air defense component to one or more other components. This process has been mirrored in the MOPADS software. The only interaction between system modules in a MOPADS simulation is through "messages" sent from one to another. All C^3 functions are represented as messages.

The type, content, and format of messages may be determined by the MOPADS modeler when a system module is developed. Thus, messages that represent commands, status updates, acknowledgements, etc. can be created and transmitted with the MOPADS message handling system. This system has several advantages: 1) the C^3 representation in MOPADS can be enhanced at a later date by adding and modifying software modules without major structural changes to the existing software, 2) the specific command and control configuation can be specified by the MOPADS modeler at run time, thus allowing different configuations to be simulated without re-programming, and 3) new air defense system modules can be developed and added to the MOPADS system with very little modification of existing system modules.

The main restriction on (3) above is that if the new system module sends or expects to receive a new message type, the already existing system modules must be modified to send/receive and process those messages.

To summarize, MOPADS assumes that the C^3 function is embedded in the operator tasks, and that, when required, operators will generate C^3 messages to be transmitted to other system components. The system modules must be developed in such a way that the operators respond properly to C^3 messages which they receive.

Finally, certain procedural aspects of C^3 for air defense systems are classified. Since MOPADS is an unclassified modeling system, generic or simplified representations of C^3 functions have been built into the MOPADS models. As stated above, the models could be modified without undue effort to include a more rich C^3 representation, but the current models are consistent with a focus on operator performance.

4-0 TACTICAL SCENARIOS

Tactical scenarios consist of an air scenario and the physical configuration of air defense components. In addition, protected sites or "critical assets" may be specified which the air defense configuration will protect.

The MOPADS modeler develops scenarios by selecting a reference position (latitude, longitude, and altitude). All other locations are specified by a rectangular coordinate system centered at the reference point, X and Y coordinates are in units of nautical miles. The positive x direction is east and positive y values are north. The Z coordinate is in feet. Positive z values are up.

Protected sites are specified by position, label, and site type. The site type is defined by the MOPADS modeler. The MOPADS software currently makes no use of the site types. The protected sites are not active participants in the simulation. In other words, there is no model of activities at these locations. The simulated air defense system will simply attempt to attack hostile aircraft that approach these locations.

The location of tactical scenario components must also be specified by the MOPADS user. The air defense configuration will automatically attempt to protect its own components. Thus the air defense configuration will attempt to protect itself and any other sites designated as critical assets. Hostile aircraft will be attacked even if they are not near a protected site; however, priority is given to attacking hostile aircraft that are approaching a protected site.

New scenarios are created by specifying the flight paths of aircraft. Aircraft may be hostile, friendly, or "other." These categories are their true category (i.e., thier primary ID's). The air defense configuration will recognize an aircraft's category only after performing normal IFF procedures.

MOPADS assumes a flat earth, and aircraft flight paths or tracks are specified as piecewise linear segments. In other words, aircraft fly from point to point (called checkpoints), and between checkpoints, the aircraft's direction and speed do not change. This is not particularly restrictive because as many checkpoints as needed may be specified in order to approximate curvilinear motions.

Tracks are specified by their initiation time (from the start or the simulation), primary ID, multiplicity, aircraft type, and initiation point. Each checkpoint is characterized by the coordinates of the checkpoint, speed, whether or not the end point of the segment is a target, and whether or not the aircraft is jamming on the segment. Currently, MOPADS does not use jamming information.

See Polito (1983b,c,d) for more detailed information on how to specify the information described in this section.

III. SOFTWARE ARCHITECTURE

1-0 OVERVIEW

The MOPADS software has been designed and developed in a modular fashion. This means that subprograms that have similar functions are grouped together. They may share an internal data structure and work together to perform various functions. Programs outside the module generally have no knowledge of the data structures or internal workings of the module. External programs make use of the module by subprogram calls, and they communicate with the module only through formal subprogram parameters.

MOPADS is also functionally modular in that the flow of control of the program is hierarchical. This structure will facilitate developing an overlay design for MOPADS if necessary. MOPADS has been developed on a virtual memory operating system and no attempt has been made to develop an overlay design to fit into a specific memory size. The design of the software, however, should lend itself readily to overlaying if needed.

2-0 SOFTWARE MODULES

Table III-1 shows the major MOPADS software modules and the primary reference for each. These programs (with the exception of FFIN2, Polito (1983e)) have been developed following MOPADS standards set up in Walker & Polito (1982). Accordingly, each module was assigned a one character suffix. All subprogram names and all COMMON variables in the module end with the assigned suffix. Also, the programs in each module are documented in the specified MOPADS volume (also in accordance with Walker & Polito (1982)). The exception is FFIN2 which is composed of programs which predate MOPADS. The programs were not modified, so they do not use a uniform suffix or follow other MOPADS programming conventions. The module is self containing, however, and the documentation is of equal quality to the other references.

Also, the SAINT software predates MOPADS, and although modifications were made to SAINT to produce a MOPADS specific variant (MSAINT), no attempt was made to make MSAINT conform to the requirements of Walker & Polito (1982). MSAINT is documented in Walker (1983).

The programs which implement the air defense system modules are contained in the modules with suffixes G, Q, and W and are documented in Goodin & Walker (1983a,b). These documents contain technical details of the software. User instructions for conducting simulations with these modules may be found in Goodin & Polito (1983a,b).

Table III-1. MOPADS Software Modules

MODULE	SUFFIX	REFERENCE
Data Base Application Program (DBAP)	A	5.18
Control System Module	C	5.19
Data Base Control System (DBCS)	D	5.13
AN/TSQ-73 System Module (Group)	G	5.15
Human Factors Moderator Functions	H	5.10
User Interface	I	5.12
Main MOPADS Module	M	5.12
AN/TSQ-73 System Module (Battalion)	Q	5.15
Free-Format Syntax Processor (FFSP)	S	5.11
MOPADS Utilities (UTIL)	U	5.9
IHAWK System Module	W	5.16
Common System Module Programs (including SAINT User Code)	Y	5.19
Free-Format Input (FFIN2)	-	5.14

The Control System module (suffix C) is a special system module that is present in all MOPADS simulations. It manages the MSAINT functions needed to simulate the flight paths of aircraft. It contains control functions to schedule aircraft checkpoints, initiate tracks, terminate tracks, and maintain the data structure used by the other system modules.

The Data Base Control System (D) is the software that maintains the MOPADS data base. Other MOPADS modules (with the exception of the user interface) seldom call this module directly. They access the data base through the Data Base Application Programs (A). These programs contain utilities for frequently required operations with the data base.

The Human Factors Moderator Functions (H) contain all of the moderator function software. This is a stand-alone module that has a uniform and well defined input/output interface. Laughery & Gawron (1983) contains information needed to use this module in non-MOPADS settings.

The User Interface (I) is a large module that implements the interactive software through which the user constructs MOPADS models, specifies simulation data, and examines simulation output. The Main MOPADS module (M) is documented with the user interface. The Main module consists of the main program for MOPADS and a few associated utilities that divert control to the user interface function or to the MSAINT simulation function. The Free Format Syntax Processor (S) is an input utility that uses FFIN2, Polito (1983e), to provide all command driven input for the user interface.

The MOPADS utilities (U) provide housekeeping functions for all other MOPADS modules. Such functions as array copying, file opening/closing, etc. are performed by programs in this module.

Finally, there are some programs that are common across all system modules. The suffix for these programs is Y. MSAINT user codes (e.g., subroutines STATE and UTASK) are contained in this module even though they do not end with the "Y" suffix.

The major structure of the MOPADS software is shown in Figure III-1. The small main module determines whether the current job is a user interface session or a simulation. Control is transferred to the appropriate side of the tree shown in Figure III-1. Control never transfers from one side to the other side of the tree. In other words, each execution of MOPADS must be either an interactive user interface job or a simulation job (which can be run as a batch job).

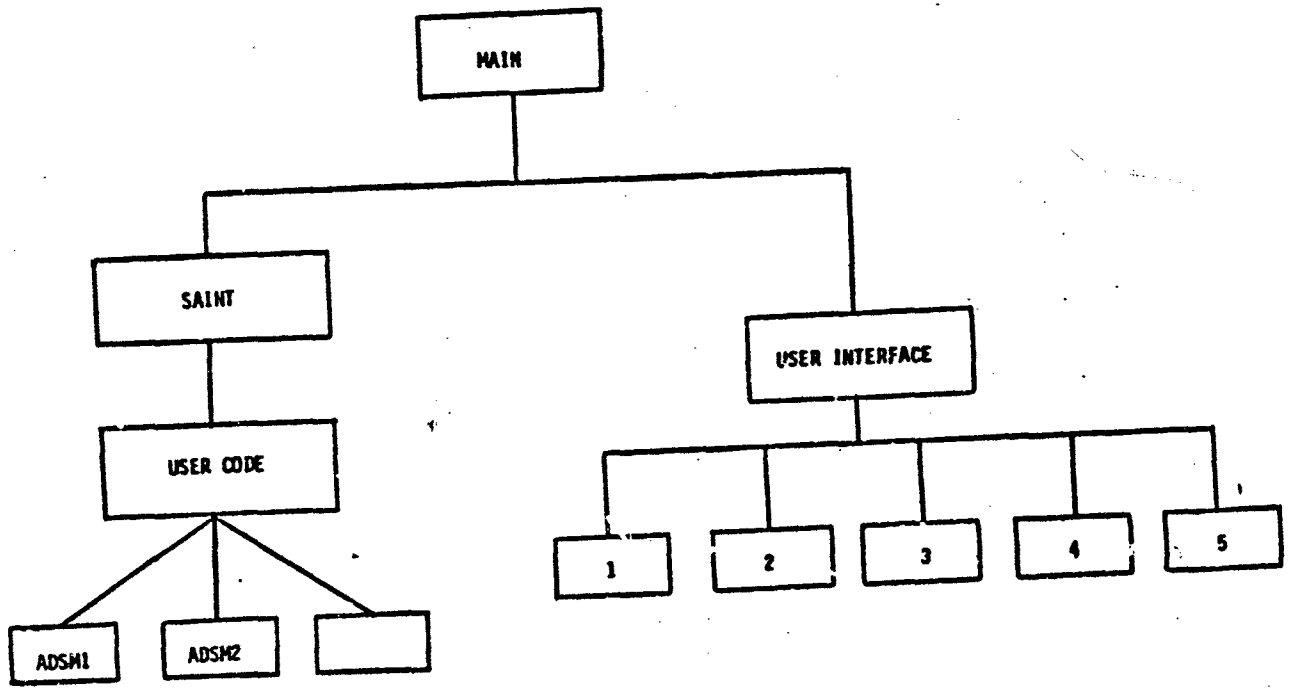


Figure III-1. Major Structure of MOPADS Software.

This structure permits the MOPADS software to be overlaid in a natural way. It also allows MOPADS to be linked as two load modules, if desired. The main module and user interface could be linked as one load module, and the main module with the SAINT side of the tree can be linked as another load module. This strategy may be useful on non-virtual memory computers with limited overlay capabilities and limited memory. It may also be desirable if MOPADS is merged with other programs or the simulation and user interface functions are performed on different computers. The independence of the two sides of the tree structure results from the fact that all communication between the two is through the MOPADS data base.

3-0 MOPADS USER INTERFACE SOFTWARE

The user interface portion of MOPADS is shown in greater detail in Figure III-2. The labels shown in the boxes are actual subprogram names. MOPADM is the name of the main program in MOPADS. For user interface sessions it calls MAINUI which calls CPROCI.

The main function of CPROCI is to determine which user interface subprocess is to be entered. The entry points to the subprocesses have names UI1I to UI5I. The structure below UI3I is typical of all the subprocesses and is not repeated for each case in the figure.

Each of the subprocesses has its own set of commands. In addition, there is a set of basic data base commands that are available in all subprocesses. When a new command is entered by the user, KMNDI is called to determine if it is a basic command or if it is one of the commands specific to the subprocess. If the command is a basic data base command, DBCØI is called which in turn calls the appropriate subprogram to process the command.

If the new command is not one of the basic commands, control is returned from KMNDI to UI3I (for example) which then calls the appropriate subprogram to process the command for that subprocess. Each subprocess has utility programs for performing functions unique to the subprocess's requirements. Furthermore all of the subprocesses make extensive use of the modules DBCS, DBAP, FFSP, FFIN2, and UTIL.

The structure of the user interface subtree obviously can be overlaid, also. CPROCI can be a root of a subtree, since only one subprocess is active at a time. Furthermore, within each subprocess, each command is processed by a separate subprogram, so opportunities exist for further overlays at even this level. The function of the user interface subprocesses are described in Polito (1983b,c,d).

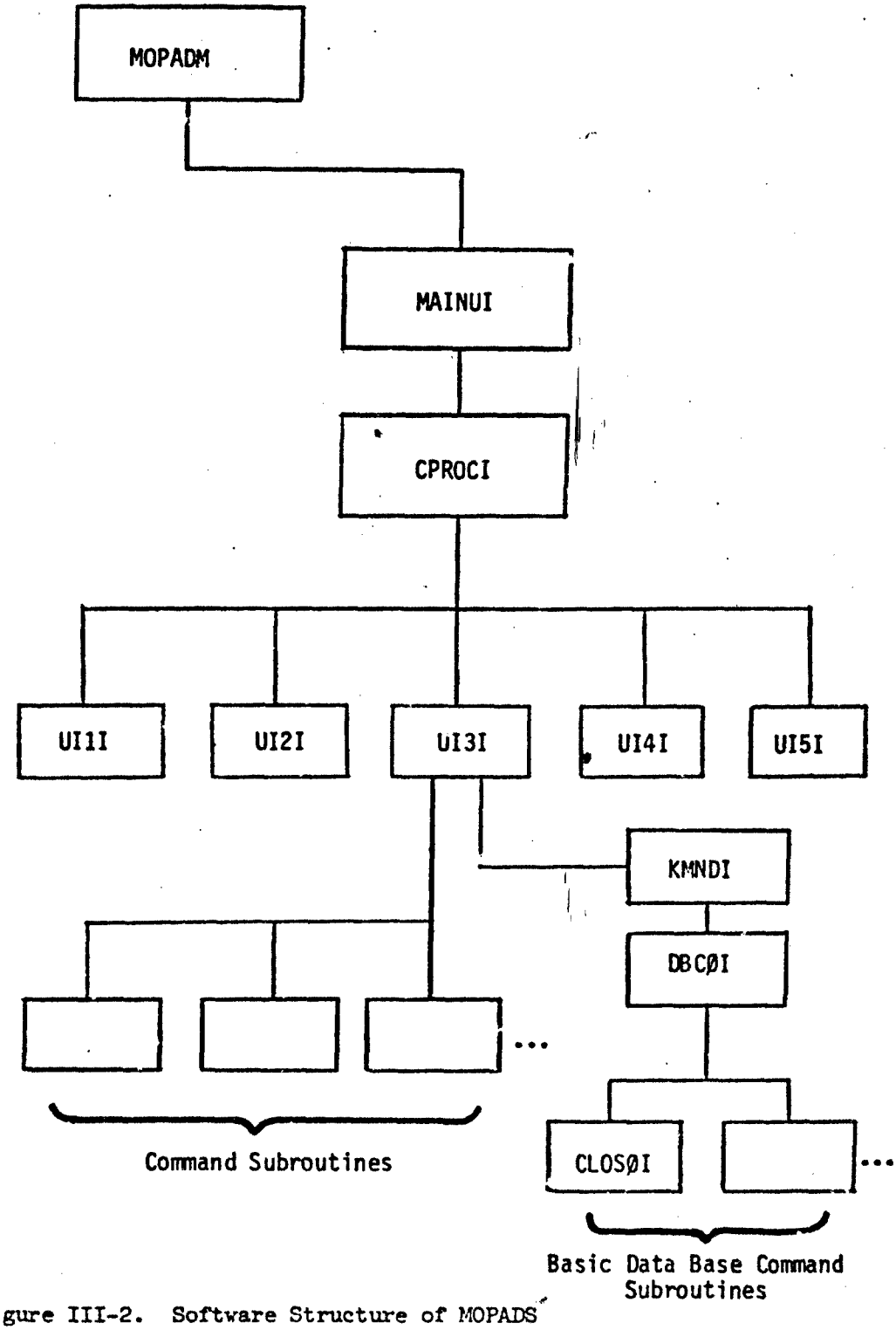


Figure III-2. Software Structure of MOPADS User Interface.

4-0 MOPADS SIMULATION SOFTWARE

4-1. Initialization.

Figure III-3 shows the major software structure for initialization during MOPADS simulation jobs. Processing from MOPADM reads the MOPADS data cards and determines that a simulation (not a user interface) job is in progress. MOPADM then calls BATCHM.

BATCHM continues to interpret MOPADS data cards. It then constructs a single SAINT network data file by concatenating the network data files for each system module that will be present in the simulation. When this is done, BATCHM calls MSAINT. SUBROUTINE MAIN is the entry point to MSAINT. MAIN initializes SAINT variables and calls the main MSAINT programs. MSAINT reads the network data files and calls SUBROUTINE UINPUT.

The function of SUBROUTINE UINPT is to allow the modeler to perform any required initialization of system modules before beginning any simulations. UINPUT calls five programs:

INITY	-	performs initialization of MOPADS data structure that is common to all system modules.
INITC	-	performs initialization of the control system module that manages air scenario information.
INITG	-	performs initialization for the Group AN/TSQ-73 system module.
INITQ	-	performs initialization for the Battalion AN/TSQ-73 system module.
INITW	-	performs initialization for the IHAWK system module.

Initializations performed from UNIPT are one-time actions that need to be done only once. MSAINT also calls SUBROUTINE USTART to perform initialization that must be performed prior to each run. USTART also calls the subprograms listed above. Parameters are passed to INITY, INITC, INITG, INITQ, and INITW so they can distinguish whether the call is from UINPT or USTART.

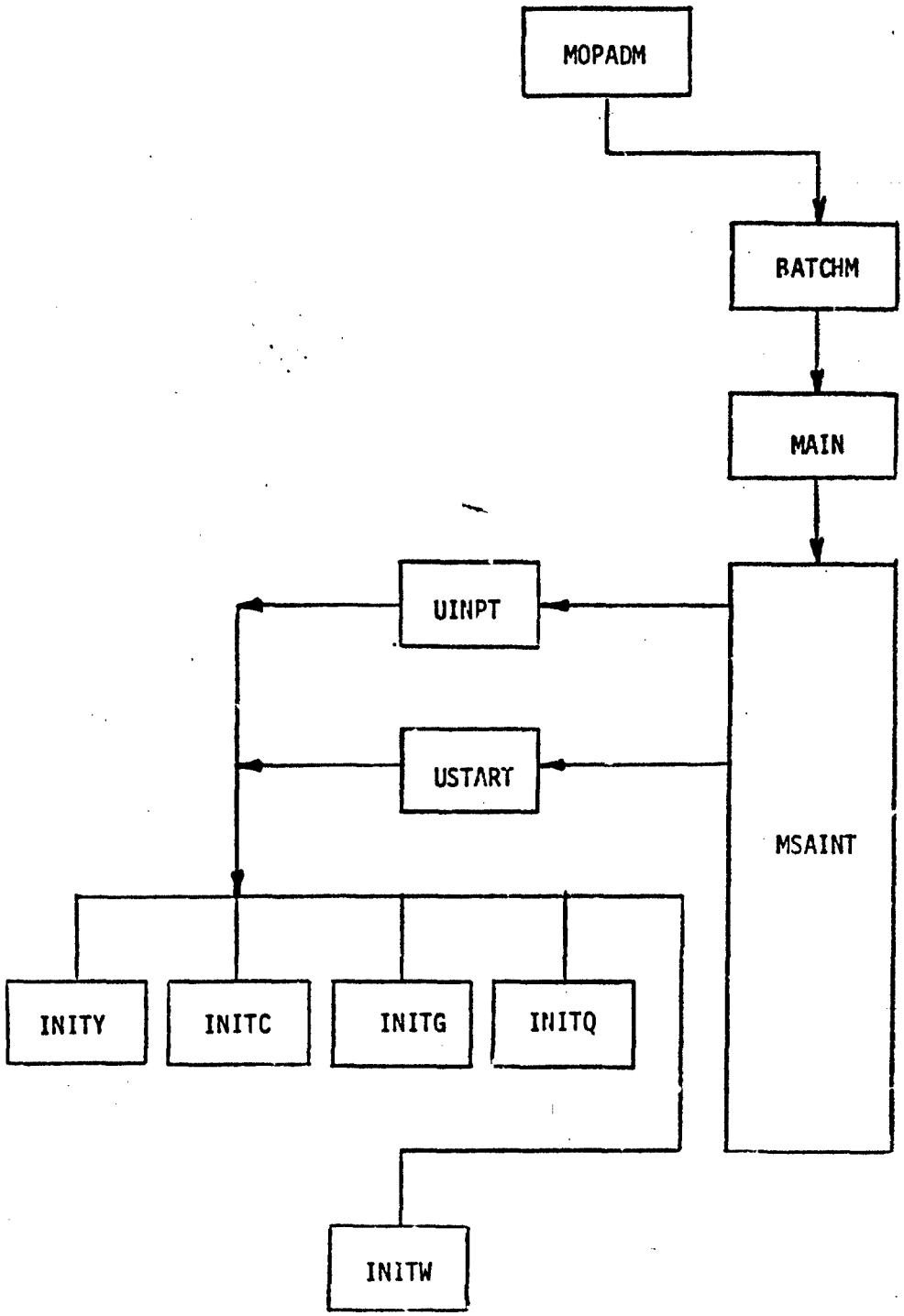


Figure III-3. Software Structure for Initialization.

4-2. Task Node Processing.

MSAINT provides the capability to execute user code at each TASK node in the MOPADS network. Figure III-4 shows the structure for accomplishing this. MSAINT calls SUBROUTINE UTASK at five points during TASK node processing (Walker, 1983a): first predecessor arrival time (if different from release time), release time, task start time, task completion time, and task clearing time. UTASK is written by the modeler to accomplish any needed programming activities at these times.

In MOPADS, UTASK calls a program for each system module: e.g., UTGRPG for the group system module. Each of these programs, in turn calls a program for each task node in the network module of the sytem module. As an example, when TASK node four in the Group AN/TSQ-73 system module is being performed, SUBROUTINE T4G will be called from UTASK and UTGRPG. T4G contains code written by the MOPADS modeler to perform activities required for the processing of that node. The structural convention shown in Figure III-4 avoids confounding of user code by segrating the processing for each TASK node in a separate subprogram.

4-3. Error Processing.

Error processing is performed at several levels in the MOPADS software. Errors detected by the Data Base Control System (DBCS),Polito (1983f),are processed internally by the DBCS. Fatal errors cause execution to terminate in the DBCS. Non-fatal DBCS errors cause return to the calling program with an error indication. If the calling program is in the user interface, the error is displayed at the terminal, and the current command is aborted. If the calling program is in the simulation software, the job is always terminated as explained below.

Errors detected in the Data Base Application Programs (DBAP), Polito (1983g),always cause termination through the DBAP error processing program, ERRORA. The error may be a DBCS error or some other logical error condition. ERRORA prints the offending data base entry for which the error condition occurred.

Certain errors can be detected by MSAINT or the system module user code. In this case, standard MSAINT error processing occurs as shown in Figure III-5. The entry point to this processing is MSAINT SUBROUTINE SERR. SERR is called with an integer error code. SERR performs standard MSAINT error output and then calls SUBROUTINE UERR. UERR is written by the MOPADS modeler. Various ranges of error code values (shown in Figure III-5) have been reserved for the different system modules. UERR calls the correct error processing program (e.g., Q73EQ for the AN/TSQ-73). These error processing subroutines are written by the developers of the MOPADS system modules to produce

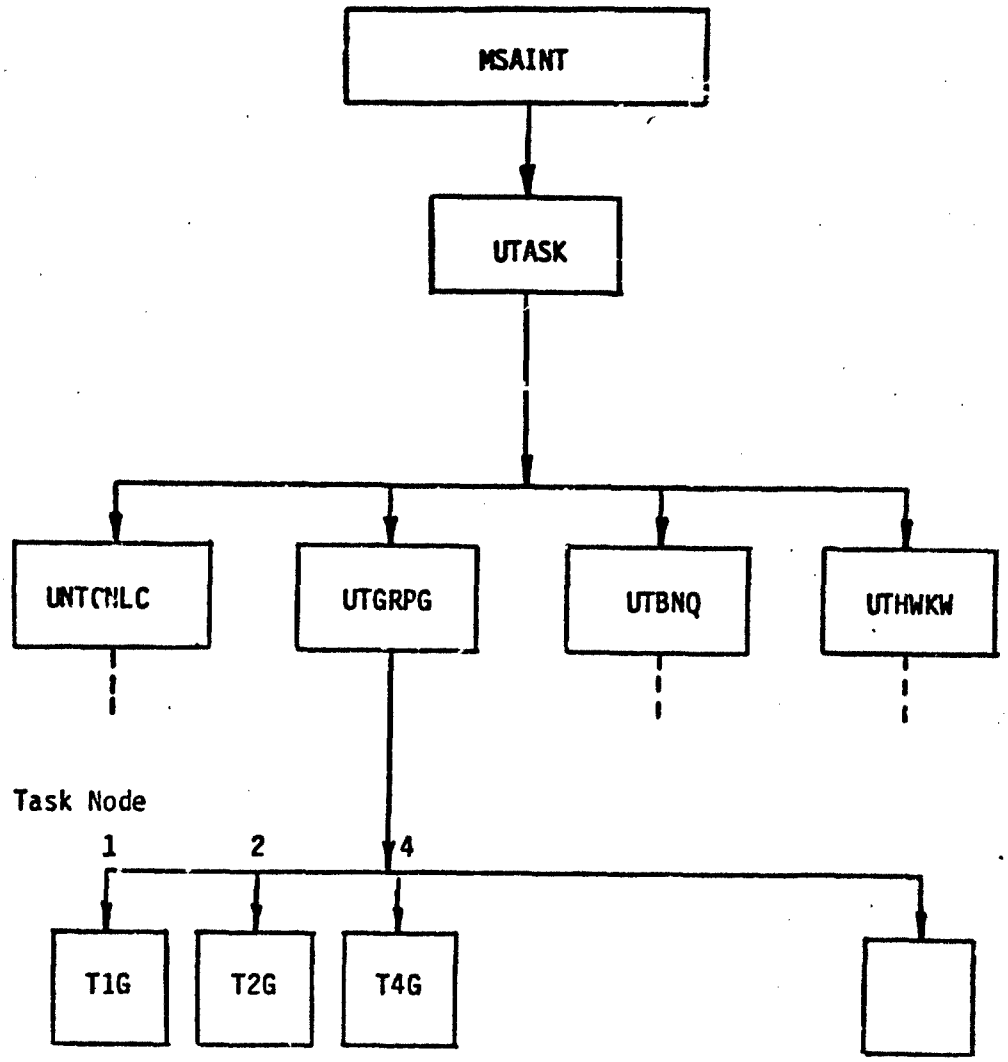
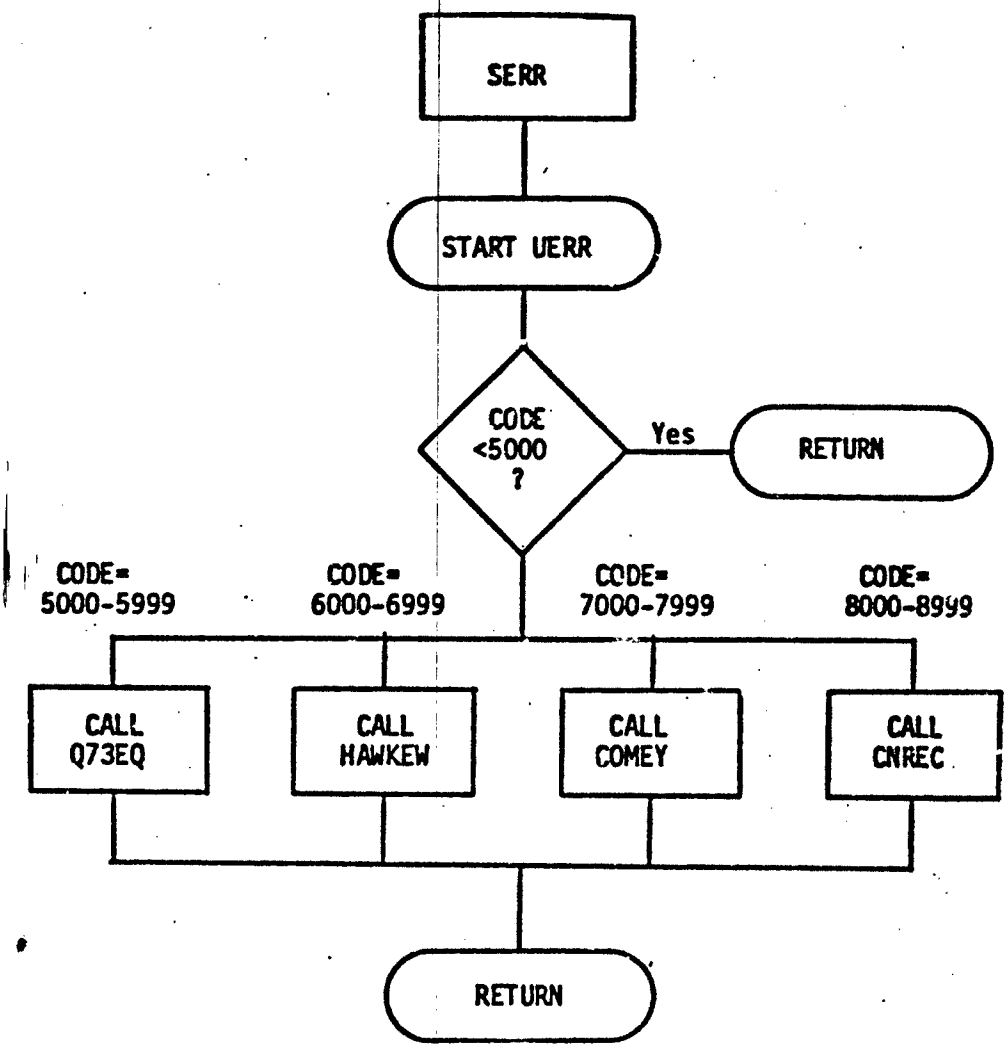


Figure III-4. Software Structure for TASK Node Processing.



Error Code Range	Type of Error
3000-4999	MSAINT Error
5000-5499	Either Group or Battalion Q-73 Error
5500-5749	Group Only Error
5750-5999	Battalion Only Error
6000-6999	IHAWK Error
7000-7999	Common System Module Programs
8000-8999	Control Error

Figure III-5. MSAINT Error Processing.

any auxiliary output necessary for the specific error codes. Note that SERR can be and is called with the system module specific error codes from programs written by the modeler to implement the module. For example, if an error is detected in SUBROUTINE T4G (see Figure III-4), SUBROUTINE SERR is called at that point with the appropriate code value.

All calls to SUBROUTINE SERR terminate execution by performing a deliberate FORTRAN error, i.e., taking the square root of a negative number. This is done so that a traceback can be obtained to the offending subprogram and line number. Note, however, that on most computer systems this traceback feature is an option which must be explicitly selected at compile and/or run time. No traceback information will be printed if the option has not been selected.

4-4. User Functions.

User functions (both the input user function, USERIN, and the exeuciton user functions, USERF) may be renumbered for each system module type. In other words, both the IHAWK and AN/TSQ-73 system module may have an execution user function numbered "1." This is implemented with the scheme shown in Figure III-6. The input and execution user functions invoke FUNCTION subprograms specific to the system module being processed. These FUNCTIONS are named in a uniform way, e.g., USERFC, USERFQ, USERNC, USERNQ, etc. This scheme allows user functions from a system module to be independent of user functions in other system modules.

4-5. MSAINT SUBROUTINE MODRF.

Unlike the user functions discussed above, the code values used for SUBROUTINE MODRF are not renumbered for each system module. Therefore, MOPADS modelers must keep track of MODRF codes as they are used across all system modules. Currently only MODRF code one has been used. Code one is used to implement the human factors moderator functions for all system modules.

4-6. Task Sequencing.

Each system module has an operator task for task sequencing. One of the nodes that make up this task, e.g, node 191 in the Battalion AN/TSQ-73 system module, has responsibility for selecting the next operator task. From UTASK, a subroutine is called to accomplish this (T191Q in the above example). T191Q can be used as a model for developing task sequencing for any future system modules. The required structure is shown in Figure III-7 for the AN/TSQ-73. The IHAWK is similar.

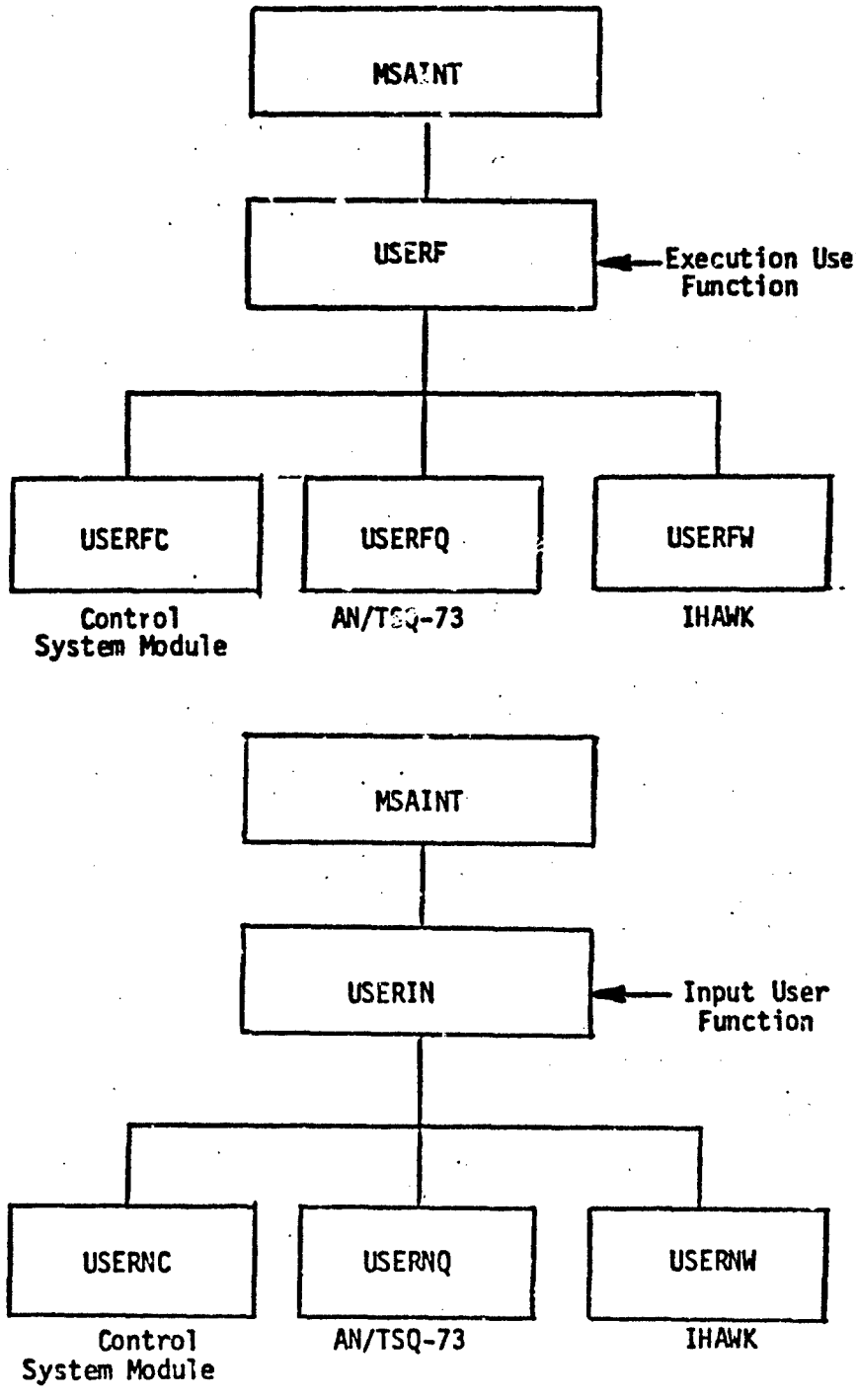


Figure III-6. Software Structure for the Input and Execution User Functions.

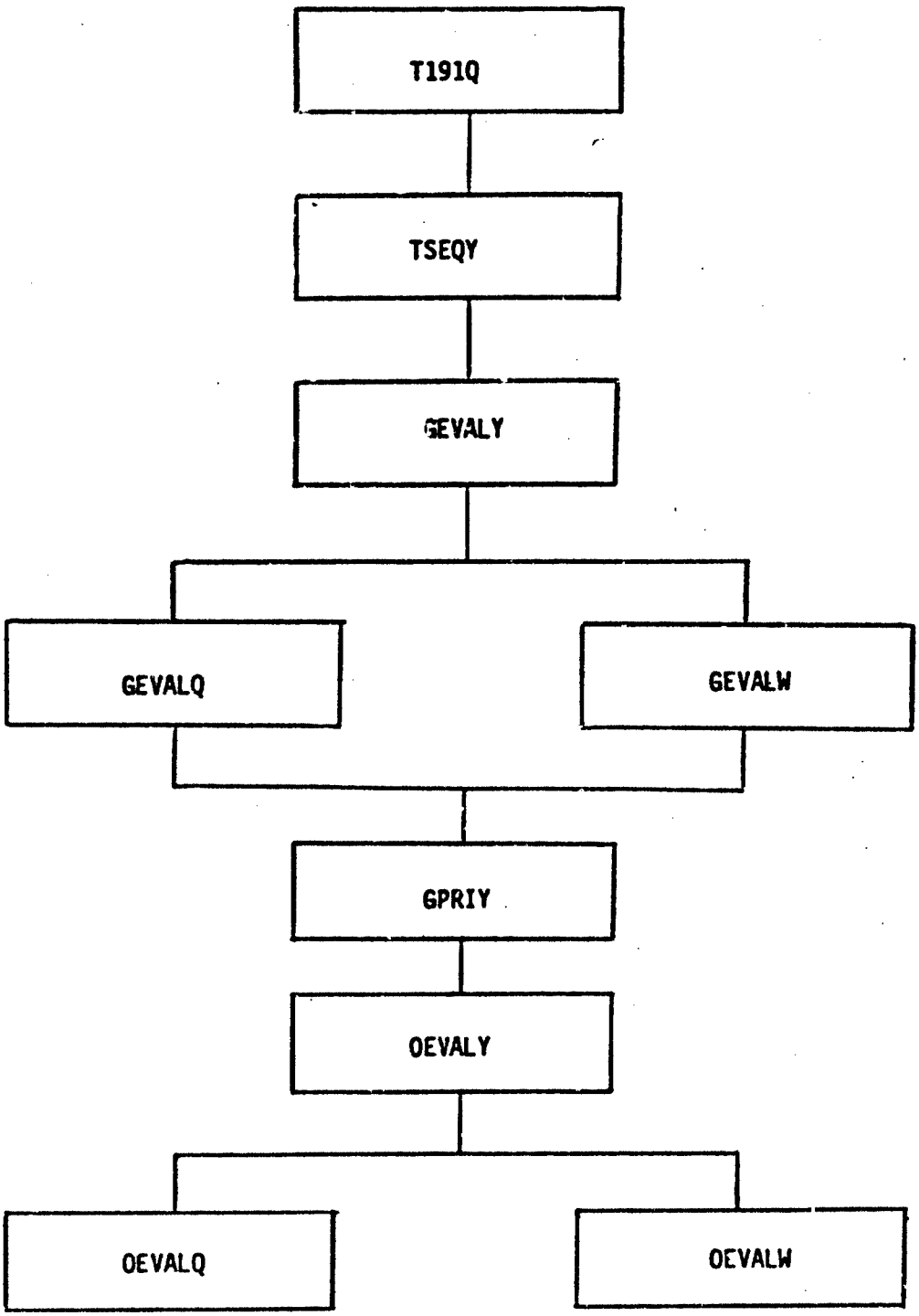


Figure III-7. Task Sequencing Software Structure for the AN/TSQ-73.

T191Q calls SUBROUTINE TSEQY to select the next task. TSEQY calls GEVALY to evaluate the current goal states. GEVALY calls programs specific to the system module (in this case, GEVALQ) to do this. On return, TSEQY calls GPRIY to evaluate the current goal priorities from the goal states (i.e., the goal dissatisfaction levels). TSEQY then calls OEVALY to estimate the impact on the goal states that will result from selecting the various operator tasks available. OEVALY calls a system module specific program to do this (in this case, OEVALQ). TSEQY returns the selected task to T191Q.

Adding new system modules will require modifying GEVALY and OEVALY. Also, programs analogous to GEVALQ and OEVALQ must be developed.

4-7. External Files.

MOPADS uses a number of external files. Figure III-8 shows the files. All file unit numbers are stored in variables in MOPADS. The main array to hold this information is KUNITM. The unit numbers shown are the current values of the corresponding elements of KUNITM. These values may be changed to conform to requirements of a particular installation by modifying only one DATA statement in BLOCK DATA program BLOCKM.

The column "Assigned By" in Figure III-8 indicates how the file is associated with the unit number. If this column has an "M" in it, it indicates that MOPADS accesses these files with FORTRAN 77 OPEN and CLOSE statements. No job control language is required on most computer systems. Those labeled JCL require that the indicated files must be associated with the correct unit number with job control language prior to executing MOPADS. Details of preparing MOPADS data files and performing simulations is contained in Polito (1983b).

5-0 MSAINT TASK NETWORKS

Very little needs to be discussed here concerning the construction of MSAINT task networks because models of system module implementation can be found in Goodin & Polito (1983a,b) and Goodin & Walker (1983a,b). Some conventions are worth discussing here, however.

Operator tasks are numbered by the MOPADS modeler (e.g., 1,2,...), and a network model of each is constructed. By convention, each task is developed with a single entry TASK node and a single exit TASK node. This helps to ensure that correct processing occurs before and after each task. Also, the node number of the entry node is always equal to the task number. For example, the entry node to operator task 8 is TASK node 8.

Certain operator information attributes have been reserved for uniform use. In other words, their meanings are the same no matter what system module the operator is from. These attribute definitions

Unit Number Variable	Unit Number	Assigned * By	File Description
KUNITM(1)	1	M	MOPADS Working Data Base
KUNITM(2)	2	M	MOPADS Reference Data Base
KUNITM(3)	3	M	MSAINT Network Input File
KUNITM(4)	4	M	MSAINT Scratch File
KUNITM(5)	5	JCL	MOPADS Batch Input File
KUNITM(6)	6	JCL	MOPADS Line Print Output
KUNITM(7)	7	-	Not used
KUNITM(8)	8	M	MSAINT Trace Output File
KUNITM(9)	9	JCL	User Interface Interactive Input
KUNITM(10)	10	JCL	User Interface Interactive Output
KUNITM(11)	11	M	System Module Network Files
KUNITM(12)	12	-	Not used
KUNITM(13)	13	-	Not used
KUNITM(14)	14	-	Not used
KUNITM(15)	15	-	Not used

NOTE:

M - MOPADS opens and closes these files

JCL - These files must be associated with the correct unit number with job control language.

Figure III-8. External Files Used By MOPADS.

are shown in Table III-2. Attributes not defined in Table III-2 may be defined to satisfy the modeling needs of a system module under development. Definitions of these attributes for the AN/TSQ-73 and IHAWK are shown in Goodin & Walker (1983a,b).

Table III-2. Standardized Operator Information Attribute Definitions

Information Attribute Number	Definition
1	Operation ID
2	Copy Row Number
3	Operator Type

See Walker (1983) for discussion of these attributes.

IV. CONCLUSION

This document has presented an overview of the MOPADS modeling approach and details of the software implementation. It's intended audience is the MOPADS modeler who will maintain or extend the MOPADS system. Details of implementation are contained in other documents, but this volume provides a guide to entry points in the documentation and source code for MOPADS.

If new system modules are developed, existing documents can serve as models (e.g., Goodin & Polito (1983a); Goodin & Walker (1983a)) for development. It is strongly recommended that the format of these reports be used to document any new system modules. Furthermore, if existing modules are modified, the documentation should also be revised to maintain up-to-date reference material.

V. REFERENCES

Goodin II, J. R., & Polito, J. User's guide for the AN/TSQ-73 system module (MOPADS Vol. 3.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Goodin II, J. R., & Polito, J. User's guide for the IHAWK system module (MOPADS 3.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Goodin II, J. R., & Walker, J. L. Documentation manual for the AN/TSQ-73 system module (MOPADS Vol. 5.15). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Goodin II, J. R., & Walker, J. L. Documentation manual for the IHAWK system module (MOPADS Vol. 5.16). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Laughery, R. L. The underlying person model behind HOMO (human operator model (MOPADS Vol. 5.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981 (a)

Laughery, R. L. HOMO establishment of performance criteria for non-decision making tasks (MOPADS Vol. 5.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981 (b)

Laughery, R. L., & Gawron, V. Human factors moderator functions (MOPADS Vol. 5.10). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. MOPADS task sequencing structure (MOPADS Vol. 5.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Polito, J. Creating references air defense system modules (MOPADS Vol. 4.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Polito, J. Creating MOPADS air scenarios (MOPADS Vol. 4.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d)

Polito, J. MOPADS free-format input programs (MOPADS/FFIN2) (MOPADS Vol. 5.14). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (e)

Polito, J. The MOPADS data base control system (MOPADS DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (f)

U. S. Army. Technical manual (TM 9-1430-652-10-3). Operator's manual initialization and operating procedures, guided missile air defense system AN/TSQ-73, August 1978, Change 6, July 9, 1981.

Walker, J. L. MSAINT user's guide: changes and additions to the SAINT user's manual (MOPADS Vol. 4.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Walker, J. L., & Polito, J. FORTRAN style and documentation requirements for MOPADS operator and non-operator software (MOPADS Vol. 4.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1982.

VI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.#1, Fort Benjamin Harrision
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Rm. 108
Boulder, Colorado 80301

VII. CHANGE NOTICES

APPENDIX E

MOPADS FINAL REPORT:

FORTRAN STYLE AND DOCUMENTATION REQUIREMENTS

TABLE OF CONTENTS

I. INTRODUCTION

The MOPADS system software contains several separate modules which interact through software interfaces. These modules will be developed independently by different programmers and will be integrated to form the MOPADS system.

In order to assure that this software development effort goes smoothly, it is necessary that common programming and documentation guidelines be followed by all software developers. Following these guidelines will facilitate debugging, modification, and maintenance of the MOPADS software.

All non-operator MOPADS software and SAINT user written FORTRAN (operator software) will conform to the requirements in this manual. For additional guidance on preparing MOPADS operator modules, see reference 1.

In Section II, the programming conventions to be followed are described. The conventions are broken down into two groups. The first group contains global considerations involving each module and its subprograms. The second group contains the programming conventions to be followed when coding individual subprograms. Included in Section II is a sample subprogram showing the standard layout to be used.

Section III contains the requirements for documenting each MOPADS software module. Each module will have its own program documentation manual, and the requirements for preparing these manuals are specified in this section.

These guidelines should place minimum restrictions on the coding style of the software development personnel while maximizing the ease with which the coordination effort and future modifications can be performed.

II. PROGRAMMING CONVENTIONS

1-0. GLOBAL CONSIDERATIONS

1-1. Each module will be assigned a one character suffix by the MOPADS Software Coordinator. This suffix will be attached to the end of all subprogram names, COMMON variable names, and labeled COMMON block names in the module.

1-2. The number of separate labeled COMMON blocks in a module will be minimized while the following guidelines are used:

1. No COMMON statement should have more than four continuations.

2. Variables may be grouped by function into separate COMMON blocks.

3. "Special purpose" COMMON blocks with only a few variables to communicate between a few subprograms are prohibited.

1-3. The variables within each COMMON block must be in alphabetical order.

1-4. No blank (unlabeled) COMMON usage is allowed.

1-5. Each module will have one error processing routine. This routine will be named ERROR? where ? is the one character module specific suffix.

1-6. Each module will have at most one BLOCK DATA program. The BLOCK DATA program will be named BLOCK? where ? is the one character module specific suffix.

1-7. All MOPADS software will be written in ANSI FORTRAN 77[2]. No nonstandard features or syntax are to be used, and no Hollerith variables will be used. (SAINT will remain in ANSI FORTRAN IV, but all user code will be written in FORTRAN 77.)

1-8. All subprograms will be placed in alphabetical order within each module.

1-9. Modular programming design will be used, and each subprogram will have a single purpose. Very long subprograms should be avoided. As a general guideline, no subprogram should be longer than 200 to 300 non-comment lines.

1-10. All software developed on the Pritsker & Associates' VAX 11/780 will use the LAMP programs and procedures.

1-11. All logical unit numbers for external files to be opened and closed for use by non-operator software will be obtained from the MOPADS Software Coordinator.

1-12. All I/O statements will use a variable name for the logical unit number. These variables will be placed in COMMON.

1-13. Whenever the program size might need to be increased, array dimensions will be specified in a FORTRAN 77 PARAMETER statement. (On the Pritsker & Associates's VAX, if the arrays are in COMMON, the PARAMETER statement will be included in the LAMP COMDECK.)

1-14. Each module will have a single initialization subprogram named INIT?, where ? is the one character module-specific suffix.

2-0. SUBPROGRAM CODING

2-1. Comments should be used frequently to explain non-obvious statements or blocks of code. As a general rule, no block of code should be longer than about ten (10) lines without a comment.

2-2. No more than one array will be initialized in a single data statement. For arrays of more than one dimension, the data statement should be layed out in a readable manner.

2-3. No variable name may be split between continuation lines.

2-4. FORMAT statements will not use all available 72 columns before continuation. FORMAT "fields" (specifications separated by commas) should not be split between continuation lines.

2-5. The list of formal parameters in a SUBROUTINE or FUNCTION statement will be in the following order:

1. Input only parametes.
2. Input/output parameters.
3. Output only parameters.
4. Alternate returns.

2-6. Each external file will be explicitly opened and closed using the OPEN and CLOSE statements.

2-7. Each block-if will be indented by two (2) spaces as in the following two examples.

1.
```
IF (I.EQ.5) THEN
  XXVAL = -1.0
  GO TO 999
ENDIF
```

2.
```
IF (NPT.LT.0) THEN
  NERR = 500
ELSE
  NERR = 1000
ENDIF
```

2-8. Each DO loop with more than one statement will have its own labeled CONTINUE statement.

2-9. The TYPE statements INTEGER, REAL, and IMPLICIT will not be used.

2-10. Each subprogram will have one and only one return statement. One of the two options shown below will be used:

1.
```
      ------
      ------
      ------
999   RETURN
      END
```

2.
```
      ------
      ------
      ------
999   CONTINUE
      RETURN    <-(Only debugging statements
      END           are allowed here.)
```

2-11. Statement numbers will be in ascending order, with the RETURN (or preceding CONTINUE to the RETURN) labeled 999 and all FORMAT statements using numbers from 1000 to 1999.

2-12. All variable length arrays passed to subprograms as formal parameters will have adjustable dimensions in the subprograms (i.e., the array bounds must be passed as formal parameters or be contained in a COMMON or PAPAMETER statement in the subprogram).

2-13. Each subprogram will be organized in the following way. Sections not needed will be excluded.

1. Comment statement with 'C:' in the first two columns.

2. Subprogram name and arguments.

3. Block of comment statements in the following order:

   a. MOPADS module name
   b. MOPADS documentation reference
   c. Description of the purpose of the subprogram
   d. Definitions of input only parameters
   e. Definitions of input/output parameters
   f. Definitions of output only parameters
   g. Descriptions of all alternate returns
   h. Comment statement with 'C::' in the first three columns
   i. Definitions of local variables whose usage is not readily apparent

4. All DIMENSION statements (perhaps preceded by their PARAMETER statements).

5. All COMMON statements (perhaps preceded by their PARAMETER statements).

6. All EQUIVALENCE statements.

7. All DATA statements.

8. Main body of the subprogram with statement numbers (1-998) in ascending order.

9. RETURN statement (labeled 999).

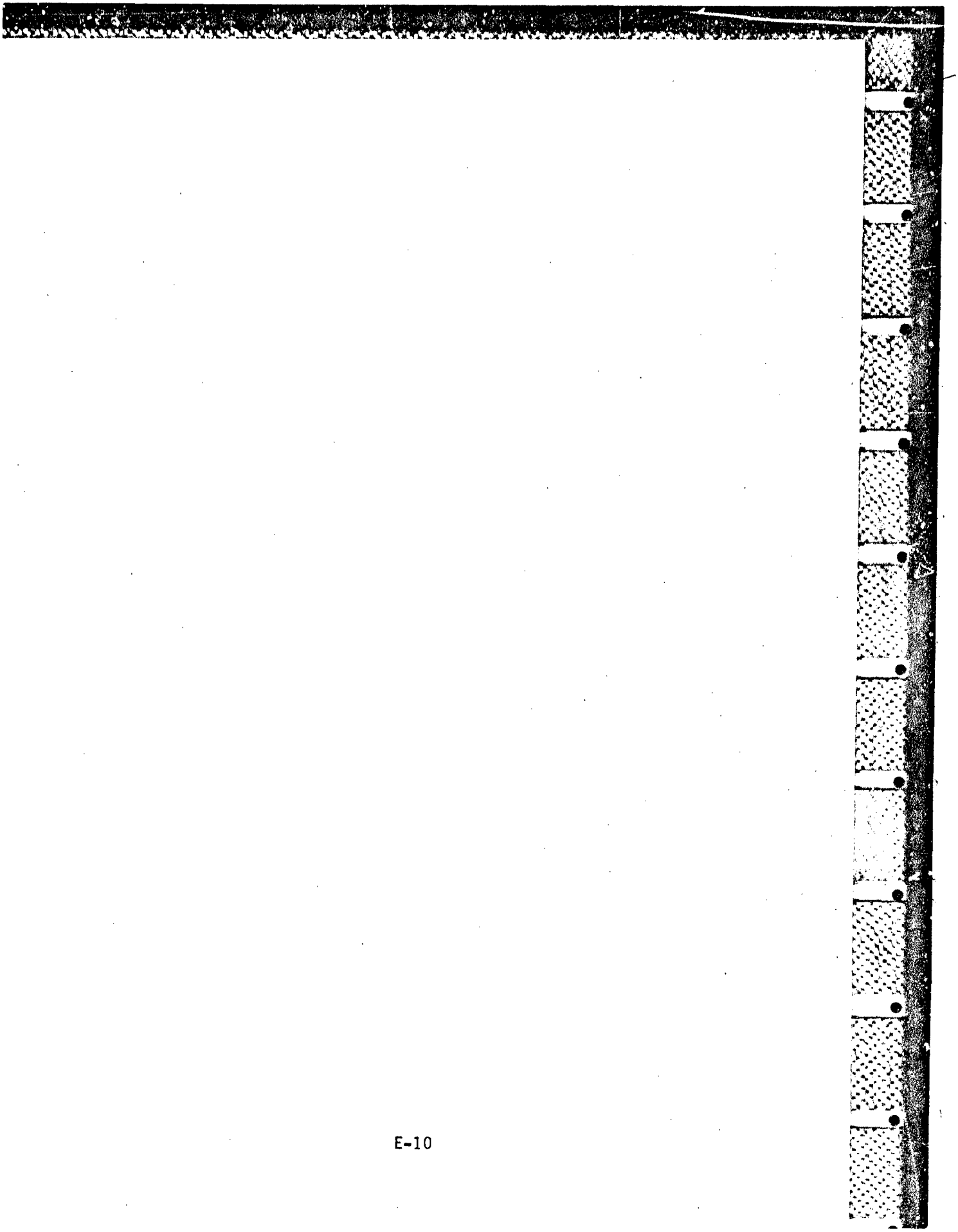
10. All FORMAT statements (1000-1999).

11. END statement.

3-C. SAMPLE SUBPROGRAM SHOWING STANDARD LAYOUT

The following example conforms to the previous requirements.

```
C:
      SUBROUTINE DFIND(XVAL,XARRAY,NDIM,TOL,NCURR,IDX,IERR,*)
C
C**MODULE:  MOPADS/MODULE NAME
C**REFERENCE:  MOPADS/VOLUME NUMBER
C
C**PURPOSE:  THIS SUBROUTINE RETURNS THE ARRAY INDEX FOR THE
C            FIRST (OR NEXT) ARRAY ELEMENT WHICH IS EQUAL TO
C            A GIVEN VALUE + OR - A SPECIFIED TOLERANCE.
C
C**INPUT PARAMETERS:  XVAL=VALUE TO BE COMPARED
C                     XARRAY=ARRAY NAME FOR ARRAY TO BE
C                          SEARCHED
C                     NDIM=DIMENSION OF XARRAY
C                     TOL=TOLERANCE USED IN COMPARING XVAL
C                         TO THE ELEMENTS OF XARRAY
C
C**INPUT/OUTPUT PAR:  NCURR=THE INDEX OF THE ARRAY WHERE THE
C                          SEARCH IS TO BE STARTED.  RETURNED
C                          AS IDX+1 IF IERR=1
C
C**OUTPUT PARAMETERS: IDX=THE INDEX OF ARRAY XARRAY SUCH THAT
C                         XVAL + OR - TOL = XARRAY(IDX)
C                     IERR=0 IF NONE FOUND
C                          1 IF NO ERRORS AND IDX WAS SET
C                          2 IF NCURR>NDIM
C
C**ALTERNATE RETURNS: THE SINGLE ALTERNATE RETURN OCCURS WHEN
C                     IERR NOT EQUAL 1
C
C::
C**LOCAL VARIABLES:   ANY LOCAL VARIABLES WHOSE USAGE IS NOT
C                     READILY APPARENT SHOULD BE DEFINED HERE
C
      DIMENSION XARRAY(NDIM)
C
      IDX = 0
      IRT = 0
      IERR = 0
      XPLUS = XVAL+TOL
      XMINUS = XVAL-TOL
      IF(NCURR.GT.NDIM) THEN
        IERR = 2
        IRT = 1
        GO TO 999
      ENDIF
C
C**SEARCH XARRAY STARTING AT ELEMENT NCURR
C
      DO 100 I=NCURR,NDIM
      IF(XARRAY(I).GE.XMINUS.AND.XARRAY(I).LE.XPLUS)THEN
        IDX = I
        NCURR = I+1
        IERR = 1
        GO TO 999
      ENDIF
  100 CONTINUE
C
C**NOT FOUND
C
      IRT = 1
C
  999 RETURN IRT
      END
```



III. DOCUMENTATION REQUIREMENTS FOR MOPADS NON-OPERATOR SOFTWARE MODULES

Each module will have its own program documentation manual. Each of these manuals will have the following table of contents. A description of each section follows.

1-0. TABLE OF CONTENTS

2-0. DESCRIPTION OF EACH SECTION

2-1. Purpose. This section will serve to introduce the reader to the module. The purpose and scope of the module will be discussed. When the reader finishes the section, he/she should understand the function of the module.

2-2. Data Structures Descriptions. Array, tree, list, and pointer usage should be described here. These descriptions may include tables, diagrams, or pictures. Modules which create entries in the MOPADS data base must define this usage in this section.

2-3. Overview of the Flow-of-Control. This entails a description of how the module fits into the MOPADS system, how the module is entered and exited, and some description of the flow-of-control of the module. A hierarchy chart may be included here.

2-4. External File Usage. A short description of the usage of each external file used by this module should be included. The type of file should be given as well as a discussion of when the file is opened and closed.

2-5. Subprogram Descriptions. Every subprogram must be described. The purpose and parameter definitions from the subprogram comment section may be included here. These subprogram

descriptions will be in alphabetical order. In addition to the purpose and parameter definitions, some discussion should be included describing the method and/or flow-of-control within the subprogram. Flowcharts should be included for very long subprograms. Any local data structures should be described. Alternate ENTRY's into a subprogram will be documented as though they were separate subprograms.

2-6. User Instructions. This section should contain descriptions of how to use the module. Examples may be provided. Reference may be made to Section V for detailed program descriptions rather than repeating them here. The instructions given in this section should be issue driven. In other words, explain how to use the module to accomplish its purpose.

2-7. Error Processing. The error processing programs will be discussed here. If error codes are used, each will be defined in this section. A discussion of the types of errors that are detected and what action is taken will be included.

2-8. COMMON Variable Definitions. A table of all COMMON variables, their definitions, and their initial values (if appropriate) will be included. The variables may be either alphabetized within or across COMMON blocks. In any case, the COMMON block name in which a variable appears must be clearly identified.

IV. REFERENCES

1. Walker, J. L. & Polito, J. Documentation requirements and development guidelines for MOPADS operator modules (MOPADS Vol. 4.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, February 28, 1982.

2. FORTRAN 77 Manual, American National Standards Institute, Document X3.9-1978.

V. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACC-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VI. CHANGE NOTICES

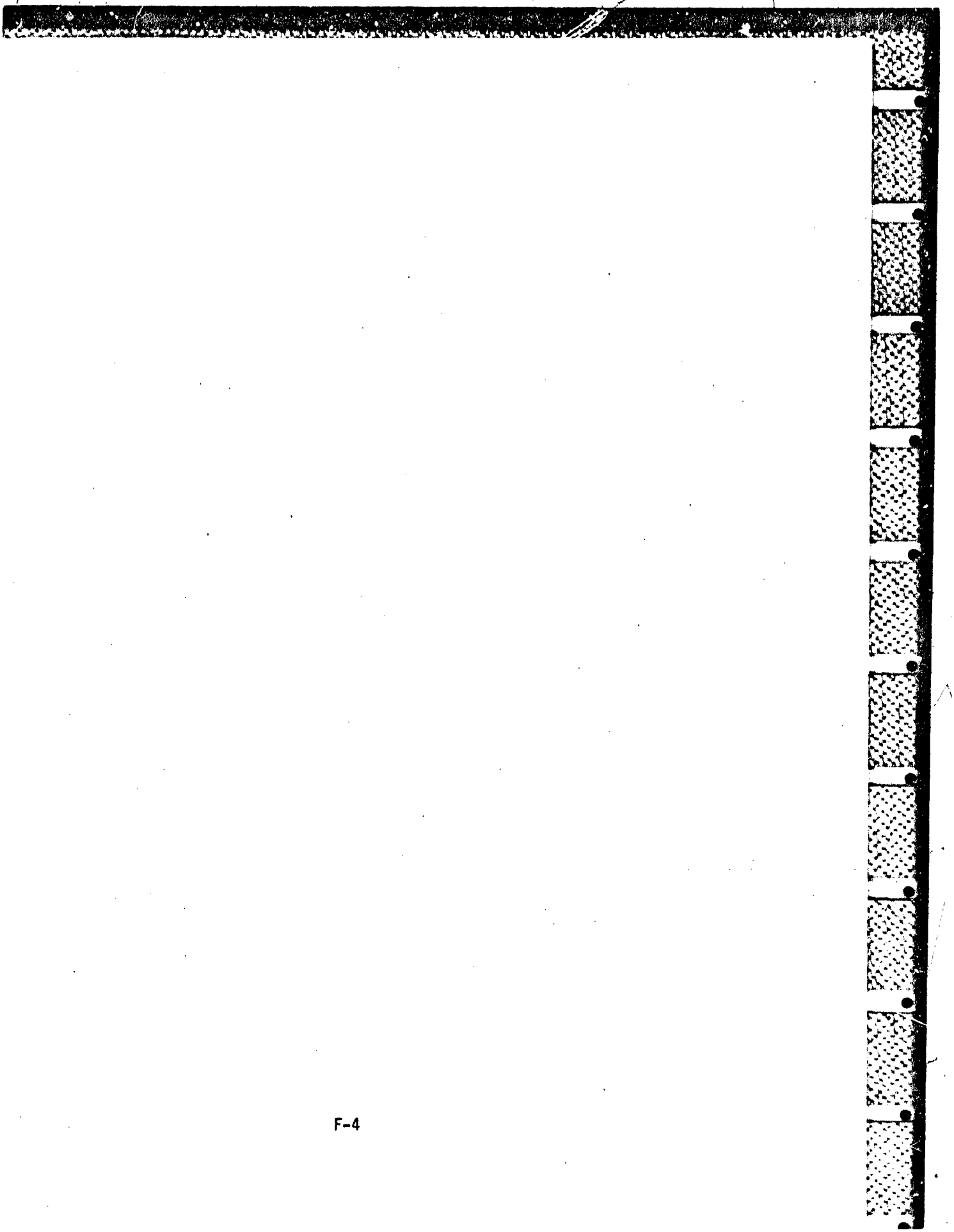
APPENDIX F

MOPADS FINAL REPORT:

DOCUMENTATION REQUIREMENTS AND DEVELOPMENT GUIDELINES FOR MOPADS AIR DEFENSE SYSTEM MODULES

TABLE OF CONTENTS

LIST OF FIGURES



TERMINOLOGY

1-0 STANDARD MOPADS TERMINOLOGY.

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

I. INTRODUCTION

The MOPADS software will allow multiple air defense system modules to be built and "plugged in" to the overall system. These SAINT modules will each represent operation of an air defense component. They will most likely be developed independently, and it can be expected that many changes will be made to the models during their useful life. This requires that each module be well documented so that other MOPADS modelers (or the same modeler at a later time) can understand the workings of each module. This report outlines the methodology that will be used in documenting all MOPADS system modules.

Each system module will have its own documentation and reference manual, and each of these manuals will follow the same basic format. Section II contains the standard table of contents for each manual.

Section III contains descriptions and explanations of the contents of each chapter listed in the standard table of contents.

A major portion of each of these manuals will be a collection of forms which describe the entities, resources, variables, etc. in the model. Sample forms with example contents are included in Section III. 3.0.

Section IV shows the blank forms needed by the MOPADS modeler. Under separate cover full-size forms will be sent to the MOPADS modeler. These forms should all be kept in a notebook and filled out as the information in them becomes available during system module development. All forms of a given type should be kept together, and the forms should be put in the same order as the forms appear in Section IV.

(Blank Page)

II. STANDARD MODULE TABLE OF CONTENTS

Since each system module documentation manual will contain the same type of information, the documentation task will be simpler if each has the same organization. A standard table of contents in shown in Figure II-1.

Figure II-1. Table of Contents for MOPADS System Module Documentation

III. DESCRIPTION OF THE DOCUMENTATION CONTENTS

In this section the contents of the system module documentation manual are explained and discussed. Each heading in this section corresponds to a chapter to be included in the manuals.

1-0 SYSTEM DESCRIPTION.

The system being modeled will be described here. This description should include discussions of which equipment is included in the system, what operators run the equipment, and how the equipment is operated. Some discussion of how this equipment interfaces with other equipment should be included. This section may include diagrams or pictures from technical manuals, soldier's manuals, or field manuals. Any special configurations of the equipment should be described.

2-0 OVERVIEW OF THE SAINT MODEL.

The modeling approach applied to the system will be explained here. This explanation will include such items as simplifying assumptions, overall modeling framework, modular aspects of the model, and the flow-of-control of the network model. Detailed discussions of the model should be reserved for the following section. The reader should be able to get a feel for the basic model structure by reading this section.

3-0 MODEL DESCRIPTION FORMS.

In order to simplify the model documentation process, several forms have been developed. These forms are intended to be kept in a notebook and filled in as the model is being developed. In this way, they serve as working documentation during model development. If the MOPADS modeler is diligent in maintaining the forms, then the final documentation tasks will be much simpler and nothing of significance to the model will be forgotten. The use of each form is explained, and a filled out sample of each form is included in this report.

3-1. Entities.

A sample Entities form is shown in Figure III-1. Entities are described in reference [1] and they may be operators or information flows. An entity is created in a SAINT network either at a source node or at a regular task node through mulitple branching. Regardless of which method is used, the task node

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
2	CWTDC Operator	1	Rank	3.0	2,CWTDC panel
		2	Skill level	7.0	5,ADP
		3	Duty time span	2.0	9,IHIPIR
5	TCC Operator	1	Rank	4.0	3,TCC panel
		2	Skill level	5.0	5,ADP
		3	Duty time span	2.5	8,ATDL-1

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: HAWK BCC
PROJECT: MOPADS

SAINT ENTITIES PAGE 1 OF 1

Figure III-1. Sample ENTITIES Form.

number where the entity is created will be entered in the first column. The entity will be briefly described in the second column, and this description should tell the reader what each entity represents. Each entity has a collection of attributes, called information attributes, which follow the entity through the SAINT network. Each attribute number will be listed in the third column, accompanied by a brief definition describing its meaning and initial value, if appropriate. Information attributes are initialized through the ATAS descriptor on the task node where the entity originates. It is probable that each entity requires the use of at least one resource to perform one or more tasks in the network. If so, each resource which the entity references throughout the network should be listed in the far right column. These entries should consist of the resource number, followed by a comma, then the resource label.

3-2. Resources.

The SAINT simulation methodology allows for the use of resources for modeling applications. (See reference 1 for further discussion on the use of resources.) A sample resources form is shown in Figure III-2.

Each resource number will be listed in the first column with the corresponding resource label adjacent to it in the second column. Under the "Description of Use" column, the resource should be defined and some discussion should be included describing how each resource is used.

Each resource may have resource attributes. Each resource attribute number will be listed, followed by a brief definition of the resource attribute and its initial value. Resource attributes are initialized on IRA cards. All task numbers requiring the resource are listed in the last column. This information is extremely valuable when network changes are made involving the resource.

3-3. Variables.

Each SAINT model will have many variables that are global in nature. There are three types of global variables:

1. state variables
2. system attributes
3. user variables.

All of these will be entered on the same form, but they will be separated by type. Examples showing the use of the variables form are shown in Figures III-3, III-4, and III-5.

RESOURCE NUMBER	RESOURCE LABEL	DESCRIPTION OF USE	RESOURCE ATTRIBUTES			TASK NUMBERS REQUIRING THE RESOURCE
			ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE	
5	ADP SYSTEM	Whenever an operator requests information from the ADP system, he can get it only if the ADP is operational and not busy processing other information.	1	Speed of elemental calculation	.0001	57,58,71,91,103
			2	Mean time between breakdowns	1.3	
			3	Number of connections	57	
1	AF COMMAND	Whenever a group Q73 requests a directive from higher authority, they can get that directive only when the command center is not busy.	None			61,62,153

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: Group Q73
PROJECT : MOPADS

SAINT RESOURCES

PAGE 1 OF 2

Figure III-2. Sample RESOURCES Form.

3.3.a. State variables. Figure III-3 shows the use of the variables form for documenting state variables. Near the bottom of the form, the type of variable is checked; in this case the state variables box is checked. The variable name is placed in the first column as SS(*) or DD(*) (where * is the state variable index number). In the second column, the definition of the state variable is given along with the defining equation. All units should be clearly defined. The initial value is placed in the third column.

3.3.b. System attributes. Figure III-4 shows the use of the variables form for defining system attributes. The system attributes box is checked in the section labeled "type of variables." The system attribute name is entered in the first column with the index number in parentheses. The definition is entered in the second column. If the definition is a code as for SA(5), the meaning of each possible code value should be defined. The initial value is entered in the third column. System attributes are initialized on ISA cards.

3.3.c. User variables. Figure III-5 shows the use of the variables form for defining user variables. A user variable is one that is defined by the user and referenced only in user-written code. It may or may not be placed in a user common block. The variable name, definition, and initial value are placed in the first three columns. The common block name where the variable is stored is entered in the last column. User variables should be entered alphabetically or sorted by common block.

The three types of variables should be kept separated for easier reference.

3-4. Monitors.

All SAINT monitors must be clearly defined on the monitors form. An example showing the use of this form is given in Figure III-6. There are two kinds of monitors: SAINT monitors and USER monitors. SAINT has a built-in capability to detect threshold crossings and take appropriate action. These are referred to on the form as SAINT monitors, and they are discussed in references [1] and [2]. Use of user monitors is described in reference [3].

The monitor number and label are entered in the first two columns, and the type of monitor (USER or SAINT) is entered in the third column. A verbal description of the significance of the monitor is entered in the fifth column with a text or equation representation of the threshold crossing conditions in the fourth column. The action to be taken when a threshold crossing occurs is entered in the last two columns. Under CODE, either MTAS or MSWT (or both) are entered, followed by a description of the action taken.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITIAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
SS(1)	GCV 1 X Coordinate (in feet) =SSL(1) + COS(H1)*V1*DTNOW (where H1 is heading, V1 is velocity)	0	
SS(3)	GCV 1 Y Coordinate(in feet) =SSL(3)+SIN(H1)*V1*DTNOW	0	

TYPE OF VARIABLES: [X] STATE VARIABLES [ ] SYSTEM ATTRIBUTES [ ] USER VARIABLES

NAME: J.L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: GCV System
PROJECT: AF 81-3

SAINT VARIABLES PAGE 1 OF 1

Figure III-3. Sample VARIABLES Form Showing State Variables.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITIAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
SA(1)	Temperature in Centigrade	22	
SA(2)	Lighting at console 1 in lumens	300	
SA(3)	Lighting at console 2 in lumens.	270	
SA(4)	DEFCON code	3	
SA(5)	Configuration arrangement (1 = 1 Group 2 Battalion,Triangular) (2 = 2 Group 4 Battalion,Triangular) (3 = 0 Group 1 Battalion,Rectangular)	2	

TYPE OF VARIABLES: ☐ STATE VARIABLES ☒ SYSTEM ATTRIBUTES ☐ USER VARIABLES

NAME: J.L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: Q73 Battalion
PROJECT: MOPADS

SAINT VARIABLES PAGE 1 OF 1

Figure III-4. Sample VARIABLES Form Showing System Attributes.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITIAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
DTIME	Amount of task time added or subtracted to TTIME in MODRF	0.00	UCOM1
XDELAY	Random variable set equal to the time it takes an operator to respond to a call on the net (in minutes).	0.10	UCOM3

TYPE OF VARIABLES: ☐ STATE VARIABLES ☐ SYSTEM ATTRIBUTES ☒ USER VARIABLES

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: Q73 Battalion
PROJECT: MOPADS

SAINT VARIABLES

PAGE 1 OF 1

Figure III-5. Sample VARIABLES Form Showing User Variables.

MONITOR NUMBER	MONITOR LABEL	USER OR SAINT MONITOR	MEANING	CROSSING CONDITIONS	ACTION TO BE TAKEN	
					CODE	MEANING
1	STOP GCV 1	SAINT	The simulation will end when both GCV's reach 500000 ft.	SS(1)≥ 500000.	MTAS	Signal Task 5 once. A second time task 5 is signaled, the simulation ends.
2	COMBINED HT	USER	The system changes status when the combined distance of the 2 GCV's reaches 800000 ft	SS(1)+SS(3)≥ 800000.	MSWT	Set switch 1 to 1 to signify new state of alert.

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: GCV Model
PROJECT: AF 81-3

SAINT MONITORS

PAGE 1 OF 1

Figure III-6. Sample MONITORS Form.

3-5. Task Descriptions.

Each task node in the SAINT model will be described using the SAINT Task Description form. An example showing the use of this form is given in Figure III-7. It would be advantageous to use a separate sheet (or sheets) for each task node so that they can easily be sorted by task number. The task number is entered in the first column, followed by a description of the node. The description will define what task (or group of tasks) is represented by the task node. Some discussion should be included here describing the subsequent branching from the node. Predecessor and subsequent predecessor requirements should be described if their usage is not clear. Each resource required for the task should be listed and described under the heading RESOURCE REQUIREMENTS.

Each task descriptor (except RESR) will be entered under the heading DESCRIPTORS. The 4-letter descriptor code is entered under CODE. The remaining descriptor parameters may be placed under MEANING, but they may be left off this form as they can be referenced in the SAINT data input section. The main reason for putting descriptors on this form is to verbally describe the usage of each descriptor. It may be advantageous, however, to list the moderator function numbers along with any MODF descriptors.

The TECHNICAL REFERENCES column is for listing any references which were used in defining the task or determining the task descriptors.

3-6. Statistics.

SAINT allows a great deal of flexibility in the generation of both task statistics and user collected statistics. (See references [1,2].) Since the task statistics are collected through the use of the STAT descriptor on task nodes, the collection of task statistics will be documented under the DESCRIPTOR heading of the form for the task nodes. For user statistics, the form shown in Figure III-8 will be used.

Each user statistic has an identification number which is placed in the second column. The statistic type will be entered in the first column. One of the following codes will be used for statistic type:

Statistic Type Code	Meaning
TPR	Time Persistent Regular
TPA	Time Persistent Averaged
OBS	OBservation Statistic
HIST	Histogram
PLOT	Plot

TASK NUMBER	DESCRIPTION (INCLUDING BRANCHING INFORMATION)	RESOURCE REQUIRMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	
4	Determine status of the GCV. This is done by calling user function 1 to set IA(2) to the flight deviation. Conditional take first branching is based on the value of IA(2). If the deviation is greater than 100, then the operator must correct the flight of the GCV (this is done at task node 6).	1	Operator who checks the GCV status.	LABL TIME ATAS MODF	Determine status Distribution Set 2 (normal) Call UF(1) to set IA(2) This is setting the GCV deviation. The time to acquire the GCV deviation is moderated by the operator skill level, lighting, and fatigue (function numbers 1,4,7)	TM34934-3731-3-10, Pg.4.12

NAME: J. L. Walker
DATE: 2/19/82
AIR DEFENSE SYSTEM MODULE: GCV Model
PROJECT: AF 81-3

SAINT TASK DESCRIPTION PAGE 4 OF 6

Figure III-7. Sample TASK DESCRIPTION Form.

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED
OBS (with HIST)	1	GCV 1 DEV	Deviation of the flight path of GCV 1 at x-coordinate of 500000 ft.	Moderator function 1 at task 5 is used to collect these observations. UHIST is also called here for histogram generation.
TPR	1	NO OPR BUSY	Number of operators busy at any time. There are 2 operators, both modelcd as resources.	The operators change busy status at task nodes 3,7, and 10. At those nodes, user function 7 is called, so there are calls to UTMST there. This statistic is printed at the end of each run by calling UTMST from ENDIT.

NAME: J. L. Walker

DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: GCV Model

PROJECT: MOPADS

SAINT USER STATISTICS PAGE 1 OF 1

Figure III-8. Sample USER STATISTICS Form.

The statistic name will be entered in the third column, and a description of the statistic will be placed in the fourth column. A verbal description of how the statistic is collected and reported is entered in the last column. This will include such information as what subprograms contain the appropriate calls to the user support subprogram, when that subprogram is called, and whether the statistic is cleared at the end of each iteration or averaged over all iterations.

3-7. User Functions.

SAINT has two user-written functions: USERF and USERIN. Function USERF is called during execution whenever the UF designation is used for task times or attribute assignments (see references [1] and [2]). Function USERIN is used only for initialization of resource or system attributes on ISA or IRA cards (see reference [3]). The form for documenting the user functions is shown in Figure III-9.

The box next to the appropriate user function name (USERIN or USERF) should be checked, and separate sheets should be used to keep the two separate. Each user function number will be entered in the first column. The user function number is the same as the parameter value passed into the user function. Each task number (USERF only) where the user function is called is listed in the second column. Under the DESCRIPTION heading, the operations performed by that user function are described.

All of the user function numbers should be in ascending order for easy reference.

3-8. Moderator Functions.

The use of moderator functions will be described using the form shown in Figure III-10. The moderator function numbers are entered in the first column in ascending order. All task numbers referencing the moderator function are listed in the second column. Under the DESCRIPTION column, the calculations involved in the moderator function are described. In the last column, any references which were used in determining the moderator function are listed.

4-0 USER-WRITTEN SUBPROGRAMS.

A section will be included in each manual describing user-written subprograms. This section will be divided into three subsections. First, an index page will list all user-written subprogram names. A standard format as shown in Figure III-11 will be used for this index. User-written subprograms will be divided into two groups: standard user-written subprograms and additional user-written subprograms. The first group contains what are referred to as "standard user-written subprograms." These are called directly

USER FUNCTION NUMBER	TASK NUMBER(S) WHERE CALLED (USERF ONLY)	DESCRIPTION
1	4	Compute the deviation of the current GCV and collect statistics on the deviation.
2	1,2	Set SA(3) and SA(4) to the x coordinates of GCV 1 and 2, respectively.
3	1,2	Set SA(1) and SA(2) to the y coordinates of GCV 1 and 2, respectively.

WHICH USER FUNCTION? ☐ USERIN ☒ USERF

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: GCV Model
PROJECT: AF 81-3

SAINT USER FUNCTIONS PAGE 1 OF 7

Figure III-9. Sample USER FUNCTIONS Form.

MODERATOR NUMBER	TASK NUMBER(S) WHERE USED	DESCRIPTION	REFERENCE
1	3,6,15,127	Adjust the task time according to operator skill level.	
2	7,23,101	Adjust the task time according to console lighting.	
3	4	Set user variable UPRIOR	

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: HAWK BCC
PROJECT: MOPADS

SAINT MODERATOR FUNCTIONS PAGE 1 OF 5

Figure III-10. Sample MODERATOR FUNCTIONS Form.

SAINT USER-WRITTEN SUBPROGRAMS

Check all Standard User-Written subprograms that are used in this model. Then add at the bottom the names of other user-written subprograms.

STANDARD USER-WRITTEN SUBPROGRAMS
☐ SUBROUTINE ENDIT
☐ SUBROUTINE INTLC
☐ SUBROUTINE MODRF
☐ FUNCTION PRIOR
☐ SUBROUTINE STATE
☐ SUBROUTINE UACCPT
☐ SUBROUTINE UERR
☐ SUBROUTINE UINPT
☐ SUBROUTINE USCOND
☐ FUNCTION USERF
☐ FUNCTION USERIN
☐ SUBROUTINE UOTPT

ADDITIONAL USER-WRITTEN SUBPROGRAMS

Figure III-11. Form for Indexing User-Written Subprograms.

by SAINT. The MOPADS modeler will simply check the appropriate boxes for the ones which are included in the model being documented. The second group are subprograms that are called by subprograms in the first group. These will be listed in alphabetical order in the box in the lower part of the form.

The index page will be followed by a section containing descriptions of the user-written subprograms. These descriptions should include the subprogram purpose, parameter definitions, a discussion of what the subprogram does, and possibly a discussion of the flow-of-control of the subprogram or a flowchart. The subprogram descriptions can be put into the two groups (standard and others) and then alphabetized, or simply alphabetized as a single group.

Following the subprogram descriptions, listings for all of the subprograms will be included. These should be alphabetized in the same way as the subprogram descriptions. Extremely lengthy listings may be bound separately if a reference to the separate binding document is given.

5-0 LISTING OF SAINT NETWORK DATA INPUT.

A listing of the SAINT data input statements will be included here. This listing will start with the GEN card and end with a FIN card and will include all task nodes and network elements as they are represented by the SAINT data cards.

6-0 NON-SAINT DATA REQUIREMENTS.

The system module has access to external data sources through user-written programs. The MOPADS modeler must describe in this section any data requirements of the module. In particular, interaction with the MOPADS data base must be described. Information in this section should include:

6-1. Data files opened and closed by the module. This includes requirements for pre-existing files and files created by the module. Detailed file formats will be given.

6-2. Interaction with the MOPADS data base. This includes entries which must be created in the data base prior to running the module and those created/destroyed by it during simulations. Details of the information transfer must be given.

7-0 OUTPUT REPORTS.

This section will include an index of all SAINT output reports generated for the air defense system module, followed by sample outputs for each statistics collected. A standard form will be used for

indexing the output reports, see Figure III-12. The uppermost box contains all of the options for statistics that SAINT will collect automatically. Simply check the boxes next to the options used for the system module. These options are divided into two sections: iteration statistics and summary statistics. The iteration statistics are printed out at the end of each iteration for a specified range of iterations. (These options and start and end iterations are indicated on the OUT card, see reference [2], pages 42-43.) The summary statistics are averaged over all iterations and printed out at the end of the simulation. For each option checked, enter a section number in the right-hand column. The section number can be assigned like a figure number, and will represent the location within this section of the indicated type of report. For example, in Figure III-12, the output for all Resource Utilization Statistics for a single iteration would be found in Section VII-1.

The bottom half of the form in Figure III-12 will be used to index all user statistic sample outputs. The statistic name, number, and whether the statistic is an iteration or summary statistic, is entered in the left-hand column. In the right-hand column the section number where the sample output is located is entered, just as it was done for the SAINT statistics.

Each sample output with explanation will be included in the order indicated by the index. For iteration statistics collected for more than one iteration, only one end of iteration report need be included.

SAINT OUTPUT INDEX

SAINT OUTPUT REPORTS		Section No.
ITERATION STATISTICS	☐ Detailed Iteration Reports	
	☒ Resource Utilization Statistics	VII-1
	☒ Task Statistics	VII-2
	☐ State Variable Value Outputs	
	☐ State Variable Statistics	
	☐ State Variable Plots/Tables	
SUMMARY STATISTICS	☒ Resource Utilization Summary Report	VII-3
	☒ Statistics Task Summary Report	VII-4
	☒ Histograms of Task Statistics, Iteration 1	VII-5
	☐ Histograms of Task Statistics, Summary	

USER STATISTICS OUTPUT REPORTS	SECTION NO.
1. Observation Statistics 1, Summary	VII-6
2. Plot of SS(3),Plot No. 1	VII-7
3. Histogram of Statistic 2, Iteration	VII-8

Figure III-12. Form for Indexing the SAINT Output Report Samples.

(Blank Page)

IV. BLANK FORMS

This section contains a reduced copy of the following blank forms.

- SAINT ENTITIES
- SAINT RESOURCES
- SAINT VARIABLES
- SAINT MONITORS
- SAINT TASK DESCRIPTIONS
- SAINT STATISTICS
- SAINT USER FUNCTIONS
- SAINT USER-WRITTEN SUBPROGRAMS
- SAINT OUTPUT INDEX

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	

NAME:

DATE:

AIR DEFENSE SYSTEM MODULE:

PROJECT:

SAINT ENTITIES

PAGE___OF___

RESOURCE NUMBER	RESOURCE LABEL	DESCRIPTION OF USE	RESOURCE ATTRIBUTES			TASK NUMBERS REQUIRING THE RESOURCE
			ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE	

NAME:

DATE:

AIR DEFENSE SYSTEM MODULE:

PROJECT

SAINT RESOURCES PAGE___OF___

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITIAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)

TYPE OF VARIABLES; ☐ STATE VARIABLES ☐ SYSTEM ATTRIBUTES ☐ USER VARIABLES

NAME: AIR DEFENSE SYSTEM MODULE:

DATE: PROJECT:

SAINT VARIABLES PAGE___OF___

MONITOR NUMBER	MONITOR LABEL	USER OR SAINT MONITOR	MEANING	CROSSING CONDITIONS	ACTION TO BE TAKEN	
					CODE	MEANING

NAME:

DATE:

AIR DEFENSE SYSTEM MODULE:

PROJECT:

SAINT MONITORS

PAGE___OF___

TASK NUMBER	DESCRIPTION (INCLUDE BRANCHING INFORMATION)	RESOURCE REQUIRMENTS		DESCRIPTORS		TECHNICAL REF.
		RESOURCE NUMBER	DESCRIPTION	CODE	MEANING	

NAME:	AIR DEFENSE SYSTEM MODULE:
DATE:	PROJECT:

SAINT TASK DESCRIPTION PAGE___OF___

TYPE OF STATISTIC	STATISTIC NUMBER	STATISTIC NAME	DESCRIPTION	HOW COLLECTED AND REPORTED

NAME:	AIR DEFENSE SYSTEM MODULE:
DATE:	PROJECT:

SAINT USER STATISTICS PAGE___OF___

USER FUNCTION NUMBER	TASK NUMBER(S) WHERE CALLED (USERF ONLY)	DESCRIPTION

WHICH USER FUNCTION? ☐ USERIN ☐ USERF

NAME: AIR DEFENSE SYSTEM MODULE:

DATE: PROJECT:

SAINT USER FUNCTIONS PAGE___OF___

MODERATOR NUMBER	TASK NUMBER(S) WHERE USED	DESCRIPTION	REFERENCE

NAME: DATE:	AIR DEFENSE SYSTEM MODULE: PROJECT:

SAINT MODERATOR FUNCTIONS PAGE___OF___

SAINT USER-WRITTEN SUBPROGRAMS

Check all Standard User-Written subprograms that are used in this model. Then add at the bottom the names of other user-written subprograms.

STANDARD USER-WRITTEN SUBPROGRAMS
☐ SUBROUTINE ENDIT
☐ SUBROUTINE INTLC
☐ SUBROUTINE MODRF
☐ FUNCTION PRIOR
☐ SUBROUTINE STATE
☐ SUBROUTINE UACCPT
☐ SUBROUTINE UERR
☐ SUBROUTINE UINPT
☐ SUBROUTINE USCOND
☐ FUNCTION USERF
☐ FUNCTION USERIN
☐ SUBROUTINE UOTPT

ADDITIONAL USER-WRITTEN SUBPROGRAMS

SAINT OUTPUT INDEX

SAINT OUTPUT REPORTS		Section No.
ITERATION STATISTICS	☐ Detailed Iteration Reports ☐ Resource Utilization Statistics ☐ Task Statistics ☐ State Variable Value Outputs ☐ State Variable Statistics ☐ State Variable Plots/Tables	
SUMMARY STATISTICS	☐ Resource Utilization Summary Report ☐ Statistics Task Summary Report ☐ Histograms of Task Statistics, Iteration 1 ☐ Histograms of Task Statistics, Summary	

USER STATISTICS OUTPUT REPORTS	SECTION NO.

V. REFERENCES

1. Wortman, D. B., Duket, S. D., Seifert, D. J., Hann, R. L., & Chubb, G. P. Simulation using SAINT: a user-oriented instruction manual (AMRL-TR-77-61). Aerospace Medical Research Laboratory, Wright Patterson AFB, OH, July 1978.

2. Wortman, D. B., Duket, S. D., Seifert, D. J., Hann, R. L. & Chubb, G. P. The SAINT user's manual (AMRL-TR-77-62). Aerospace Medical Research Laboratory, Wright Patterson AFB, OH, June 1978.

3. Walker, J. L. & Polito, J. Program and user documentation for modifications to the SAINT simulation language in support of MOPADS operator modules (MOPADS Vol. 4.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, July 31, 1982.

(Blank Page)

VI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

(Blank Page)

VII. CHANGE NOTICES

APPENDIX G

MOPADS FINAL REPORT:

DEVELOPMENT METHODOLOGY FOR MOPADS AIR DEFENSE

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AD	Air Defense
CPM	Critical Path Method
DTD	Directorate of Training Development
GCV	Ground Controlled Vehicle
JADC	Joint Air Defense Commander
LOAP	List of Applicable Publications
PERT	Program Evaluation Review Techniques
TDECC	Tactical Display and Engagement Control Console
TEWA	Threat Evaluation and Weapons Assignment

STANDARD MOPADS TERMINOLOGY

AIR DEFENSE SYSTEM

A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IH4WK and the AN/TSQ-73.

AIR DEFENSE SYSTEM MODULE

Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.

AIR SCENARIO

A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.

BRANCHING

A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.

DATA BASE CONTROL SYSTEM

That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.

DATA SOURCE SPECIALIST

A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE

An element of an Environmental State Vector.

I. INTRODUCTION

A major part of the MOPADS software will be the individual system modules. These modules will be SAINT models of various air defense systems, such as the AN/TSQ-73, which together form air defense configurations. As new equipment is developed, new system modules will be added to the MOPADS software. The methodology for developing these system modules is provided here.

System module development will involve two major types of activities: 1) SAINT modeling, and 2) data collection. The data collection activities will involve locating and obtaining documents, identifying pertinent information within those documents, locating subject matter experts and experienced operators, and getting pertinent information from those experts. SAINT modeling will involve taking the pertinent information and using it to create a SAINT model. The SAINT model, with its accompanying parameters and interfaces to the MOPADS software system, is referred to as an air defense system module. The modeling activities will be performed by one or more SAINT modelers with Operations Research experience and expertise. It is expected that the MOPADS modeler(s) will receive help in performing the data collection activities from a Data Source Specialist. This person (or persons) must be experienced in Army documentation and air defense systems.

A CPM (or PERT) chart, Figure II-1, was developed to show the breakdown of activities and precedence relationships. The data collection activities are distinguished from the SAINT modeling activities by using dashed lines for the data collection activities. Figure II-1 is described in detail in Section II.

In Section III, each task shown on the CPM chart is described in further detail. Blank forms to be used in performing some of these tasks are given in Section IV. The SAINT modeler should collect and compile these forms and keep them in a notebook. All forms of a given type should be kept together, and the groups of forms should be ordered in the same order as the forms are presented in Section IV.

(Blank Page)

II. CPM CHART

1-0 DESCRIPTION

The CPM (Critical Path Method) chart for system module development is shown in Figure II-1. It is a project management chart showing all of the activities on the lines of the network. The nodes (circles) represent milestones in the project. The flow of the network goes from left to right. Every activity (represented by a line) going into a node must be completed before any activities coming out of the node can be started. If more than one activity enters a node, the last activity completed will determine the start time of any outgoing activities. The longest path through the network is referred to as the critical path. Any delays in activities along the critical path result in delays in the project, whereas delays in non-critical activities may or may not delay the project completion. The bold solid (or dotted) line in Figure II-1 represents the expected critical path. Individual circumstances or manpower shortages may change the critical path, but the activity precedence relationships would still hold. Dotted lines represent data collection activities, while solid lines represent SAINT modeling activities.

Each activity on the CPM network is given a two part identifying number. The "1." numbers represent SAINT modeling activities, while the "2." numbers represent data collection activities. Each activity is explained in detail in Section III.

The activities contained in the CPM chart are divided into four groups. These are shown with the large braces along the bottom of the chart in Figure II-1. The activity groupings are defined below.

1-1. Operator and Equipment Definition. During this phase of the project, the operator duties and equipment requirements are characterized.

1-2. Task Definition. A task is defined as a sequence of task elements (such as pushing a button) which accomplishes some purpose. During this phase of the project, the tasks performed during normal operations of the equipment are identified, listed, and characterized.

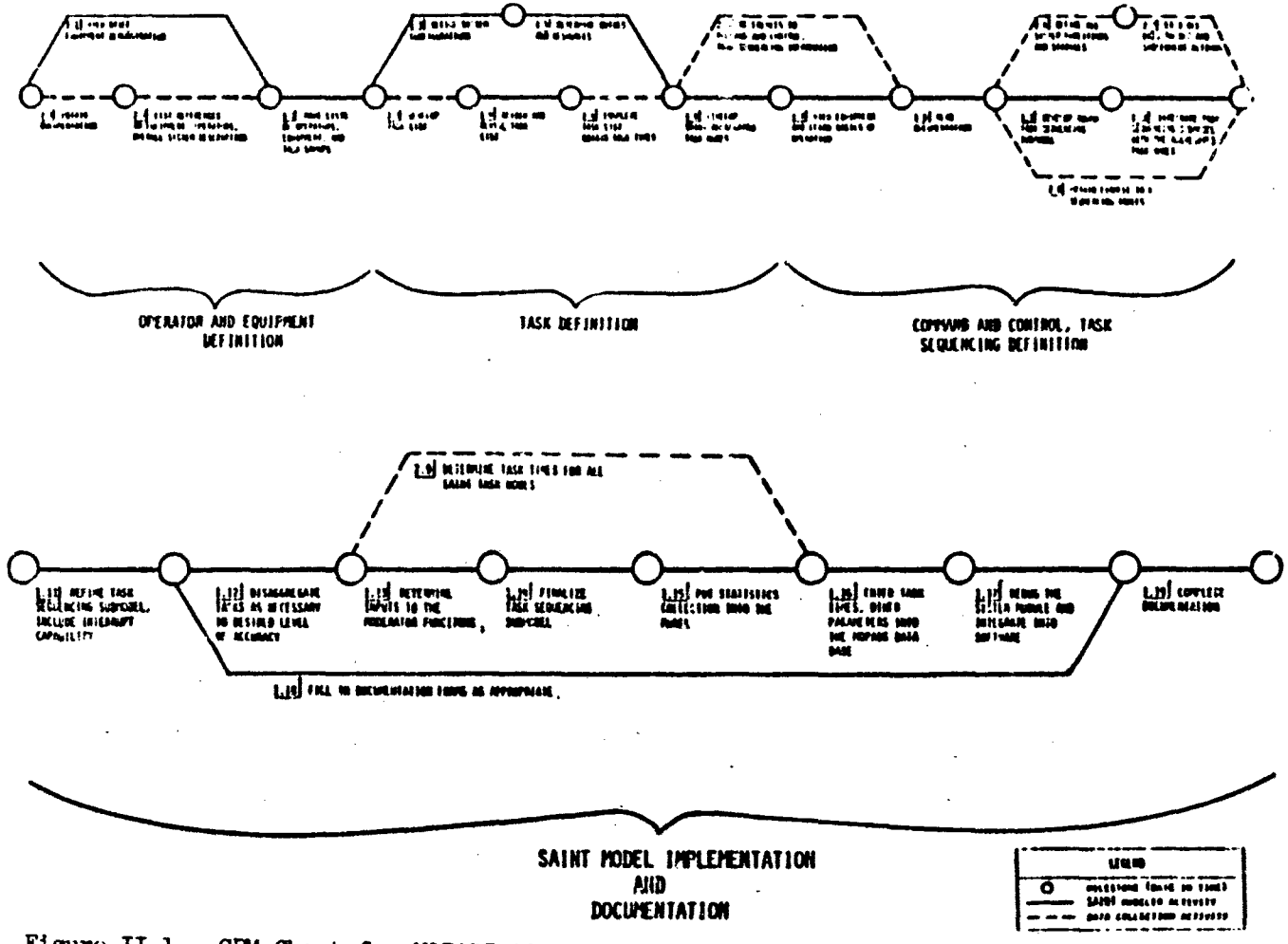


Figure II-1. CPM Chart for MOPADS Air Defense Module Development.

1-3. Command and Control and Task Sequencing Definition. During this phase of the project, the decision processes for sequencing tasks are characterized. This involves understanding the operators' interaction with other air defense components.

1-4. SAINT Model Implementation and Documentation. The task sequencing submodel and task performance models will be disaggregated to the desired level of fidelity. The model will be integrated with the MOPADS software and required documentation will be completed.

III. ACTIVITY DESCRIPTIONS

1-0 SAINT MODELING ACTIVITIES.

1-1. Operator and Equipment Definition.

Activity 1.1 As early as possible the MOPADS modeler should receive a brief demonstation of the equipment operation. The modeler need not develop a working understanding of the operation at this state. This demonstration's purpose is to give the modeler a general understanding of what the equipment does and how it is operated.

Activity 1.2 The SAINT modeler will have a list of references to documentation describing the overall system, operator responsibilities, and equipment. From this documentation the SAINT modeler will construct three lists.

1. The first list will contain all equipment components that are essential for the operators to perform their tasks. List equipment which may have a status such as "operational," "partially operational," or "non-operational." If a piece of equipment may degrade or break down, and this would have an effect on the operator task performance, then that piece of equipment should be included in this list. The equipment list will be used to assign SAINT resources and to characterize operator tasks. No standard form has been developed for listing pertinent equipment. The list will include the component name, accompanied by a short description of its function.

2. The second list will identify the operators of the air defense system. A form has been created to list the operators, and a sample of this form is shown in Figure III-1. Each operator will be assigned an identification number which is entered in the first column. The operator title is entered in the second column. The mission and duties of each operator are described in the third column, and the equipment used by each operator in performing his duties is listed in the fourth column. These equipment designations should correspond to those in the above mentioned equipment list.

Sometimes an air defense system may have different configurations of operators such as two officers or one officer and one enlisted person. In this case a decision must be made concerning the number of options to be modeled. A standard or most likely configuration may be selected, or multiple configurations with user specified options may be selected. In the latter case it may be required to develop several sets of the documents described subsequently, one for each option.

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
1	Tactical Officer	Target acquisition, identification, evaluation, assignment, tracking, and firing. He makes decisions regarding these missions and either performs them himself or assigns them to another operator.	TDECC Panel Status Panel (located directly above the TDECC)

NAME: J. L. Walker	AIR DEFENSE SYSTEM MODULE: IHAWK Module
DATE: 3/17/82	PROJECT: MOPADS

OPERATOR DEFINITIONS PAGE 1 OF 1

Figure III-1. Sample OPERATOR DEFINITIONS Form.

3. The third list to be produced is a list of categories for all the tasks performed by each operator. Some typical categories are tracking, target acquisition, TEWA, and target identification. The number of these categories will normally be less than ten. Some tasks may be placed under more than one category. It is not necessary to assign tasks to categories at this time; the task categories need only be defined here.

The documentation on these subjects may not be complete enough to allow the MOPADS modeler to successfuly complete this activity. If so the modeler may want to consult subject matter experts or experienced operators to obtain more complete information. It may be desirable to consult such experts even if the documentation is considered thorough. In this way, the modeler may obtain additional confidence in the lists produced during this activity.

1-2. Task Definition.

Activity 1.3 There may be more than one way that the system can be configured. In this case, those configurations to be modeled must be selected. A diagram showing the chosen configuration and how it interfaces with the overall AD system should be drawn for later reference. If more than one configuration is selected, it may be required to develop multiple models. A decision must be made as to whether it is more economical to model the system as a single module or as separate modules.

Activity 1.4 At this point, a list of tasks will have been completed. This list will give references to the documentation, the operator performing the task, and which group the task is in. The MOPADS modeler will carefully examine this list to determine if any tasks have been left out or if some of the tasks may be removed from the list.

Activity 1.5 Based on the operator and equipment definitions, the MOPADS modeler will decide whether to model each operator as an entity or resource, and which equipment to model explicitly as resources. The forms shown in Figures III-2 and III-3 are filled out at this time [1].

Activity 1.6 The task list will have been finalized and nominal task times will have been determined. Based on the task list and the entity and resource definitions, the modeler will develop aggregate SAINT models(perhaps single task nodes) for each task. Final statistics, priorities, etc. will not be needed at this time.

A standard form is available for creating these aggregated models. A sample is shown in Figure III-4. The information on this form is taken from the task list which is completed during Activities 2.4 and 2.5. No branching or precedence relationships to and from the task will be included at this time.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
2	CWTDC Operator	1	Rank	3.0	2,CWTDC panel
		2	Skill level	7.0	5,ADP
		3	Duty time span	2.0	9,IHIPIR
5	TCC Operator	1	Rank	4.0	3,TCC panel
		2	Skill level	5.0	5,ADP
		3	Duty time span	2.5	8,ATDL-1

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULES: HAWK BCC
PROJECT: MOPADS

SAINT ENTITIES PAGE 1 OF 1

Figure III-2. Sample ENTITIES Form.

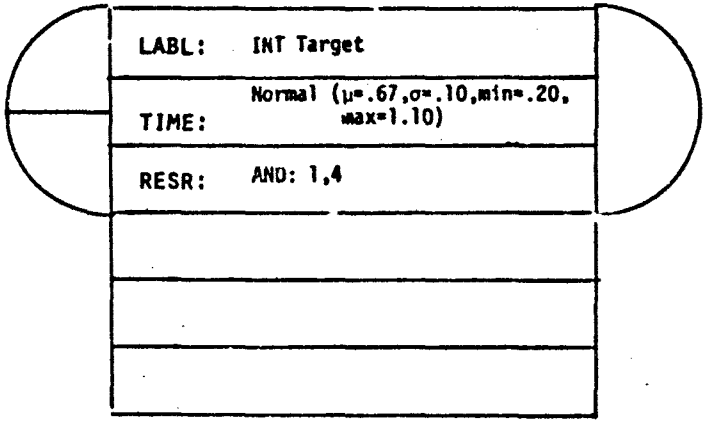
RESOURCE NUMBER	RESOURCE LABEL	DESCRIPTION OF USE	RESOURCE ATTRIBUTES			TASK NUMBERS REQUIRING THE RESOURCE
			ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE	
5	ADP System	Whenever an operator requests information from the ADP system, he can get it only if the ADP is operational and not busy processing other information.	1	Speed of elemental calculation	.0001	57,58,71,91,103
			2	Mean time between breakdowns	1.3	
			3	Number of connections	57	
1	AF Command	Whenever a group Q73 requests a directive from higher authority, they can get that directive only when the command center is not busy.	None			61,62,153

NAME: J. L. Walker
DATE: 2/19/82

AIR DEFENSE SYSTEM MODULE: Group Q73
PROJECT : MOPADS

SAINT RESOURCES PAGE 1 OF 2

Figure III-3. Sample RESOURCES Form.



TASK NAME AND DESCRIPTION	OPERATOR	REFERENCE(S)
Interrogate a target or sector.	Either console operator	TM 9-1430-652-10-3 (1978, Change 6) Figure 4-39
NAME: J.L. Walker DATE: 3/17/82	AIR DEFENSE SYSTEM MODULE: MODEL Q-73 PROJECT: MOPADS	
AGGREGATE SAINT TASK MODELS		PAGE 39 OF 70

Figure III-4. Sample TASK MODELS Form.

1-3. Command and Control and Task Sequencing Definition.

Activity 1.7 Once the operator tasks have been identified, the MOPADS modeler needs to view the equipment in operation. The modeler should obtain an understanding of the operators' task sequencing decisions and their communication procedures.

Activity 1.8 The MOPADS modeler will examine appropriate sections of the documentation in order to better understand how the operators go from one task to another. The modeler should understand how the operators sequence their tasks in order to maximize their goal (or mission) achievement.

Activity 1.9 It is expected that each of the system modules will have a similar structure. A schematic network of this basic structure is shown in Figure III-5. The operator arrives at the task sequencing submodel. This may be only one task node or a series of task nodes. The purpose of the task sequencing submodel is to determine what task the operator performs next. The operator branches to the appropriate task node (or sequence of task nodes), then returns to the task sequencing submodel to repeat the process.

During Activity 1.9 the modeler will develop a preliminary version of the task sequencing submodel.

Activity 1.10 The modeler will integrate the decision submodel with the aggregate task models to form a basic SAINT model. This aggregate model will be run to verify the logic contained in the task sequencing submodel.

1-4. SAINT Model Implementation and Documentation.

Activity 1.11 At this point, the MOPADS modeler will have a large amount of information on system status', system conditions, trial drills, alerts, messages, and interrupting and non-interrupting cues. The task sequencing submodel will be expanded to incorporate interrupt features so the module can ultimately be integrated with the command and control elements of the MOPADS software.

Activity 1.12 At this point, the task sequencing portion of the SAINT network will be well defined, but the task models will still be aggregated. Many of these task models will be expanded into subnetworks of task nodes representing the flowchart for that task. This subnetwork will normally have only one input and one output branch (to and from the task sequencing submodel), but it may have complex branching within it.

It is expected that not all aggregated task models will require disaggregation. Some modeler judgement will be required here to determine which tasks to disaggregate and how detailed a breakdown is necessary.

This activity is expected to be the most time consuming of all activities in the system module developemnt process.

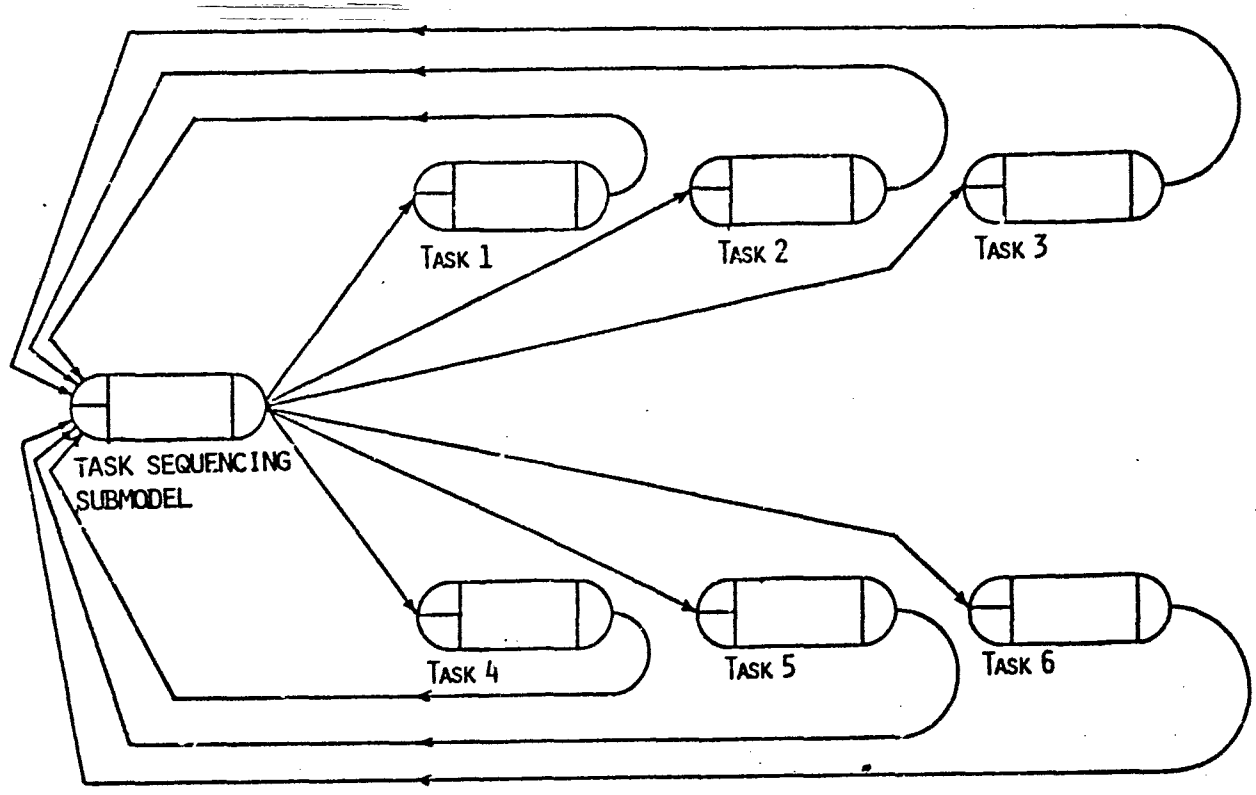


Figure III-5. MOPADS System Module Structure.

Activity 1.13 At this point, the SAINT network for the module will have been determined. Now the MOPADS modeler must address the human factors aspects of the model. A generalized moderator function will be provided as part of the MOPADS software which requires as input a list of skills necessary to perform each task or task element and the relative weights for the skills. The modeler must provide this data for each task node.

Activity 1.14 The task sequencing submodel will have associated moderators which require parameters that determine the operators' goal seeking procedures. These will be determined and entered into the MOPADS data base. This will complete the development of the task sequencing submodel.

Activity 1.15 Most output statistics will be optional and will be obtained through the MOPADS data base. However, it is required that certain statistics be specified on the SAINT network. These standard statistics will be entered into the model base. They may be SAINT task statistics or user-created statistics.

Activity 1.16 The MOPADS modeler will have obtained a complete list of task times for all SAINT task nodes in the network. These will be entered into the MOPADS data base for reference by all SAINT task nodes. Any other required parameters will be entered into the MOPADS data base.

Activity 1.17 The system module, now in its final form, is ready for final testing and debugging. This will be done by running the model alone with the MOPADS software but no other system modules, if this is possible. If not, the modeler will run the new system module interfaced with a likely configuration containing other system modules.

Activity 1.18 In reference 1, the documentation requirements for MOPADS system modules are outlined and explained. Several forms are provided for the modeler to complete while the module is being developed which will become part of the documentation. This "documentation during development" approach will ensure completeness and reduce the end-of-project documentation chores.

Activity 1.19 Much of the documentation has been completed while the module was being developed. The only remaining activity is to complete the documentation and put it in its final form per the instructions contained in reference 1.

2-0 DATA COLLECTION TASKS.

2-1. Operator and Equipment Definition.

Activity 2.1 Once the decision has been made as to what equipment is to be modeled, pertinent documentation describing the operation will be obtained. This may include Field Manuals, Soldier's Manuals, Technical Manuals, and any other documents that may exist. LOAPs may be used in locating pertinent document titles.

Activity 2.2 A quick look through the documentation will be made and all references to discussions of the equipment and individual responsibilities of the operators should be noted. A list will be made giving ranges of page numbers for later reference. Also, references to discussions of the overall system will be listed. Any references to task groupings should be especially noted.

2-2. Task Definition.

Activity 2.3 At this time, the form shown in Figure III-1 will have been completed so the operators to be modeled will have been identified. Also, a list of task groups and a list of equipment the operators will run will have been developed. Once this information is complete, the task list can be developed. In order to develop this task list, the form shown in Figure III-6 is provided. A good place to start is in the corresponding Soldier's Manuals for the appropriate operators. More accurate task information may be obtain from the DTD.

An example listing of a task is shown in Figure III-6. A task number must be assigned to each task, and this is placed in the first column. The task name is entered in the second column. All references to descriptions and/or flowcharts for the task are listed in the third column. These would normally include a reference to the Soldier's Manual and/or a reference to the Technical Manual. All references should give the date of the publication and the latest change number.

Information in the fourth, fifth, and sixth columns will reference the operator, equipment, and task grouping lists which were developed during Activity 1.2 Information necessary to fill in these three columns should come from the documentation or from subject matter experts at DTD. Experienced operators may also be consulted.

The far right column is left blank during the completion of Activity 2.3, unless the information is easily attainable at this time.

Activity 2.4 The task list will have been reviewed and revised to assure that no critical tasks were left out and to remove those tasks which will not be included in the system module. Now, the task list is finalized.

TASK NUMBER	TASK NAME	REFERENCE	OPERATOR(S)	EQUIPMENT	TASK CATEGORY	TASK TIME
10	Assign weapons	TM 9-1430-652-10-3(1978, Change 6) Figure 4-10	Officer at the console	Display console, TACK SELECTIONS and TASK FUNCTIONS buttons	TEWA	Normal μ = 1.1 σ = .25 min = .50 max = 1.6

NAME: J. L. Walker
DATE: 3/17/82

AIR DEFENSE SYSTEM MODULE: Q-73
PROJECT: MOPADS

MOPADS TASK DEFINITIONS

PAGE 7 OF 20

Figure III-6, Sample TASK DEFINITIONS Form.

Once the task list contains all appropriate tasks, the time estimates are added in the far right column of the task description forms. This time estimate will be obtained from equipment drills, expert estimation, or DTD documentation. These time estimates may be constants (i.e., 5 minutes) or distributions with parameters (i.e., Normal Distribution, $\mu = .5$, $\sigma = .15$, min = .2, max = .8).

Activity 2.5 A list will be made of all references to sections in the documentation describing command and control systems and procedures. Also, discussions of how the operators function in the command and control system should be referenced. This list will give ranges of page numbers for sections in the documentation describing these topics. Any discussions of how the operators sequence their tasks should be referenced.

2-3. Command and Control and Task Sequencing Definition.

Activity 2.6 All system conditions will be defined using the form shown in Figure III-7. A system condition is a state of alert, warning status, readiness condition, or rule of engagement that has more than one code or status value. They are normally set by higher levels of command and communicated to the appropriate air defense systems. The condition name and definition are entered in the first two columns. In the third and fourth columns, each code value is listed and defined. In the fifth column, the person or command center responsible for determining the current code value is listed. In the last column, the way in which the operators are made aware of code changes for the alert is described.

Also, as a part of this task all system status conditions are defined. A status may be the fire unit status, the condition of certain equipment, or the availability of a resource. A form has been developed to perform the task of defining status, and an example is shown in Figure III-8. In the first column, the equipment for which the status applies is entered. This may be general such as the whole IHAWK system or more specific such as a TDECC panel or launcher. The name of the status and definition are entered in the second and third column. Each possible value (or code) is entered in the fourth column along with its meaning in the fifth column. If the status value is simply a number representing a level of some resource, this should be noted here. The way in which status values are communicated is described in the last column. This communication may be either upward or downward in the chain of command.

Activity 2.7 The operation of air defense equipment usually involves waiting for various messages or cues to act and then taking appropriate action. Certain symbols, codes, and messages may appear on a radar tracking screen, and these would be considered cues to act. Some cues require immediate action while other cues do not. These are referred to as interrupting and non-interrupting cues, respectively. Cues will be listed using the form shown in Figure III-9.

CON-DITION NAME	DEFINITION	CODE VALUE	MEANING	RESPONSIBLE AUTHORITY	NOTIFICATION METHOD FOR CODE CHANGES
AIR DEFENSE WARNING	Evaluation of the air threat	RED	Attack imminent or in progress	Joint Air Defense Defense Commander (JADC)	Light changes on the Status portion in the upper right hand corner of the TDECC panel. Confirming message may be transmitted verbally.
		YELLOW	Attack is probable		
		WHITE	Attack not probable or imminent		

NAME: J. L. Walker
DATE: 3/17/82

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SYSTEM CONDITIONS PAGE 1 OF 2

Figure III-7. Sample SYSTEM CONDITIONS Form.

EQUIPMENT	STATUS NAME	STATUS DEFINITION	STATUS VALUE	MEANING	METHOD OF COMMUNICATIONS STATUS VALUE CHANGES
IHAWK	Launcher Readiness Status	Number of missiles ready to fire	0 to 3	This number represents the number of missiles that are mounted onto the launcher and ready to be fired. If some missiles are mounted but the launcher is not operational, the status is 0	

NAME: J. L. Walker	AIR DEFENSE SYSTEM MODULES: IHAWK
DATE: 3/17/82	PROJECT: MOPADS

SYSTEM STATUS PAGE 1 OF 5

Figure III-8. Sample SYSTEM STATUS Form.

OPERATOR	CUE DESCRIPTION	INTERRUPT OR NON-INTERRUPT (I OR N)	SUBSEQUENT TASK SEQUENCE	DESCRIPTION OF PRIORITY
Tactical officer at the TDECC	Symbol ⌽ begins	I	Check weapons control status, if not known in memory. Request permission to fire from battalion Q73, if permission needed or track not already assigned. Then assign the track to a fire control operator.	Unless currently handling a flashing ⌽ symbol, drop the current activity. If currently handling a ⌽ symbol, finish that and immediately handle the flashing ⌽ symbol.

NAME: J. L. Walker	AIR DEFENSE SYSTEM MODULE: IHAWK
DATE: 3/17/82	PROJECT: MOPADS

OPERATOR CUES TO ACT PAGE 1 OF 3

Figure III-9. Sample CUES TO ACT Form.

The operator responding to the cue is entered in the first column. A description of the cue is entered in the second column. In the third column, an I or N is entered to designate it as an interrupting or non-interrupting cue. The subsequent actions performed to handle the situation are described and/or listed in the fourth column. This would normally be a reference to one or more tasks in the task list. In the far right column, the priority for subsequent action is described. This may include such information as which tasks would be pre-empted by this action and which tasks would not be pre-empted.

Three possible approaches may be taken in obtaining the information necessary to fill out the form shown in Figure III-9

1. A Technical Manual may exist for the air defense system containing alert definitions. For example, see [2] for the AN/TSQ-73. If so, this would be a good place to start obtaining information on operator cues to act.

2. The forms shown in Figure III-9 is used to list only the cues in the second column. The rest of the form is left blank at first. The cues could be obtained from lists of symbols, alerts, or messages as contained in the documentation, or from subject matter experts. This list of cues is given to an experienced operator or subject matter expert to fill in the subsequent actions for each cue to act. The system module task list should be given to them so the subsequent actions column may also be filled in.

3. The fourth column of the form shown in Figure III-9 may be filled in with each task name from the task list. With only column four filled out, the forms are then given to a subject matter expert or experienced operator, who would fill in the corresponding cue (or cues) to act for each task listed.

A combination of these approaches may be used in performing Activity 2.7.

Activity 2.8 Examples outlining every task that was performed during equipment drills should be obtained. If these are not available, references to documentation of example scenarios should be found. These should show actual task sequencing performed by experienced operators.

2-4. SAINT Model Implementation and Documentation.

Activity 2.9 Nominal times for each task element (SAINT task node) are determined. This data may be obtained from equipment drills, subject matter experts, or computerized scenario traces. It may be necessary to obtain a consensus on the task times from several different sources.

IV. BLANK FORMS

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT

NAME:	AIR DEFENSE SYSTEM MODULE:
DATE:	PROJECT:

OPERATOR DEFINITIONS. PAGE___OF___

TASK NUMBER	TASK NAME	REFERENCE	OPERATOR(S)	EQUIPMENT	TASK CATEGORY	TASK TIME

NAME:	AIR DEFENSE SYSTEM MODULES:
DATE:	PROJECT:

MOPADS TASK DEFINITIONS PAGE___OF___

TASK NAME AND DESCRIPTION	OPERATOR	REFERENCE(S)
NAME: DATE:	AIR DEFENSE SYSTEM MODULE: PROJECT:	
AGGREGATE SAINT TASK MODELS		PAGE___OF___

CON-DITION NAME	DEFINITION	CODE VALUE	MEANING	RESPONSIBLE AUTHORITY	NOTIFICATION METHOD FOR CODE CHANGES

NAME:	AIR DEFENSE SYSTEM MODULES:
DATE:	PROJECT:

SYSTEM CONDITIONS PAGE___OF___

EQUIPMENT	STATUS NAME	STATUS DEFINITION	STATUS VALUE	MEANING	METHOD OF COMMUNICATIONS STATUS VALUE CHANGES
NAME:				AIR DEFENSE SYSTEM MODULES:	
DATE:				PROJECT:	
SYSTEM STATUS					PAGE__OF___

OPERATOR	CUE DESCRIPTION	INTERRUPT OR NON-INTERRUPT (I OR N)	SUBSEQUENT TASK SEQUENCE	DESCRIPTION OF PRIORITY

NAME:	AIR DEFENSE SYSTEM MODULES:
DATE:	PROJECT:

OPERATOR CUES TO ACT PAGE___OF___

V. REFERENCES

1. Walker, J. L. & Polito, J. Documentation requirements and development guidelines for MOPADS air defense system modules (MOPADS Vol. 4.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, April 15, 1982.

2. Technical manual: operators's manual: displays, ARO, and alert definitions, guided missile air defense system AN/TSQ-73 (TM 9-1430-652-10-1). Headquarters, Department of the Army, October 5, 1978, Change 0.

VI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACC-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VII. CHANGE NOTICES

APPENDIX H

MOPADS FINAL REPORT:

MSAINT USER'S GUIDE: CHANGES AND ADDITIONS
TO THE SAINT USER'S MANUAL

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1-0 STANDARD MOPADS TERMINOLOGY

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system The standard version of SAINT has been modifie for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAIN and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

2-0 OTHER TERMINOLOGY

AIRCRAFT CHECKPOINTS	The aircraft paths in MOPADS air scenarios are assumed to be piecewise linear paths. The points where the aircraft speed and/or heading change are called checkpoints.
ASO	Azimuth Speed Operator - the operator in the IHAWK system responsible for detecting low flying approaching aircraft.
BCC	Battery Control Central - the command center for an IHAWK fire unit.
COPY	The version of a system module being used to simulate an individual air defense component.
DB	Data Base (MOPADS)
DBAP	Data Base Application Program
ENGAGEMENT MARKERS	A mark on the screen of a Q-73 panel to show where hostile aircraft are being engaged.

ENTITY	Also called a transaction, a simulation unit which flows through a network of task nodes and branches.
ESV	Environmental State Vector
FCO	Firing Console Operator - the operator in the IHAWK system responsible for obtaining a radar lock on a hostile aircraft and pushing the fire button.
FIRE UNIT	A ground based air defense unit which has fire power capable of destroying hostile aircraft.
HOOKED ITEM	At a Q-73 panel, a track or air defense unit may be hooked. Hooked items have a special section of alphanumeric data describing them. Also items must be hooked in order to send communication messages concerning them.
IA	Information Attribute - a variable amongst a vector of values (called the IA vector) which follow an entity through a simulation network.
IFF	Identification Friend or Foe.
OSV	Operator State Vector
PAIRING LINES	Lines on a Q-73 screen representing which fire unit is assigned to which target (hostile aircraft).
Q-73	Short notation for the AN/TSQ-73 missile minder.

RA	Resource Attribute - a variable amongst a group of variables which follow a resource through a simulation network.
RUN	Also called an iteration. A complete simulation from start time through ending time. Statistics are summarized over each run. Multiple runs can be made.
SA	System Attribute - a variable which is "global" to a given copy. Each copy has a vector of SA variables.
SC	Scaled Constant - now it has a constant value of 1.0.
SCREEN ALPHANUMERICS	A Q-73 display option which allows the display of 10 coded characters adjacent to each symbol on the screen.
SITE	A Q-73 unit or a protected area (such as an airfield or command center).
SM	System Module - see Air Defense System Module in the MOPADS terminology.
TCA	Tactical Control Assistant - IHAWK operator responsible for IFF procedures and assisting the TCO.
TCO	Tactical Control Officer - Commanding Officer of an IHAWK BCC unit.
TD	Tactical Director - the officer in charge of a Q-73 unit.

TDA	Tactical Director Assistant - the Q-73 operator who assists the TD.
TRACK	A raw video mark or symbol on an air defense operator's screen which represents the location of an aircraft.
TTG VECTOR	A line emanating from a track on the Q-73 screen representing the location an aircraft will be at in a given time if it keeps the same speed and heading.
VELOCITY VECTOR	A line emanating from a track on a Q-73 screen representing the relative velocity of the track.

I. PURPOSE

The MOPADS framework requires a task event oriented simulation language to run MOPADS simulations. The SAINT simulation language was chosen for this purpose because it not only has a task event orientation, but it also has many other features useful to MOPADS, such as task clearing and moderator functions. However, several major changes and additions had to be made to SAINT to render it compatible with the MOPADS framework. A more flexible clearing methodology was implemented, many new user-callable utility routines were added, and a new method of allowing multiple air defense units to be simulated using only one network (called shareable networks) was implemented. Several SAINT options were nullified by the new design constraints, but these options were not necessary or appropriate for MOPADS. The new version of SAINT, enhanced exclusively for MOPADS, is now called MSAINT.

This document describes the changes in the usage of SAINT. Instructions and definitions for all additional capabilities are also included. It is assumed that the reader is already familiar with the material in Wortman, Duket, Seifert, Hann & Chubb (1978a,b,c).

Section II of this document shows all user changes by section to Wortman, et al (1978a). For clarity and ease, reference Wortman et al (1978a) will be shown as Reference 1 so the reader may write the changes into his/her copy. Section III describes all of the additions to SAINT implemented in MSAINT.

II. CHANGES TO THE SAINT USER'S MANUAL

1-0 INTRODUCTION

Several changes have been made in the usage of existing features in SAINT. Since the use of these features is described in reference 1, this section will be broken down into subdivisions based on changes to individual sections in reference 1. This section is somewhat cryptic. However, Section III contains more expansion on the specific modifications for MSAINT.

2-0 CHANGES TO SECTION II (REFERENCE 1)

Page 13

The following task description codes **can no longer be used:** DMOD, REGL, SWIT.

Page 14-20

Ignore the descriptions of the usage of DMOD, REGL, and SWIT cards.

Pages 27-28

Task Modification **is no longer allowed.**

Pages 29-31

Monitors **can no longer be used** because appropriate SS indices cannot be predicted at run time. This is because the tracks only temporarily use SS variables. The MONF, MTAS, and MSWT designators can no longer be used.

3-0 CHANGES TO SECTION III (REFERENCE 1)

Pages 33-35 - Table 5

The following cards **can no longer be used:**

SGE, OUT, UPL, NMO, DMO, SWI, REG,
MON, MTA, MSW, SST, PLO, VAR.

The SIM and FIN cards have **new definitions** shown below:

FIN - Signals the end of a system module data input deck

SIM - Signals the end of all decks, thus it is the last card in the deck file.

Page 37

Change 9.a and b to the following and delete 9.c.

a. GEN, POP, and DIS cards must be input before any other card type.

b. FIN cards follow each system module deck. A SIM card signifies the end of the data input decks.

Page 39

Replace with Figure II-1.

Page 40

Disregard this page.

Page 41

Replace with Figure II-2.

Pages 42-43

Disregard these two pages.

GEN - General System Module Information

Field	Value	Default	Description
1	A	--	Card Identification (GEN)
2	A	SAINT	System Module Author Name (max.of 8 characters)
3	I	1	Month of SM Completion
4	I	1	Day of SM Completion
5	I	2001	Year of SM Completion
6	I	0	System Module Type =1 Control =2 Group Q-73 =3 Battalion Q-73 =4 IHAWK

Figure II-1. New GEN Card Description.

POP - Program Options for This System Module

Field	Value	Default	Description
1	A	--	Card Identification (POP)
2	I	0	Largest resource number in this SM
3	I	0	Number of RA's per resource in this SM
4	I	2	Number of IA's per information packet in this SM
5	I	0	Number of SA's in this SM
6	I	1	Maximum moderator function number used by this SM

Figure II-2. New POP Card Description.

Page 44

Note that distribution sets are renumbered starting with 1 for each system module. These distribution sets are used only for tasks which do not have a task time and weighted skill list in the MOPADS DB.

Pages 47-49

User observation, time persistent, and histogram statistics are all cleared at the beginning of each run. Therefore, they are only used for statistics collected and reported at the end of each run. User statistics of each type are renumbered starting with 1 for each system module.

Pages 50-53

It is strongly suggested that UPL and VAR cards not be used. The meaning of any plots would be hard to determine because each SS variable is used for an x, y, or z coordinate value of a track only for the time the track is in the system. When a track leaves the system, its SS variables are freed up to possibly be used by tracks that arrive to the system later.

Page 54

Moderator functions are not renumbered for each system module. However, each copy may have its own active or inactive status for each moderator function. The IMO card can be used to

set the initial status of each moderator function in each copy of the current system module to the same value.

Page 55

This card can be used to set all initial resource attribute values in each copy of the current system module to the same initial value. If certain copies have unique initial values, use subroutine GRESA to initialize these values (Polito (1983a)).

Note that UF designations in field 4 and subsequent function fields on this card result in calls to function USERIN, not USERF. See Section III, 3-0 for a description of the usage of USERIN.

If the SC function is specified, the scaled constant is always 1.

Page 56

Usage of the ISA card is similar to that for the IRA card. The ISA card will initialize all specified initial system attribute variables in all copies of the current system module to the same value. If certain copies have unique initial system attribute values, these are initialized using the DBAP routine GSYSA (Polito (1983a)).

Like the IRA card, the UF designations on the ISA card result in calls to function USERIN, not USERF. See section III, 3-0 for a discussion on the usage of function USERIN.

If the SC function is used, the scaled constant is always 1.

Page 58

Disregard this page.

Pages 60-61

If field 6 is SC, then the scaled constant used is always 1. Also note that a newer TAS card description is given in Wortman & O'Reilly (1982).

Page 64

Moderator functions themselves are not unique to each copy. However, each moderator function has a current status of active or inactive for each copy. It is suggested that all moderators be kept inactive and then activated only for a specific task node. Therefore, fields 4 and 5 (and subsequent status and duration fields) would normally be A and T respectively.

Page 69

A newer condition code list for conditional branching is given in Wortman & O'Reilly (1982).

Pages 72-73

Disregard these two pages.

Pages 76-85

Disregard these pages.

Pages 87-90

No updated version of Table 6 will be provided.

4-0 CHANGES TO SECTION IV (REFERENCE 1)

Page 92

1. Subroutine INTLC is not used because the analyst cannot initialize SS variables. SS variables are used for tracks and are set internally from input in the air scenario.

2. The variable PFIRB is no longer used.

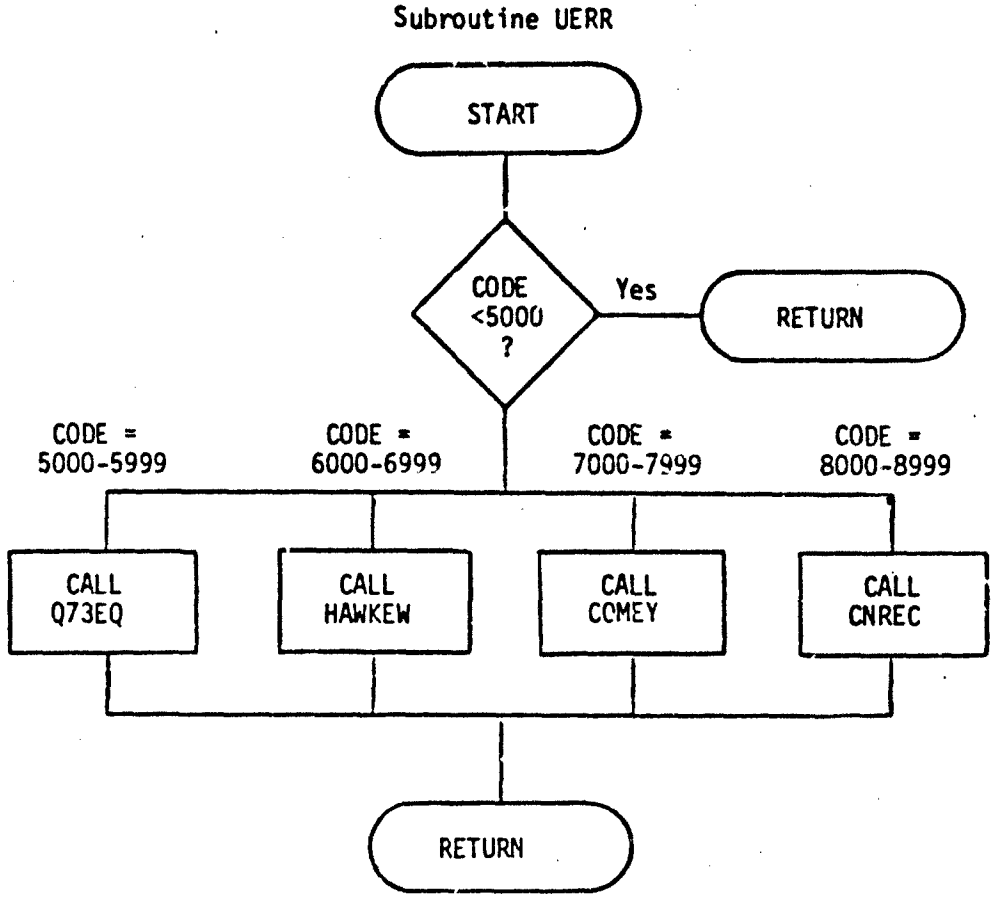
3. Function PRIOR is no longer used.

4. Subroutine STATE has already been written and it is included in the control system module.

Page 93

1. Subroutine UERR is called when a fatal error occurs to give the analyst a chance to print out special information. The fatal error code value is passed into UERR as a parameter. Various ranges of error codes have been assigned, and these are given in Figure II-3. Also shown is a flowchart showing UERR calls to handle errors by system module.

2. Function USERF is called only for UF designators on TAS cards and ATA cards. It is used only to set task times or attribute values. General user code should be called through subroutine UTASK.



Error Code Range	Type of Error
3000-4999	MSAINT error
5000-5499	Either Group or Battalion Q-73 error
5500-5749	Group only error
5750-5999	Battalion only error
6000-6999	IHAWK error
7000-7999	Common System Module Programs
8000-8999	Control error

Figure II-3. UERR Flowchart and Error Codes.

5-0 CHANGES TO SECTION V (REFERENCE 1)

Pages 95-96

These routines can only be called to get information on the current entity at the current task.

Page 97

Do not use: GETPR, PUTPR, QRANK or TIMEQ.

Pages 99-101

Subroutines: CLRHI, CLROB, CLRTP, UCLCT, UHIST, UTMSA, and UTMST

have been rewritten. They are now used to clear or collect statistics only and not to print out any statistics. See Figure II-4 for descriptions of how to use these routines.

Page 102

None of these routines are available now.

6-0 CHANGES TO SECTION VII (REFERENCE 1)

Several additional error codes have been added. These are defined in Table II-1. This table could be considered an extension of Table 14 in reference 1.

7-0 CHANGES TO SECTIONS VIII, IX, AND X (REFERENCE 1)

Page 119

See reference 6 for a description of the implementation of MOPADS and MSAINT.

Pages 120-121

Disregard these two pages. MSAINT and MOPADS are now set up to be run in a virtual environment; therefore, no overlay structure is used.

Pages 122-126

See Section III, 2-5 for a description of the handling of additional dimensioning requirements.

Page 127

See Figure III-4 for a new MSAINT common listing. See Figure II-5 for a listing of subroutine MAIN (excluding the common blocks).

```
      SUBROUTINE CLRHI(IND,NCPY)
C
C*****CLEARS THE STORAGE CELLS FOR EITHER ONE HISTOGRAM
C*****FOR ONE COPY OR ALL HISTOGRAMS FOR ALL COPIES.
C*****THIS IS FOR USER HISTOGRAMS.
C
C*****INPUT PARAMETERS:
C     IND - >0 HISTOGRAM NUMBER TO BE CLEARED
C           <0 CLEAR ALL USER HISTOGRAMS
C     NCPY - COPY NUMBER FOR HISTOGRAM TO BE CLEARED
C            (NOT USED IF IND<0)
C
```

```
      SUBROUTINE CLROB(IND,NCPY)
C
C*****CLEARS THE STORAGE CELLS FOR EITHER ONE OBSERVATION
C*****STATISTIC FOR ONE COPY OR ALL OBSERVATION STATISTICS
C*****FOR ALL COPIES.  THIS IS FOR USER OBSERVATION STATS.
C
C*****INPUT PARAMETERS:
C     IND - >0 USER OBSERVATION STAT NUMBER TO BE CLEARED
C           <0 CLEAR ALL USER OBSERVATION STATS
C     NCPY - COPY NUMBER OF THE COPY WHOSE OBSERVATION
C            STATISTIC IS TO BE CLEARED (NOT USED IF
C            IND<0)
C
```

```
      SUBROUTINE CLRTP(IND,NCPY)
C
C*****CLEARS THE STORAGE CELLS FOR USER TIME-PERSISTENT
C*****STATISTICS FOR EITHER ONE TP STAT FOR ONE COPY
C*****OR ALL TP STATS FOR ALL COPIES.
C
C*****INPUT PARAMETERS:
C     IND - >0 TIME-PERSISTENT STAT TO BE CLEARED
C           <0 CLEAR ALL USER TIME-PERSISTENT STATS
C     NCPY - COPY NUMBER OF STAT TO BE CLEARED
C            (NOT USED IF IND<0)
C
```

Figure II-4. Usage of Statistics Collection Utilities.

```
      SUBROUTINE UCLCT(XX,ICLCT,NCPY)
C
C*****THIS SUBROUTINE NOW ONLY COLLECTS USER OBSERVATION STATS
C
C*****INPUT PARAMETERS:
C     XX - VALUE TO BE RECORDED AS AN OBSERVATION
C     ICLCT - USER OBSERVATION STATISTIC NUMBER
C     NCPY - COPY ROW NUMBER
C
```

```
      SUBROUTINE UHIST(XX,IHIST,NCPY)
C
C*****NOW USED ONLY FOR COLLECTION
C
C*****INPUT PARAMETERS:
C     XX - OBSERVATION VALUE TO BE PLACED IN A CELL
C     ICLCT - USER HISTOGRAM NUMBER
C     NCPY - COPY NUMBER
C
```

```
      SUBROUTINE UTMSA(XX,T,ISTAT,NCPY)
C
C*****NOW USED ONLY TO COLLECT USER AVERAGED TIME-PERSISTENT
C*****STATISTICS. SEE THE SAINT USER'S GUIDE(JUNE 1978) FOR
C*****A DESCRIPTION OF XX,T,AND ISTAT.  NCPY IS THE CURRENT
C*****COPY NUMBER.
C
```

```
      SUBROUTINE UTMST(XX,T,ISTAT,NCPY)
C
C*****NOW USED ONLY FOR COLLECTION
C*****XX,T, AND ISTAT ARE DESCRIBED IN THE SAINT USER'S
C*****GUIDE (JUNE 1978). NCPY IS THE COPY NUMBER FOR THE
C*****STATISTIC TO BE COLLECTED.
C
```

Figure II-4 (continued)

Table II-1. Additional Execution Error Code Definitions

ERROR CODE	ERROR CONDITION	LOCATION OF CALL TO SERR
3607	Zero copy number passed into DRAND. Check user calls to DRAND.	DRAND
4002	A user task clearing could not be performed as requested. Check calls to UTCLR.	CLRF4
4003	A user resource clearing could not be performed as requested. Check calls to URCLR.	CLRF4
4004	Overflow of JSIG array. Too many user task and resource clearings and signalings at the same time. Increase the dimension of the JSIG array.	URCLR UTCLR
4005	Exactly one Task must be signaled with these programs	URCLR UTCLR
4010	A non-source task was returned from subroutine GSOURA	GASP
4011	Too many Control Entities	GASP (others)

```
      SUBROUTINE MAIN
C
C--FORTRAN COMMON STATEMENTS HERE
C
C
C*****INITIALIZE READ, PRINT, AND PUNCH KEYS
C
      NCRDR=KUNITN(3)
      NPRNT=KUNITN(6)
      NPUNCH=0
      NRNIT=KUNITN(8)
       NKENT = KUNITN(4)
      CALL OPENSU(NKENT,'SCRATCH.DAT ','SCRATCH','UNFORMATTED',IOERR,
     -            *900)
      CALL OPENSU(NRNIT,'TRACE.DAT','UNKNOWN','FORMATTED',IOERR,*900)
      REWIND NRNIT
C
C***INITIALIZE ARRAY MAXIMA IN COMMON BLOCK COM01
C
      ID=1000
      IM=2
      INN=3
      INNA=750
      IMN=200
      MAXDS=10000
      MNNPT=40
      MDNSS=60
      MDDAT=2000
      MDQP=1100
      MDSTR=2400
      MEQT=1000
      MFLAG=20
      MMDFN=20
      MMSTN=900
      MNCEL=1500
      MNCLS=3000
      MNCLT=20
      MNCUP=1100
      MNHIS=20
      MNUPA=1800
      MNPLT=10
      MNSIP=20
      MNSUA=20
      MNVAR=10
      MNVPP=10
      MOPHO=20
      MPARM=100
      MPLOT=10
      MSTAT=20
      MSTAT=30
      MTCNR=2000
      MXSTA=20
      MYADA=2550
      MQQ=4
C
C***INITIALIZE ARRAY MAXIMA IN COMMON BLOCK COM26
C
      MMSYS=MMSTSM
      MMAB2=150
      MSSRV=5
      MRQUS=17
      MCOLS=200
      MNRW=11
      MNCL=100
      LFDP=15
      LFDD=20
      MFROW=25
      MFCOLS=30
      MMNEG=150
      MCCOL=24
      MTRIB = 300
      MMBIR = MNCPT + 1
      MOPR = (LFDD-LFDP) + MNCPT
      MSCRN = 10
      MNTRD = 3
      MIDDB = 300
C
C***ARRAY MAXIMUM IN COMMON BLOCK COM25
C
      MJSIG=500
C
C*****EXECUTION CYCLE -- READ INPUT DATA, THEN INITIATE THE SIMULATION
C
      IFIN=0
  100 CALL DATIN
      IF(IERRF.EQ.0)CALL GASP
      RETURN
C
  900 CALL ERRORA( SYSTEM IO ERROR OPENING SCRATCH OR TRACE FILE ,1,
     -            'MAIN',,IOERR,1,0.,0,0)
C
      END
$
```

Figure II-5. Subroutine MAIN.

Page 129

Add the rows contained in Table II-2 to Table 17 in reference 1.

Pages 130-133

No revised version of this table is provided.

Table II-2. Additional MSAINT Common Block Characteristics

COMMON BLOCK	CHARACTERISTICS
COM25	User Clearings
COM26	MSAINT Variables Used During Execution
COM27	MSAINT Variables Used During Input
COM28	Current Copy, System Module Number
COM29	Character Variables

III. ADDITIONS TO SAINT

1-0 INTRODUCTION

Several features and capabilities have been added to SAINT in the implementation of MSAINT. The usage of these new capabilities is described in this section.

2-0 SHAREABLE NETWORKS

2-1. General Discussion

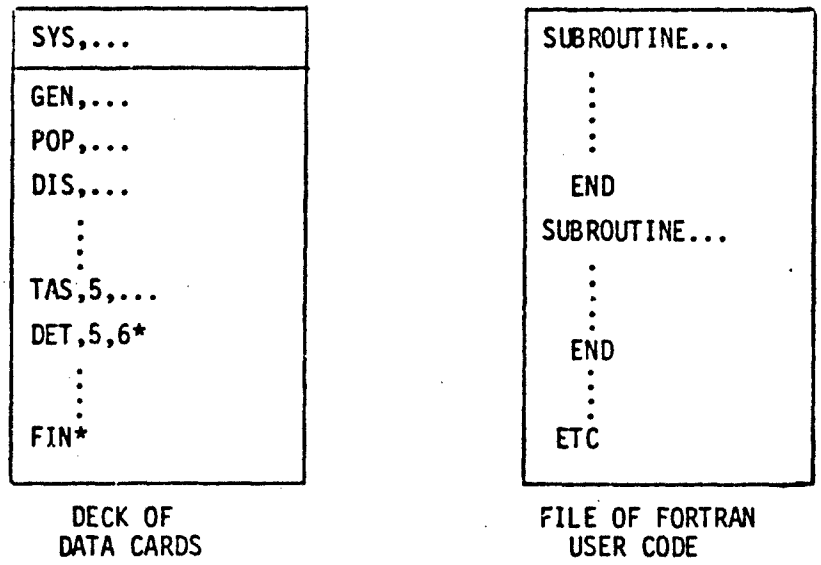
In the original SAINT, a simulation could be run only with one deck of data cards which represents one system. Each such deck of data cards could then have a file of FORTRAN subprograms which are called during the simulation. These subprograms are henceforth referred to as user code. Reference 1 describes the way this user code was interfaced with the SAINT software.

The MOPADS framework requires a much more flexible system of inputting and using SAINT data card decks and their associated user code. Generic system models are developed for each type of air defense component, such as an AN/TSQ-73 missile minder or an IHAWK fire unit. Each generic model contains a deck of MSAINT data cards and a file of user code. This generic model is called a system module. Each MOPADS simulation requires only one system module input for each type of air defense component in the simulation. In this way, multiple air defense components of the same type can be modeled using only one set of data input. This capability is called shareable networks, since each of the like components are sharing the same network data input.

If five AN/TSQ-73 units are simulated, they would all use the same system module input. Each of the five units would be called a "copy" of the AN/TSQ-73 system module. Since a different trace of events will occur for each copy during a MOPADS simulation, MSAINT was modified to keep track of certain data on a copy specific basis. Statistics are collected for each copy.

Since each MOPADS simulation will most likely have more than one type of air defense component, multiple system modules must be included in the data input. To input the multiple decks of data, the decks are stacked. A new data card type was added (called the SYS card). This card is placed at the beginning of each system module input deck. See Figure III-1 for a pictorial representation of the inputs to MSAINT.

One system module consists of the following:



If a MOPADS simulation required the input for 3 system modules, the data card decks would be stacked and all user code would be linked to the MSAINT program.

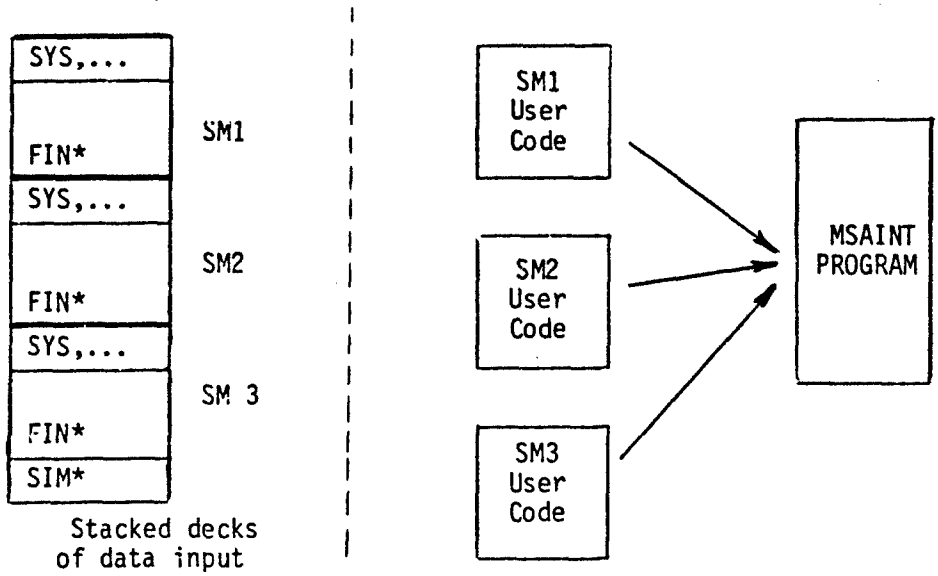


Figure III-1. Inputs to MSAINT.

2-2. SYS Card.

Each system module input deck must have a SYS card as its first card and a GEN card as the second card. The SYS card is automatically created by the MOPADS software and placed at the top of each deck when the decks are stacked (to be read in by MSAINT). Its format is described in Figure III-2.

2-3. Standard Attribute Values.

Each entity which flows through the MSAINT networks must have at least two Information Attributes (IA's). The meaning of the first two IA's is fixed. The first IA value is the entity ID. Entities are either operators or control entities (non-operators). If the entity is an operator, its ID is a positive integer. Control entities have negative ID's. Operator ID's are assigned by the MOPADS software. Some control entities are needed to handle air scenario processing. These are all assigned an entity ID = -1. All other control entity ID's are internally assigned in the order that they are first scheduled at a source node.

The second standard IA value is reserved for the copy number of the copy "owning" the entity. Copy numbers (also called copy row numbers) represent the row number in the NCOPY array and other copy-specific arrays containing data on the copy. Copy numbers are also assigned internally when the entities are scheduled at source nodes.

SYS - System Module Header Card

Field	Value	Default	Description
1	Alphanumeric	*	Card Identification (SYS)
2	Integer	*	System Module Number 1- Control System Module 2- Q-73 Group 3- Q-73 Battalion 4- IHAWK
3	Alphanumeric	N	Echo Option N- No echo provided P- Partial echo (Data cards only) F- Full echo (Data cards plus formal echo report)
4	Integer	1	Number of copies of this SM needed for the current simulation.

* No default allowed, this field must be specified.

Figure III-2. SYS Data Card Description

2-4. New Random Number Scheme.

Pseudo-random numbers are used by MSAINT to sample probabilistic branching and for the generation of random deviates for task times. Formerly in SAINT, one pseudo-random number stream was input (or the default was used) and all uses of random numbers used the same stream.

In MSAINT, one seed value can be input through the MOPADS user interface. From this seed value, random samples are taken to produce initial seed values for each copy in the simulation. During the simulation each copy will then sample from its own stream of pseudo-random numbers. Within each copy, the same stream is used for both probabilistic branching and random deviate generation.

A new FUNCTION DRAND was implemented in MSAINT. See Figure III-3 for a listing of the new FUNCTION DRAND.

2-5. Copy and System Module Specific Arrays.

A dimension was added to several arrays in MSAINT to account for storage on a system module specific or copy specific basis. The system module specific arrays need to keep track of data only for each system module type, since the values do not change between copies of the same system module. The copy specific arrays need to keep track of data which is unique to each copy, regardless of the system module type.

During the usage of MOPADS and MSAINT, it may become necessary to increase the number of system modules allowed or the number of copies allowed in MSAINT. This would involve increasing the value of one or both of the following variables:

MMCPY - maximum number of copies

MMSYS - maximum number of system module types

Currently, MMCPY = 30 and MMSYS = 4. In addition to increasing one or both of these variable values (set in the MAINT program of MSAINT), the analyst must also modify the dimensions of the system module specific and/or copy specific arrays in MSAINT common. A listing of MSAINT common is shown in Figure III-4.

The system module specific arrays are listed in Table III-1. The copy specific arrays are listed in Table III-2. The dimension number refers to which dimension must be increased. For example, if the second dimension of array XX(5,20,3), was to be increased to 30, then the array would be dimensioned to XX(5,30,3).

```
      FUNCTION DRAND(ISTRM)
C
C*****ALGORITHM FOR SEED GENERATION IS ADAPTED FROM FUNCTION
C*****RAND DEVELOPED BY L. SCHRAGE AND USED WITH THE AUTHOR'S
C*****PERMISSION.
C*****DRAND NOW IS PASSED IN THE COPY NUMBER OF THE COPY
C*****NEEDING A RANDOM NUMBER.  EACH COPY HAS ITS OWN
C*****STREAM NOW.  ALL STREAMS ARE INITIALIZED USING
C*****THE ONE SEED VALUE PASSED IN FROM THE DATA BASE.
C*****A DEFAULT INITIAL STREAM IS PROVIDED IF THE USER
C*****DOES NOT PROVIDE ONE THROUGH THE USER INTERFACE.
C
C
C::
      COMMON /COM06/ TNOW,TTNEX,MFAD,SEED,ISEED,NCRDR,NPRNT,NPUNCH,
     *               NRNIT,NRENT,MNDC,NDC,NDTN,NNTC,IISED(31)
      COMMON /COM26/ MMCPY,MMSYS,NTRACK(3400),NABA2(150),MNAB2,MMCOP(4),
     *               NXITR,NTRID2(3,300),NSSRW,MROWS,NCOLS,NXTID,
     *               NSITE(1100),MNRW,MNCL,NCOPY(30,26),LFOP,LFDB,NXMON,
     *               NXDAY,NXYR,NFSTOR(750),MFROW,NFCOLS,NFSET,
     *               NEGID(150),MMNEG,MCCOL,TSTRTR,TFINR,DELT,MTRID,
     *               NSETR,NNSTR,NCNEG,NDB4(4,2,30),NTRDB(3,3),
     *               NSCRN(150,10),NNDBA,NNDB4,MOFR,MSCRN,MMTRD,
     *               NDBAD(2,300),NXDBD,LASTM,NMES(4),NOFRI(150,3)
      DIMENSION XSTOR(750),XSITE(1100),XTRACK(3400)
      EQUIVALENCE (XSTOR(1),NFSTOR(1)),(XSITE(1),NSITE(1)),
     +            (XTRACK(1),NTRACK(1))
      DIMENSION LSEED(31)
      DATA LA,LB15,LB16,LP/16807,32768,65536,2147483647/
      SAVE LSEED
      IF(ISTRM) 20,10,70
   10 CALL SERR(3607)
   20 ISTRG = -ISTRM
      IF(ISTRG-NNSTR) 30,30,10
   30 JSEED = IISED(ISTRG)
      GO TO 60
   50 LSEED(ISTRG) = JSEED + LP + 1
      GO TO 100
   60 IF(2*(JSEED/2).EQ.JSEED) JSEED = JSEED + 1
      IF(JSEED) 50,60,65
   65 LSEED(ISTRG) = JSEED
      GO TO 100
   70 ISTRG = ISTRM
      IF(ISTRG-NNSTR) 80,80,10
   80 JSEED = LSEED(ISTRG)
      LHI = JSEED/LB16
      LALO = (JSEED-LHI*LB16)*LA
      LEFTLO = LALO/LB16
      LFHI = LHI*LA+LEFTLO
      K = LFHI/LB15
      JSEED = (((LALO-LEFTLO*LB16)-LP)+(LFHI-K*LB15)*LB16)+K
      IF(JSEED.LT.0)JSEED = JSEED + LP
      LSEED(ISTRG) = JSEED
  100 DRAND = FLOAT(JSEED)*4.656612875E-10
      RETURN
      END
```

Figure III-3. DRAND Listing.

```
      COMMON /COM01/ ID,IM,IMM,IMNA,IMN,MAXDS,MDNPT,MDNSS,
     *               MDOAT,MDOP,MDSTR,MEQT,MFLAG,MNPTS,MMDFN,MMSTU,
     *               MNCEL,MNCLS,MNCLT,MNCUP,MNHIS,MNOPA,MNPLT,MNPTO,
     *               MNSTP,MNSWA,MNVAR,MNVPP,MOPNO,MPARM,MPLOT,MSTAT,
     *               MSYAT,MTCHR,MXSTA,MYABA,NOQ
      COMMON /COM02/ ATRIB(3),JTRIB(2),QSET(3000),NSET(4000),MFA,
     *               MXX,MFE(4),MLE(4),NQ(4)
      COMMON /COM03/ IPOS,JPOS,KPOS,LPOS,MPOS,NFA,NAN,IERRW,IERRF,IFIN,
     *               IIECH,INDXS,INDXT,INDX,JNDX,NNDX,IF,NUMFL,ICONT,
     *               HIVAL,IBLNK,IZERO,LA,LB,LC,LD,LE,LF,LG,LH,LI,
     *               LJ,LK,LL,LM,LN,LO,LP,LQ,LR,LS,LT,LU,LV,LW,LX,LY,LZ
      COMMON /COM04/ IDFAL(4),KREAD(45),IFLAG(50),IRSUL(50),RESUL(50),
     *               IABC(8,50),KARD(80),IDIG(9)
      COMMON /COM05/ NPROJ(4),MON(4),NDAY(4),NYR(4),NAME(2,4),NRUN,
     *               NRUNS,NSKSR,NSKST,NNEQD,NNEQS,NNEQT
      COMMON /COM06/ TNOW,TTNEX,MFAD,SEED,ISEED,NCRDR,NPRNT,NPUNCH,
     *               NRNIT,NRENT,MNDC,NDC,NDTN,NNTC,IISED(31)
      COMMON /COM07/ NDE,NOPAT(4),NSYAT(4),NDOP(4),NMDFN(4),NN(4),
     *               NFRMS(4),JFLPR,KRNK,XINN,NFLAG,NNCLT(4),NNHIS(4),
     *               NNPLT,NNSTA,NNSTP(4),NPLOT,NSTTS(4)
      COMMON /COM08/ NEIF,NSIF,ITRACE,NRTSP,NRTEF,KTRACE,MTRACE,
     *               NSVVS,NSVVE,LTRACE,NRTSS,NRTES
      COMMON /COM09/ PARAM(100,5,4),PARM1(100,4),PARM4(100,4)
      COMMON /COM10/ CACIN(200,3,4),ITCHR(200,4),LLTSK(200,2,4),
     *               LSINK(200,4),MACIN(200,4,4),MFEN(200,4),
     *               MFSTT(200,4),MOF(200,4),MPO(200,4),NDCH(200,2,4),
     *               NDEL(200,30),NDPT(200,30),NPOP(200,4),NPOR(200,4),
     *               NPSGN(200,4),NREL(200,30),NRELP(200,4),NREL2(200,4)
     *               ,NSIGN(200,4),NTC(200,30),NTCHR(200,4),NTYPE(200,4)
     *               ,KMARK(200,4),XMARK(200,30)
      COMMON /COM11/ BUSY(20,30),LLRES(20,2,4),NBUS(20,30),NOPTR(20,30),
     *               TLST(20,30),NOPA(1800),RSTAT(80,30),KFLAG(20,30)
      COMMON /COM12/ YABA(2550),NABA(750),STCHR(2000)
      COMMON /COM13/ MDFNS(20),MFSTL(20,30),MFSTU(900)
      COMMON /COM14/ NSINK(20,4),KSTPE(20,4),KSTTM(20,4),XSTUS(20,30),
     *               NCELS(20,2,30),XLOW(20,4),WIDTH(20,4),
     *               SUMAI(20,5,30),SUMAF(20,5,30),JCELS(3000)
      COMMON /COM15/ DESCR(10000),DOATT(2000),NDSTR(2400),SYSAT(30,30)
      DIMENSION NESCR(10000)
      EQUIVALENCE(DESCR(1),NESCR(1))
      COMMON /COM16/ AAERR,DTMAX,DTMIN,DTSAV,IITES,LLERR,RRERR,TTLAS,
     *               TTSAV,DTSUC,DTFUL,DTNOW,ISEES,RESLS,DTACC,LLSAV,
     *               LSAVE
      COMMON /COM17/ SS(1000 ),SSL(1000 ),DD(1000),DDL(1000)
      COMMON /COM18/ IS(20)
      COMMON /COM19/ LFLAG(20),NPOSS(20),NPOST(20),LLMON(20,2),
     *               NABAT(40),NABAS(60),THRES(120)
      DIMENSION ITHRS(120)
      EQUIVALENCE(ITHRS(1),THRES(1))
      COMMON /COM20/ NSTAI(20),LLSVS(20,2),SSTPV(20,6)
      COMMON /COM21/ DTFLT(10),IITAP(10),NNPTS(10),NNVAR(10),NNVP(10),
     *               LLPLT,NNPT,LLPHI(10),LLPLO(10),LLSYM(10),FPHI(10),
     *               FPLO(10),NVP(100),LLSVP(11,2),QPSET(1100)
      COMMON /COM22/ TTIME,PFIRB
      COMMON /COM23/ LLUGC(20,2,4),USOBV(20,5,30),LLUGT(20,2,4),
     *               TTCLR(20,30),USTPV(20,6,30),LLUGH(20,2,4),
     *               NNCEL(20,2,30),HHLOW(20,4),HHWID(20,4),JJCEL(1500)
      COMMON /COM24/ DPLOT(10),ITAPE(10),NPTSV(10),NVARS(10),LPLOT,
     *               NPTEX,LPHIH(10),LPLOW(10),LSYMB(10),PHIH(10),
     *               PLOW(10),LLUGP(11,2),UPSET(1100)
      COMMON /COM25/ JSIG(500),MJSIG,NEXTSG
      COMMON /COM26/ MMCPY,MMSYS,NTRACK(3400),NABA2(150),MNAB2,NMCOP(4),
     *               NXTTR,NTRID2(3,300),NSSRW,MROWS,NCOLS,NXTID,
     *               NSITE(1100),MNRW,MNCL,NCOPY(30,26),LFOP,LFDB,NXMON,
     *               NXDAY,NXYR,NFSTOR(750),MFROW,NFCOLS,NFSPT,
     *               NEGID(150),MMNEG,MCCOL,TSTRTN,TFINK,DELT,MTRID,
     *               NSPTR,NNSTR,NCNEG,NDB4(4,2,30),NTRDB(3,3),
     *               NSCRN(150,10),NNDBA,NND34,MOFR,MSCRN,MMTRD,
     *               NDBAD(2,300),MXDBD,LASIM,NMES(4),NOPRI(150,3)
      DIMENSION XSTOR(750),XSITE(1100),XTRACK(3400)
      EQUIVALENCE (XSTOR(1),NFSTOR(1)),(XSITE(1),NSITE(1)),
     +            (XTRACK(1),NTRACK(1))
      COMMON /COM27/ NCURS,NXDDAT,NCT(20,4),NCTU(20,4),NCNEXT
      COMMON /COM28/ NCURCO,NCURSY,ISTATZ
      COMMON /COM29/ NTRID(300),NXPROJ(8),NXNAME(8)
      CHARACTER NTRID*8,NXPROJ,NXNAME
```

Figure III-4. Listing of MSAINT Common.

Table III-1. System Module Specific Arrays.

COMMON BLOCK	ARRAY NAME	DIMENSION NUMBER
COMØ5	NPROJ	1st
	MON	1st
	NDAY	1st
	NYR	1st
	NAME	2nd
COMØ7	NOPAT	1st
	NSYAT	1st
	NDOP	1st
	NMDFN	1st
	NN	1st
	NPRMS	1st
	NNCLT	1st
	NNHIS	1st
	NNSTP	1st
	NSTTS	1st
COMØ9	PARAM	3rd
	PARM1	2nd
	PARM4	2nd
COM1Ø	CACIN	3rd
	ITCHR	2nd
	LLTSK	3rd
	LSINK	2nd
	MACIN	3rd
	MFEN	2nd
	MFSTT	2nd
	MOP	2nd
	NDCH	3rd
	NPOP	2nd
	NPOR	2nd
	NPSGN	2nd
	NREL1	2nd
	NREL2	2nd
	NSIGN	2nd
	NTCHR	2nd
	NTYPE	2nd
	KMARK	2nd
COM11	LLRES	3rd

Table III-1 (continued)

COMMON BLOCK	ARRAY NAME	DIMENSION NUMBER
COM14	NSINK	2nd
	KSTYP	2nd
	KSTIM	2nd
	XLOW	2nd
	WITLI	2nd
COM23	LINGC	3rd
	LLUGT	3rd
	LLUGH	3rd
	HHLOW	2nd
	HHWID	2nd
COM26	NMCOP	1st
COM27	NCT	2nd
	NCTU	2nd

Table III-2. Copy Specific Arrays.

COMMON BLOCK	ARRAY NAME	DIMENSION NUMBER
COM09	NDEL	2nd
	NDPT	2nd
	NREL	2nd
	NTC	2nd
	XMARK	2nd
COM11	BUSY	2nd
	NBUS	2nd
	NOPTR	2nd
	TLST	2nd
COM13	MFSTW	2nd
COM14	XSTUS	2nd
	NCELS	3rd
COM15	SYSAT	2nd
COM23	USCBV	3rd
	USTPV	3rd
	TTCLR	2nd
	NNCEL	3rd
	NCOPY	1st
	NDBL	2nd

3-0 NEW USER-WRITTEN ROUTINES

In Section IV of Reference 1, several user-written routines are described. Also, in Section II, 4-0 of this document some changes in the usage of these user-written routines are described. In addition to these changes, three new user-written routines have been added to MSAINT.

SUBROUTINE UTASK(NT,NPLACE)

Subroutine UTASK is called once each time a task node reaches a first predecessor completion (FPC),release time, start time, completion, or clearing. Two parameters are passed into subroutine UTASK, the task node number (NT) and the task occurrence time (NPLACE). The occurrence times use the following codes:

1 - First Predecessor Completion (FPC)-(if different from the Release time. If not, then UTASK is not called with NPLACE = 1.)

2 - Release

3 - Start

4 - Completion

5 - Clearing

Subroutine UTASK controls the task driven user code in the system module user code files. It has a rigid structure which is shown in the form of a hierarchy chart in Figure III-5. UTASK controls the calling of subroutine TRACE, which prints trace output to an external file. It then calls a special controlling routine based on the current system module (the system module type for the current entity being processed). These routines then call individual subroutines for each task node which requires unique processing. The parameter NPLACE is passed into these task node routines. Each task node routine may then have a computed GO TO statement with NPLACE as the parameter (with values 1 to 5). In this way code which is unique to the task occurrence time can be easily reached.

As each system module is added to MOPADS, one controlling routine must be added to be called by UTASK. The individual task routines must then be called by it for the tasks in that system module.

WARNING

Information attributes of the current entity may not be changed at node release time (NPLACE=2). This is necessary to implement self clearing (see Section III, 4-3).

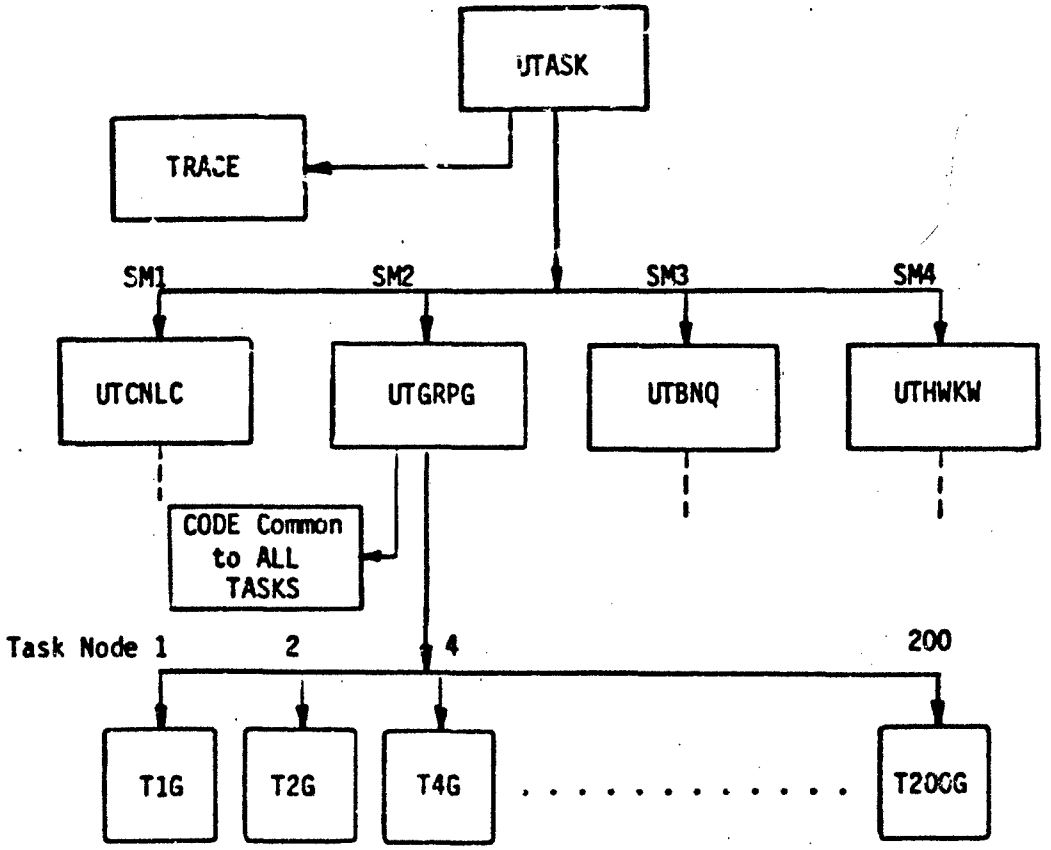


Figure III-5. UTASK Hierarchy Chart.

SUBROUTINE USTART(NRUN)

Subroutine USTART is called to allow the analyst to initialize variables and arrays at the beginning of each run. NRUN is passed in as the run number of the run about to be made. Separate routines should then be called to initialize the variables and arrays by system module.

FUNCTION USERIN(NCODE)

The MOPADS analyst can use IRA and ISA cards in system module data input decks. For a description of the usage of IRA and ISA cards, see reference 1. These cards are used to set initial values of resource attributes and system attributes. The same initial value is then used for the attributes in each copy of the system module.

If the UF designation is used to set these initial attributes, then function USERIN will be called. The UF code will be passed in as the parameter NCODE. The codes for calling USERIN and USERF are independent of each other. USERF is called for all UF designations on all card types other than IRA and ISA cards.

4-0 NEW USER-CALLABLE ROUTINES

Several new utility routines were added to MSAINT to perform specialized processing from system module user code. The usage of these is defined in this section.

4-1. Copy and System Module Information.

System modules are numbered 1 through 4 currently. The code values are defined below.

Codes for System Modules

1 - Control (track processing, simulation time)
2 - Group Q-73
3 - Battalion Q-73
4 - IHAWK

Other codes will be added as system modules are added to MOPADS.

Copies are numbered 1 through 30 currently. Copy numbers are assigned when the copy data is placed in the NCOPY array from the MOPADS DB.

In addition to these identifying numbers, each copy has what is called a copy ID. The copy ID is stored in column 1 of the NCOPY

array. It is a 4-digit number unique to each copy which is assigned by the MOPADS software. There is currently no reason for the analyst to use the copy ID because the copy row number is the number used to uniquely address copies in user code.

The utility routine descriptions are included in Figure III-6. The comment sections of each routine should sufficiently describe their usage.

```
      SUBROUTINE BDCOPY(NCID,NNSYS,NNCOP)
C
C*****BREAKS DOWN THE 4 DIGIT COPY ID INTO THE
C*****SYSTEM MODULE NUMBER AND THE COPY NUMBER
C*****WITHIN THAT SYSTEM MODULE.
C
C*****INPUT PARAMETERS:
C     NCID - COPY ID
C
C*****OUTPUT PARAMETERS:
C     NNSYS - SYSTEM MODULE NUMBER
C     NNCOP - COPY NUMBER
C
```

```
      SUBROUTINE FINDC(NZSYS,NDIM,ITEMP,INUM)
C
C*****FINDS ALL COPY ROW NUMBERS FOR A GIVEN SYSTEM
C*****MODULE NUMBER
C
C*****INPUT PARAMETERS:
C     NZSYS - SYSTEM MODULE NUMBER
C     NDIM - DIMENSION OF ITEMP
C
C*****OUTPUT PARAMETERS:
C     ITEMP - ARRAY CONTAINING THE COPY ROW NUMBERS
C     INUM - -2 ILLEGAL SYSTEM MODULE NUMBER
C            -1 TOO MANY COPY ROW NUMBERS FOR ITEMP
C            >=0 NUMBER OF COPY ROW NUMBERS IN ITEMP
C
```

Figure III-6. Descriptions of Utilities Providing Copy and System Module Information.

```
      SUBROUTINE GTCOPY(IROW,NF,NL,IDIM,ISTORE)
C
C*****USED BY VARIOUS UTILITIES TO GET INFORMATION
C*****OUT OF THE NCOPY ARRAY, GIVEN THE COPY ROW #.
C
C*****INPUT PARAMETERS:
C     IROW - COPY ROW #
C     NF - FIRST COLUMN OF INFORMATION
C     NL - LAST COLUMN OF INFORMATION ( SAME AS NF IF
C          ONLY ONE WORD WANTED)
C     IDIM - DIMENSION OF ISTORE (NUMBER OF WORDS WANTED)
C
C*****OUTPUT PARAMETERS:
C     ISTORE - ARRAY CONTAINING THE DESIRED VALUES FROM
C              THE NCOPY ARRAY
C
```

```
      FUNCTION NCOP(ICN)
C
C*****GIVEN THE COPY ROW NUMBER, RETURNS THE 4 DIGIT
C*****COPY ID FROM THE NCOPY ARRAY.
C
C*****INPUT PARAMETERS:
C     ICN - COPY ROW #
C
C*****OUTPUT PARAMETERS:
C     NCOP - COPY ID
C
```

```
      FUNCTION NCOPR(IOPR,KFLG)
C
C*****RETURNS THE COPY ROW NUMBER FOR A GIVEN ENTITY ID
C*****ALSO SETS THE CURRENT COPY AND SYSTEM MODULE IF
C*****KFLG = 1.
C
C*****INPUT PARAMETERS:
C     IOPR - ENTITY ID (<0 NON-OPERATOR, >0 OPERATOR)
C     KFLG - FLAG TO INDICATE IF THE CURRENT COPY
C            AND SYSTEM MODULE ARE TO BE SET
C
C*****OUTPUT PARAMETERS:
C     NCOPR - 0 MEANS THE ID HAS NO COPY ID
C             >0 COPY ROW NUMBER CONTAINING THE ENTITY
C                WITH ID IOPR
C
```

Figure III-6. (continued)

```
      FUNCTION NSYS(ICN)
C
C*****RETURNS THE SYSTEM MODULE NUMBER GIVEN THE COPY ROW #
C
C*****INPUT PARAMETERS:
C     ICN - COPY ROW NUMBER (IN THE NCOPY ARRAY)
C
C*****OUTPUT PARAMETERS:
C     NSYS - SYSTEM MODULE NUMBER
C
```

Figure III-6. (continued)

4-2. Task, Entity, Resource, and Attribute Data.

Entities flow through the MSAINT networks from task node to task node. Task nodes sometimes require resources to perform the tasks. There are three types of attributes: 1) information, 2) resource, and 3) system. See Wortman et al (1978b) for a more detailed description of these items.

Information involving these simulation items can be obtained using the utility routines described in this section. See Figure III-7 for these descriptions.

4-3. User Clearing.

Formerly in SAINT, clearing was restricted to the following procedure. When one task reaches its completion time, if it has a TCL or RCL designator on it, it would cause the specified other task to be cleared from its current task to a specified task to be signaled. This procedure turned out to be overly restrictive for usage in the MOPADS framework. For this reason, user clearing was implemented in MSAINT. User clearing allows any task to be cleared by any other task in the same copy from calls to UTCLR or URCLR in user code. In addition, entities can clear themselves at release times from their current task node by calling subroutine SCLEAR.

The usage of these three utility routines is described in Figure III-8.

```
      FUNCTION CTIME(NTASK,NOPR,IERR)
C
C*****RETURNS THE SCHEDULED TASK COMPLETION TIME
C*****FOR A GIVEN OPERATOR-TASK.  ASSUMES THE FIRST
C*****INFORMATION ATTRIBUTE IS THE OPERATOR ID.
C
C*****INPUT PARAMETERS
C     NTASK - TASK NUMBER
C     NOPR - OPERATOR ID
C
C*****OUTPUT PARAMETERS
C     IERR - 1 IF NO ERRORS
C            2 IF TASK NUMBER ILLEGAL
C            3 TASK NOT ACTIVE FOR THE GIVEN


C              OPERATOR, THEREFORE NO SCHEDULED
C              TASK COMPLETION TIME
C     CTIME - SCHEDULED TASK COMPLETION TIME
C
```

```
      SUBROUTINE GETIAI(NAT,NVAL)
C
C*****WORKS SAME AS THE ROUTINE GETIA EXCEPT IT RETURNS
C*****THE VALUE AS AN INTEGER
C
C*****INPUT PARAMETERS:
C     NAT - INFORMATION ATTRIBUTE NUMBER


C
C*****OUTPUT PARAMETERS:
C     NVAL - THE CURRENT VALUE OF INFORMATION ATTRIBUTE NAT FOR
C            THE CURRENT ENTITY
C
```

```
      SUBROUTINE GTIAV(LOPR,XIA,IADIM,IERR)
C
C*****OBTAIN THE IA VALUES FOR A GIVEN OPERATOR
C
C*****INPUT PARAMETER
C     LOPR - OPERATOR ID NUMBER (ASSUMED TO BE
C            INFORMATION ATTRIBUTE 1
C            (IF 0 USE CURRENT ENTITY)
C
```

Figure III-7. Descriptions of Utilities Providing Task, Entity Resource, and Attribute Data.

```
C*****OUTPUT PARAMETERS
C     XIA - ARRAY CONTAINING THE CURRENT VALUES OF
C           THE INFORMATION ATTRIBUTE VECTOR FOR
C           OPERATOR LOPR (MUST BE DIMENSIONED
C           TO IADIM IN THE CALLING SUBPROGRAM)
C     IADIM - DIMENSION OF XIA
C     IERR - = -1 TOO MANY IA'S FOR ARRAY XIA
C            =  0 OPERATOR/IA'S NOT FOUND
C            >0   NUMBER OF IA'S PUT IN XIA
C
------------------------------------------------------------------------

      SUBROUTINE GTNUM(NTDIM,NFILE,NUMT,IERR)
C
C*****FINDS THE TASK NUMBERS FOR ALL TASKS IN EITHER
C*****THE EVENT FILE OR THE WAIT-FOR RESOURCES FILE.
C*****ALSO RETURNS THE COPY NUMBER FOR EACH TASK FOUND.
C
C*****INPUT PARAMETERS
C     NTDIM - NUMBER OF ROWS IN NUMT
C             (NUMT ALWAYS HAS TWO COLUMNS)
C     NFILE - FILE TO SEARCH FOR TASKS
C           = 1 SEARCH THE EVENT FILE
C           =2 SEARCH THE WAIT-FOR-RESOURCES FILE
C
C*****OUTPUT PARAMETERS:
C     NUMT - ARRAY CONTAINING THE TASK NUMBERS AND
C            COPY NUMBERS OF ALL TASKS IN FILE NFILE.
C            NUMT(I,1) IS THE TASK NUMBER OF THE I TH
C            TASK FOUND AND NUMT(I,2) IS THE COPY ROW
C            NUMBER OF THE I TH TASK IN FILE NFILE.
C     IERR - =-2 IF AN ILLEGAL NFILE NUMBER PASSED IN
C            =-1 IF MORE THAN NTDIM TASKS FOUND
C            =0 IF NO TASKS FOUND IN FILE NFILE
C            >0 NUMBER OF TASKS FOUND AND PLACED IN NUMT
C
------------------------------------------------------------------------

      SUBROUTINE GTOPRT(IDENT,NTASK,IERR)
C
C*****FOR A GIVEN OPERATOR, LOCATES THE TASK
C*****WHERE THE OPERATOR IS AT.  ASSUMES THAT
C*****EACH OPERATOR IS AN ENTITY WITH INFORMATION
C*****ATTRIBUTE 1 BEING THE ID.
C
C*****INPUT PARAMETER:
C     IDENT - OPERATOR ID NUMBER
C
```

Figure III-7. (continued)

```
C*****OUTPUT PARAMETERS
C     NTASN - TASK NUMBER WHERE OPERATOR IDENT
C             NOW IS(RETURNED AS NEGATIVE IF THE
C              TASK IS WAITING FOR RESOURCES)
C     IERR - 1 IF TASK NUMBER WAS OBTAINED
C            2 IF NO TASK NUMBER FOUND
C
```

```
      SUBROUTINE GTSTAT(NRESS,ICPY.NSTTA,NTASK,IERR)
C
C*****RETURNS THE STATUS OF A GIVEN RESOURCE FOR THE INDICATED COPY
C
C*****INPUT PARAMETERS
C     NRESS = RESOURCE NUMBER
C     ICPY - COPY ROW NUMBER
C
C*****OUTPUT PARAMETERS
C     NSTTA - STATUS OF RESOURCE NRESS
C           = 1 IF RESOURCE BUSY
C           = 0 IF IDLE
C     IERR - 1 IF RESOURCE FOUND, NO ERRORS
C            2 ILLEGAL RESOURCE NUMBER
C     NTASK - TASK NUMBER WHERE THE RESOURCE IS BEING
C             USED( =0 IF IDLE OR IERR = 2)
C
```

```
      SUBROUTINE PUTIAI(NAT,NVAL)
C
C*****WORKS THE SAME AS SUBROUTINE PUTIA EXCEPT THAT
C*****IT PASSES IN THE VALUE AS AN INTEGER
C
C*****INPUT PARAMETERS:
C     NAT - INFORMATION ATTRIBUTE NUMBER
C
C*****OUTPUT PARAMETERS:
C     NVAL - VALUE TO BE PLACED IN INFORMATION ATTRIBUTE NAT
C
```

Figure III-7. (continued)

```
      SUBROUTINE URCLR(IRNUM,NSIG,NUMSG,ICPY,IERR)
C
C*****USER CALLABLE ROUTINE TO INITIATE A RESOURCE CLEARING
C*****FROM USER CODE. AN ENTRY IS PUT INTO FILE 4 TO REPRESENT
C*****A CLEARING EVENT. THESE ENTRIES ARE THEN PULLED OUT
C*****AND PROCESSED BEFORE THE NEXT TASK COMPLETION.
C
C*****INPUT PARAMETERS:
C     IRNUM - RESOURCE NUMBER TO BE CLEARED
C     NSIG - ARRAY CONTAINING TASK NUMBERS TO BE SIGNALED
C     NUMSG - NUMBER OF TASK NUMBERS IN NSIG
C     ICPY - COPY ROW NUMBER
C
C*****OUTPUT PARAMETERS:
C     IERR - 0 NO ERRORS
C            1 RESOURCE IRNUM NOT BUSY; NO CLEARING DONE
C            2 RESOURCE NUMBER IRNUM IS ILLEGAL
C            3 AT LEAST ONE OF THE TASK NUMBERS IN NSIG
C              IS ILLEGAL
C
```

```
      SUBROUTINE UTCLR(NCLR,IOPR,NSIG,NUMSG,LOPT,IERR)
C
C*****USER CALLABLE ROUTINE TO INITIATE A TASK CLEARING
C*****EVENT FROM USER CODE. AN ENTRY IS PUT INTO FILE 4 TO
C*****REPRESENT A CLEARING EVENT.
C
C*****INPUT PARAMETERS:
C     NCLR - TASK NUMBER TO BE CLEARED
C     IOPR - OPTION FOR SPECIFYING JUST AN OPERATOR ID.
C          <0 CLEAR ALL INCIDENCES OF TASK NCLR FOR THE
C             CURRENT COPY
C          >0 CLEAR ONLY THE INCIDENCE OF TASK NCLR
C             WITH OPERATOR IOPR AT IT
C     NSIG - ARRAY CONTAINING ALL TASK NUMBERS TO BE
C            SIGNALED
C     NUMSG - NUMBER OF TASK NUMBERS IN NSIG
C     LOPT - 1 CLEAR THE TASK ONLY IF ACTIVE
C            2 CLEAR THE TASK ONLY IF WAITING FOR RESOURCES
C            3 CLEAR THE TASK IF EITHER ACTIVE OR WAITING
C              FOR RESOURCES
C
```

Figure III-8. User Clearing Utility Routine Descriptions.

```
C*****OUTPUT PARAMETERS:
C     IERR - 0 NO ERRORS
C            1 UNABLE TO CLEAR THE TASK
C            2 TASK NUMBER NCLR IS ILLEGAL
C            3 AT LEAST ONE TASK NUMBER IN NSIG IS ILLEGAL
C
```

```
      SUBROUTINE SCLEAR(NCURR,IOPR,NTASB,NTIME,IERR)
C
C*****PERFORMS A SELF CLEARING OPERATION. THIS IS ACCOMPLISHED
C*****BY CALLING UTCLR TO PLACE A USER CLEARING EVENT IN FILE 4.
C*****ONLY ONE TASK CAN BE SIGNALED(BRANCHED TO). THIS ROUTINE
C*****IS SET UP TO BE CALLED FROM UTASK ONLY AT RELEASE TIMES.
C
C*****INPUT PARAMETERS
C     NCURR - CURRENT TASK NUMBER TO BE CLEARED
C     IOPR - CURRENT OPERATOR ID TO BE CLEARED
C     NTASB - TASK NUMBER TO BE SIGNALED
C     NTIME - PASSED IN FROM UTASK. USED FOR ERROR  CHECKING TO
C             ASSURE SCLEAR WAS CALLED AT A RELEASE TIME
C
C*****OUTPUT PARAMETERS
C     IERR - 0 - NO ERRORS. CLEARING AND SIGNALING WAS DONE
C            1 - NOT CURRENTLY A RELEASE TIME
C            2 - ENTITY TO BE CLEARED IS ILLEGAL( WRONG TASK
C                NO. OR OPERATOR ID SPECIFIED)
C            3 - TASK NO. TO BE SIGNALED IS ILLEGAL
C            4 - ERROR IN SETTING UP USER CLEARING(TASK TO
C                BE CLEARED NOT IN WAIT FILE)
C
```

Figure III-8. (continued)

4.4. General Simulation-Related Data.

Two general purpose utilities were added to MSAINT. Subroutine GTTIM retrieves the current simulation time and subroutine PFILE prints out the list of entities and their first five IA values for all entities in the given file. File 1 is the event file and file 2 is the wait-for-resources file. See Figure III-9 for full descriptions of the usage of these routines.

```
      SUBROUTINE GTTIM(TT)
C
C*****OBTAINS THE CURRENT SIMULATION TIME
C
C*****OUTPUT PARAMETERS
C     TT - CURRENT SUMULATION TIME
C
```

```
      SUBROUTINE PFILE(NFIL)
C
C*****PRINTS OUT THE CONTENTS OF A FILE. IT DOES
C*****THIS BY PRINTING ONE ROW OF INFORMATION FOR EACH FILE
C*****ENTRY.  EACH ROW CONTAINS THE ENTRY NUMBER, TASK
C*****NUMBER OF THE ENTRY, AND THE FIRST FIVE INFORMATION
C*****ATTRIBUTE VALUES.
C
C*****INPUT PARAMETERS
C     NFIL - FILE NUMBER OF THE FILE TO BE PRINTED OUT
C
```

Figure III-9. Descriptions of General Utilities in MSAINT.

5-0 USER ACCESSIBLE DATA STRUCTURE DEFINITIONS

5-1. General Discussion.

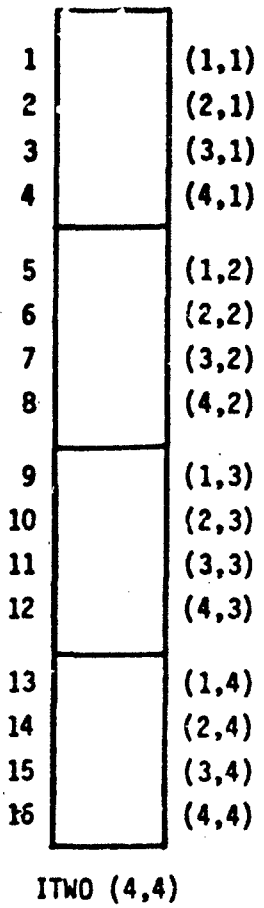
Several arrays and variables have been added to MSAINT labeled common blocks to store MOPADS data which is frequently accessed. Some of these variables are used internally by MSAINT; therefore, their meaning and usage are of no consequence to the analyst. However, several of the new variables contain information that may be accessed by system module user code. The MOPADS analyst needs to know the meaning of these variables and how to access them. This information is provided here.

Several arrays have been set up as one-dimensional arrays in MSAINT common, but they are treated as two-dimensional arrays. These are referred to as pseudo-two-dimensional arrays. Each two-dimensional array is actually stored by FORTRAN as one long stack of columns, see Figure III-10. Now assume that you have a one-dimensional array partitioned into sections. Each section has the same number of elements, and the last word of each section is a pointer variable. The array would look like the one in Figure III-11(a). Now take each section and put them side by side and refer to each section as a "column" (see Figure III-11(b)). All of the pointer elements now line up to be the last row of the columns. Columns can be used or unused. For unused columns, the pointer element points to the next unused column. All used columns have a -1 in the pointer element. A separate variable points into the array to the first unused column. Hence, a one-dimensional linked list system is used to keep track of the unused columns. If a new column is needed, the first unused column (pointed to by the outside variable) is used and the pointer structure is updated. The pointer structure is also updated when a column is freed up. Each array of this type then has a fixed number of columns and rows. The product of the number of columns and rows must of course be equal to the dimension of the array. This type of data structure provides the flexibility needed to keep track of items which may come and go from the system. Also, the MOPADS analyst may want to vary the number of columns and rows to account for additional data.

Track, site, and fire unit data are stored in pseudo-two-dimensional arrays in MSAINT labeled common. Some additional information, such as alphanumeric labels for the track sites, and fire units, is also stored in that type of array. In Section 5-2 these arrays are defined.

Several utility routines have been included in the MOPADS software for handling these arrays: SETVU, PTILU, GTELU, GTRWU, PTRWU. These routines are described in Polito & Goodin (1983).

A list of MSAINT common can be seen in Figure III-4.



The numbers down the left hand side represent the sequential order of elements of ITWO as they are stored. The numbers down the right hand side represent the array elements and their position in the "stack."

Figure III-10. ARRAY ITWO (4,4) As Stored By FORTRAN.

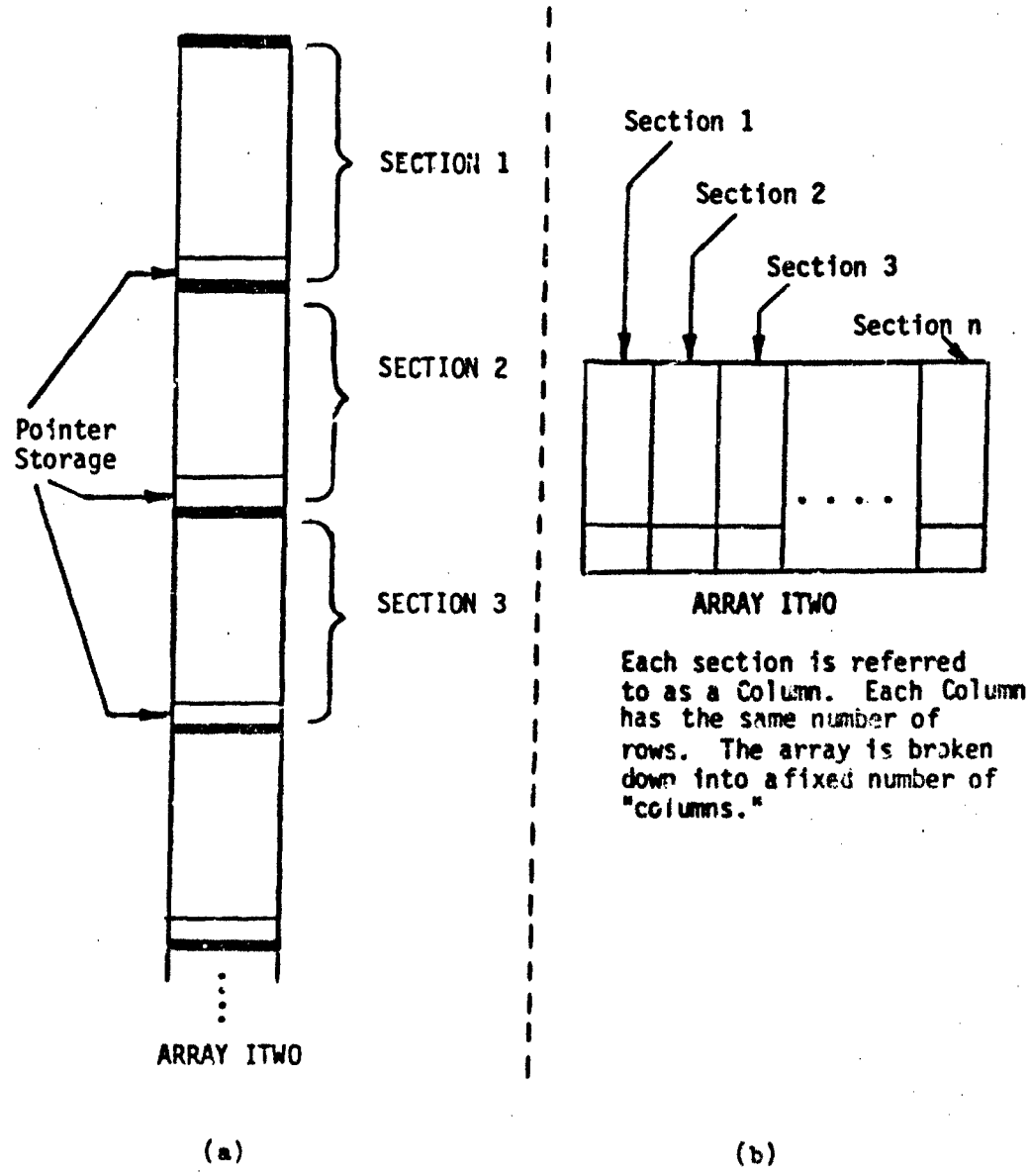


Figure III-11. Breakdown of a Pseudo-Two-Dimensional Array.

5-2. Pseudo-Two Dimensional Arrays.

In this section the information in the pseudo-two-dimensional arrays is described. This information may be accessed by system module user code at any time. If actions are taken at a task node in a system module which would cause any of these array elements to change values, the analyst must set the appropriate values in user code called by that task node.

5-2.a. Track Information. Track information is kept in the two arrays NTRACK and SS. Both are treated as pseudo-two-dimensional arrays, and both have the same number of columns. Each track has a column in NTRACK and a column in SS, and both arrays use the same column number. This column number will be used in system module user code. Some of the values in NTRACK will be set automatically by the Control system module, while the MOPADS analyst will set other values. The column assignments will be done automatically. The NTRACK array is always equivalenced to the array XTRACK so that each element may be either real or integer. MSAINT will automatically update the contents of the SS array. The SS array has no pointer row because it uses the same column number as NTRACK.

See Figure III-12 for a complete description of the track information storage arrays.

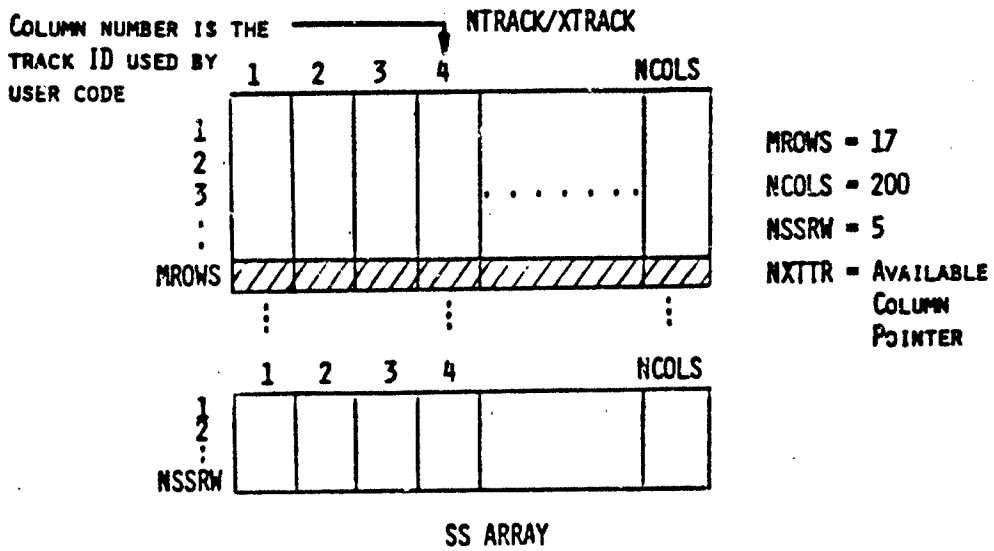
5-2.b. Site Information. A site is either a special radar, a transmittable Q-73, or a non-transmittable critical site, such as an airfield, ammunitions depot, encampment, or command center. Sites do not usually come and go, but they may. A site that was destroyed may be removed from the NSITE array. NSITE is equivalenced to XSITE in each routine which has the labeled common block/COM26/.

See Figure III-13 for a complete description of the site information storage arrays.

5-2.c. Fire Unit Information. A fire unit is an air defense unit armed with missiles or other firepower with the purpose of destroying enemy aircraft to protect itself and critical assets. Each fire unit has a column in the NFSTOR array.

See Figure III-14 for a complete description of the fire unit information storage arrays.

5-2.d. Common Information. Each track, site, or fire unit may have a five-character alphanumeric identifier. These are stored in the character array NTRID. A companion array NTRID2, which is dimensioned to the same number of columns as NTRID, contains cross-referencing information to find the appropriate data in the NTRACK, NSITE, or NFSTOR arrays given the alphanumeric identifier. Columns may be added or removed from NTRID/NTRID2 just as with the other



NTRACK ROW DEFINITIONS

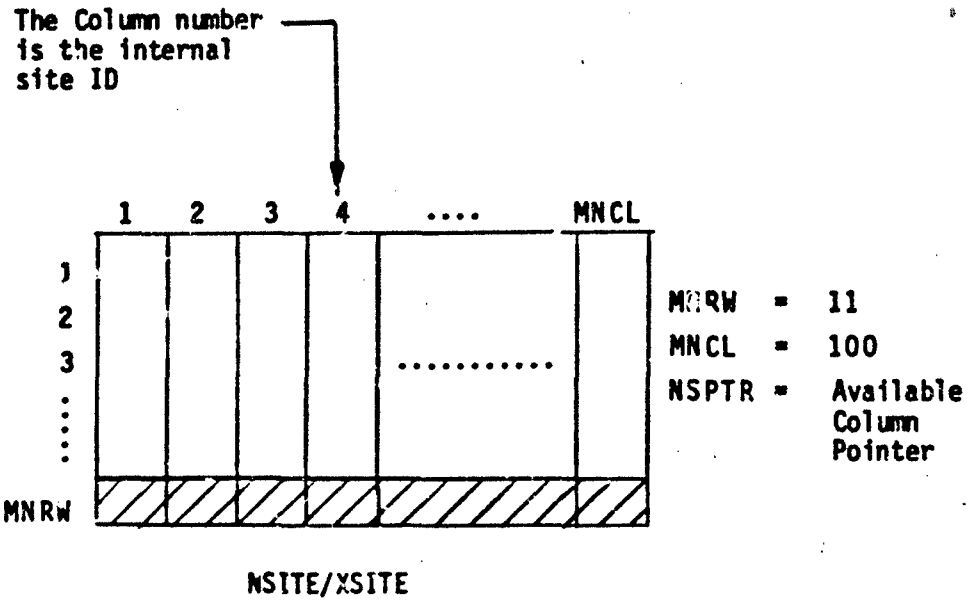
1. POINTER TO NTRID/NTRID2 ARRAY.
2. $\dot{X}$ - DIRECTION COMPONENT OF THE VELOCITY ($\dot{X}$ IN KNOTS)
3. $\dot{Y}$ - DIRECTION COMPONENT OF THE VELOCITY ($\dot{Y}$ IN KNOTS)
4. $\dot{Z}$ - DIRECTION COMPONENT OF THE VELOCITY ($\dot{Z}$ IN FT/MIN)
5. ACTUAL PRIMARY ID (1-FRIENDLY, 2-HOSTILE, 3-OTHER).
6. * ROW # IN DATA BASE FOR CHECKPOINT PROCESSING.
7. 8. IFF DATA (YET TO BE DEFINED, CURRENTLY UNUSED).
9. COPY ROW # OF THE UNIT THE TRACK IS LOCAL TO (0 IF NONE).
10. COLUMN IN NSITE OF THE VIEWER THAT SEES THE TRACK(0 IF UNSEEN).
11. CURRENT TRACK QUALITY (NOT DEFINED, UNUSED).
12. TRACK SIZE
13. * AIR SCENARIO TRACK ID.
14. AIRCRAFT TYPE
15. JAMMING (UNUSED, 1-YES, 2-NO).
16. IS THE END OF SEGMENT A TARGET?(1-YES, 2-NO)
17. POINTER VARIABLE FOR NEXT AVAILABLE COLUMN.

SS ROW DEFINITIONS

1. X POSITION
2. Y POSITION
3. Z POSITION
4. 5. UNUSED

* USED INTERNALLY, NOT TO BE ACCESSED OR SET BY THE ANALYST.

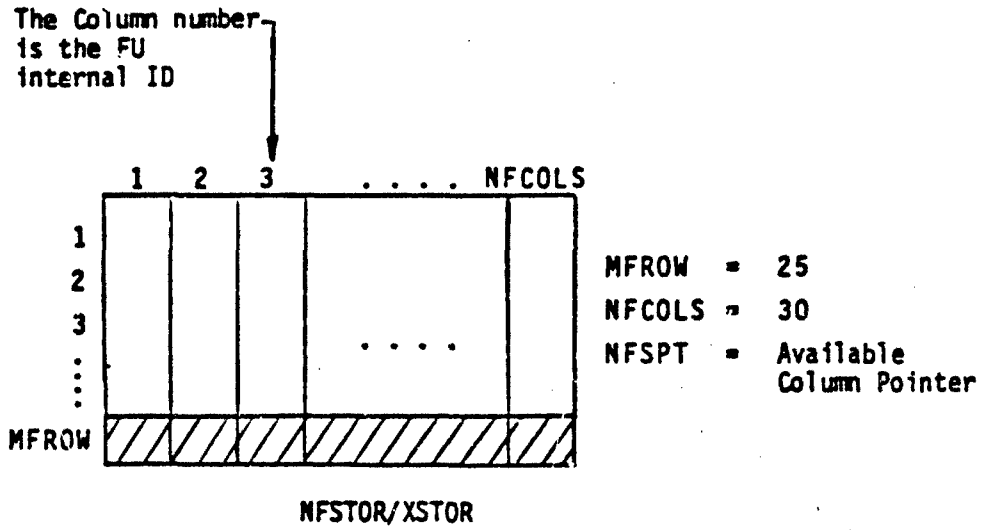
Figure III-12. Track Information Storage.



NSITE ROW DEFINITIONS

1. Pointer to NTRID/NTRID2 arrays.
2. Site type (1-Protected assets, 2-Remote Radar, 3-Air defense unit other than FU's)
3. X-coordinate
4. Y-coordinate
5. Z-coordinate
6. Transmittable (1-Yes, 0-No)
7. Active Status (1-Active, 0-Inactive)
8. Copy row number
   If type =1, of the FU assigned to protect this asset
   If type =2, of the unit owning the viewer (radar)
   If type =3, of the air defense unit
9. Pointer to NDBAD for viewer DB address
   If type =2, Ø otherwise
10. Viewer number, If type ≠ 1 or 3
11. Available column pointer

Figure III-13. Site Information Storage.



NFSTOR ROW DEFINITIONS

1. Pointer to the NTRID/NTRID2 arrays
2. Copy row number
3. System Module Type of the FU (e.g., 4-IHAWK)
4. X coordinate
5. Y coordinate
6. Z coordinate
7. Defended Sites (column numbers in NSITE array)
8. Defended Sites (column numbers in NSITE array)
9. Number of hot missiles
10. Number of cold missiles
11. Alert Status (unused)
12. Fire Section Status (for IHAWK, the codes are as follows: 1-Ready, 2-Weapon assigned, 3-Acknowledge, 4-Tracking, 5-Firing, 6-Effective, 7-Broken engagement, 8-Not effective, 9-Partially effective, 10-All missiles expended, 11-Out of action, 12-Hot missiles expended)
13. Primary track assignment (column number in NTRACK, 0 if none) (negative implies cover command)
14. Secondary track assignment (column number in NTRACK, 0 if none) (negative implies cover command)
15. Current track being engaged (column number in NTRACK, 0 if none)
16. Current command (for IHAWK, the codes are as follows: 1-Engage, 2-Engage ripple, 3-Investigate/Assign, 4-Cover, 5-Unused, 6-Hold fire, 7-Cease fire, 8-Cease engage, 0-No command)

FU Status of Fire Section A

17-24. Repeat of rows 9-16 for Fire Section B
25. Available column pointer

Figure III-14. Fire Unit Information Storage.

pseudo-two-dimensional arrays. Columns are pointed to by the first row element in either NTRACK, NSITE, or NFSTOR.

See Figure III-15 for a full description of the usage of the of the NTRID/NTRID2 arrays.

5-3. Copy, System Module, and Operator Data.

The MSAINT simulation program has a current entity it is processing. This entity is either an operator or a non-operator control entity. Each entity belongs to a copy, which has a copy row number (the row number in the NCOPY array). Each entity also has associated with it a system module type. Therefore, each entity, while it is current, also has a current copy row number and current system module number.

Several variables in common can be accessed by the user. These are defined in Table III-3.

Some of the copy-related information accessible to the MOPADS analyst is stored in the NCOPY array. See Figure III-16 for a complete description of the NCOPY array. It is not a pseudo-two-dimensional array. The copy row number referred to throughout this document is the row number in the NCOPY array.

Some operator-specific information is stored in the NOPRI array. It is defined in Figure III-17.

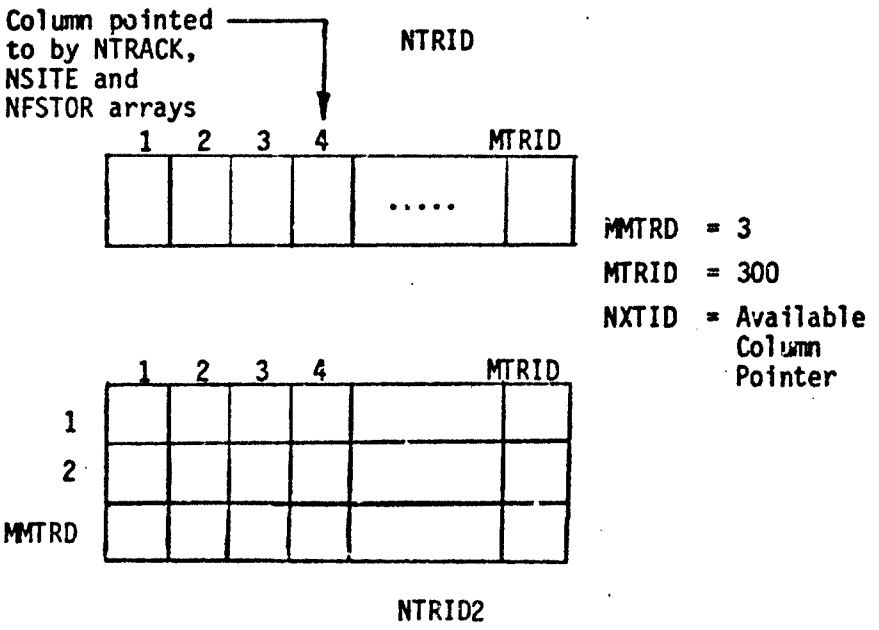
5-4. Display Information.

Each operator is assumed to have some type of visual display panel which provides information about the current air defense battle. These displays may have one or more options which affect how the system information is displayed and which information is displayed. These options are stored in the NSCRN array. Each operator may have up to 10 display options. The row dimension of the NSCRN array is the operator ID. The column definitions depend on which type of operator it is. Figure III-18 shows the structure of NSCRN and gives, as an example, the screen data for AN/TSQ-73 operators.

5-5. Data Base Addresses.

In order to reference data in the MOPADS data base (DB) from MSAINT, various DB addresses must be accessible. These are stored in several arrays which depend on the usage of the DB address.

Several types of DB addresses are stored in the NDBAD array. These are pointed to by the values in columns 20-22 of the NCOPY



NTRID ROW DEFINITION

1. 8-character alphanumeric label (NTRID is a character array)

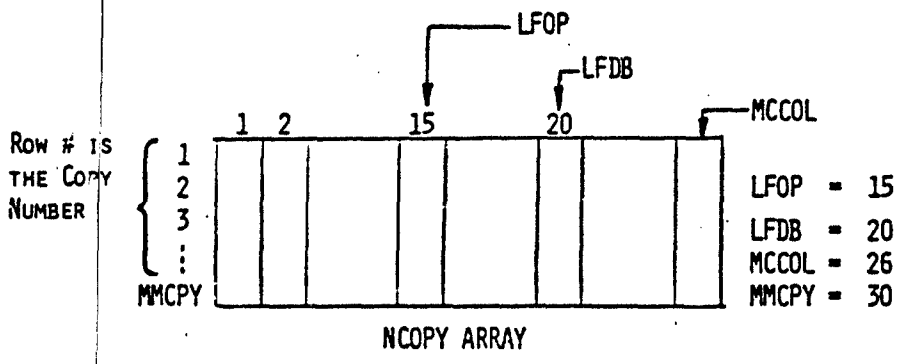
NTRID2 ROW DEFINITIONS

1. Type of air defense component (1-Track, 2-Site, 3-Fire unit)
2. Column number in NTRACK, NSITE, or NFSTOR.
3. Available column pointer

Figure III-15. Common Information Storage.

Table III-3. MSAINT Common Definitions

VARIABLE NAME	COMMON BLOCK	DEFINITION
MMCPY	COM25	Maximum copy number (currently - 30)
MMSYS	COM26	Maximum system module number (currently - 4)
NCURS	COM27	The current system module number during input. Not used during the execution of the simulation.
NCURCO	COM28	The current copy row number (for the current entity).
NCURSY	COM28	The current system module number (during the simulation).
ISTATZ	COM28	A flag used to indicate if the current entity is an operator or not =1 current entity is an operator =2 current entity is not an operator
NMCOP(4)	COM25	NMCOP(i) is the number of copies of system module type i
NEGID(150)	COM26	NEGID(i) is the copy number of the entity with ID = -i
MMNEG	COM26	Maximum array dimension of the NEGID array
NCNEXT	COM27	Next available row in NCOPY



ASSOCIATED POINTER VARIABLE DEFINITIONS

MMCPY - MAXIMUM NUMBER OF COPIES, USED AS THE MAXIMUM ROW DIMENSION (CURRENTLY = 30)

LFOP - POINTER TO THE FIRST COLUMN CONTAINING OPERATOR ID'S (CURRENTLY = 15)

LFDB - POINTER TO THE FIRST COLUMN CONTAINING DATA BASE ADDRESSES (CURRENTLY = 20)

MCCOL - MAXIMUM COLUMN DIMENSION OF NCOPY (CURRENTLY = 26)

COLUMN DEFINITIONS

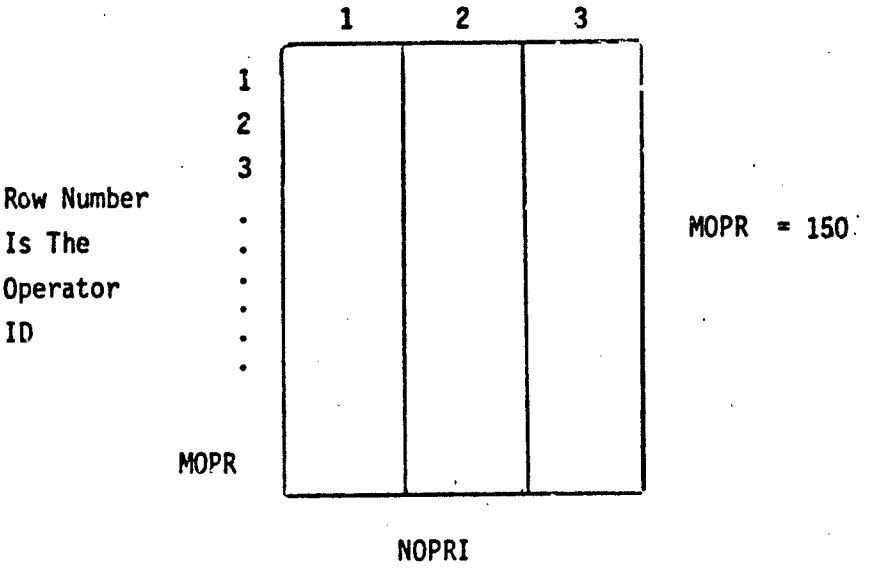
1. 4-DIGIT COPY IDENTIFIER
2. POINTER TO NSITE OR NFSTOR
   <0 = COLUMN NUMBER IN NSITE
   >0 = COLUMN NUMBER IN NFSTOR
3. SYSTEM MODULE TYPE (1-CONTROL, 2-GROUP Q-73, 3-BATTALION Q-73, 4-IHAWK)
4. HIGHER COMMAND (COPY ROW NUMBERS)

5-14. SUBORDINATE UNITS (COPY ROW NUMBERS)

15-19. OPERATOR ID'S FOR OPERATORS IN THIS COPY

20. POINTER TO TASK DATA DB ADDRESS IN THE NDBAD ARRAY
21. POINTER TO ESV DATA DB ADDRESS IN THE NDBAD ARRAY
22. POINTER TO MOPADS RESOURCE ADDRESS IN NDBAD
23. POINTER TO TRACK DATA DB ADDRESS IN NDBAD
24. INDEX OF THIS COPY WITHIN SYSTEM MODULE TYPE (1,2,...)
25. COPY SAMPLING OPTION (1-UNMODERATED DETERMINISTIC, 2-UNMODERATED STOCHASTIC, 3-MODERATED DETERMINISTIC, 4-MODERATED STOCHASTIC)
26. COLUMN NUMBER IN THE NTRID/NTRID2 ARRAYS

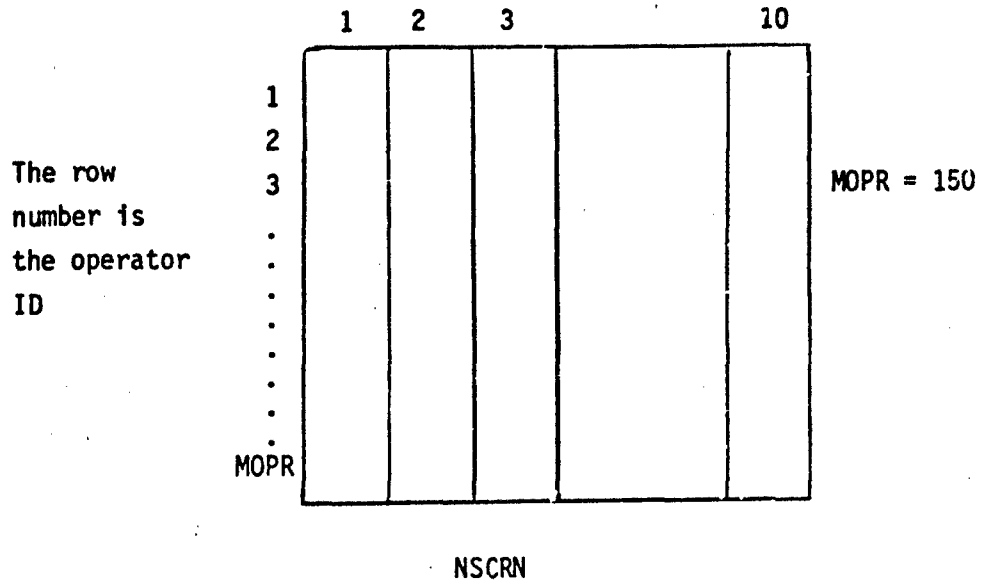
Figure III-16. NCOPY Array Description.



COLUMN DEFINITIONS

1. Copy row number of the copy that "owns" the operator.
2. Operator type =1 - Q-73 TD
   2 - Q-73 TDA
   3 - IHAWK TCO
   4 - IHAWK TCA
   5 - IHAWK ASO
   6 - IHAWK FCO A
   7 - IHAWK FCO B
   8 - GROUP TD1
   9 - GROUP TD2
3. Number of message currently addressed to the operator.

Figure III-17. NOPRI Array Description.



EXAMPLE: COLUMN DEFINITIONS FOR Q-73 OPERATORS

1. The current hooked item (column number in NTRID2)
2. Velocity vectors or TTG Vectors (Ø-Neither, 1-Velocity Vectors, 2-TTG Vecto
3. Screen Alphanumerics (Ø-off, 1-on)
4. Engagement Markers (Ø-off, 1-on)
5. Pairing Lines (Ø-off, 1-on)
6. Map (Ø-off, 1-on)
7. Screen Range (n.mi.)
8. X position of screen center
9. Y position of the screen center
10. Unused

NOTE: Elements 7, 8, and 9 of Display information must be defined as above for all system modules.

Figure III-18. NSCRN Array Description for Q-73 Operators.

array, see Figure III-16. NDBAD has two rows since these DB addresses require two words. The variable MXDBD is the maximum column dimension of NDBAD.

One 4-word DB address is stored in the NMES array. This address points to the message directory in the DB.

Two 4-word DB addresses for each copy are stored in the NDB4 array. For a full description of the NDB4 array see Figure III-19.

5-6. Miscellaneous Variables.

Several miscellaneous variables were added to MSAINT common to represent general information. These are defined in Table III-4.

6-0 MISCELLANEOUS ADDITIONS TO SAINT

Several miscellaneous changes were made to SAINT. These are described below.

1. Comment Cards

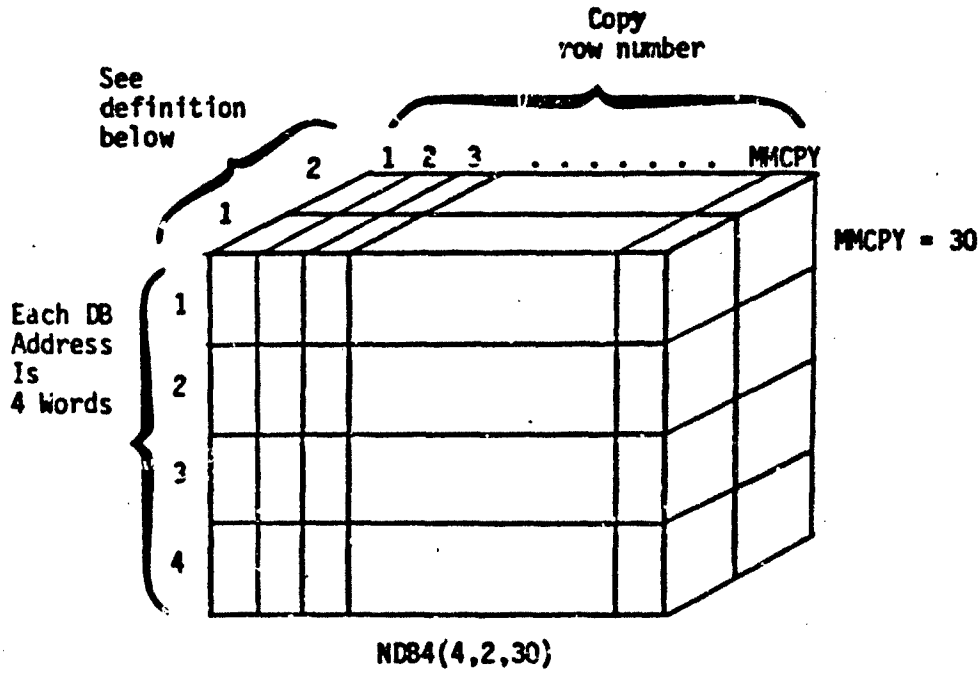
Formerly in SAINT, when a data card had an asterisk(*) in column 1, it was interpreted as a FIN card. This precluded the capability of inserting comment cards into the data input deck. This feature was modified so that now, if an asterisk is placed in column 1, the card is ignored. Thus, all comment cards must have an asterisk in column 1. Comments can also be placed on any data card after the asterisk which terminates the data fields.

2. New Trace Option

MSAINT no longer prints out an iteration trace for the beginning and end trace run. Instead, rows of data are written out to an external file when each significant event occurs. The MOPADS data base references this file to produce specialized traces as specified by the MOPADS user. See Polito (1983b) for a complete description of the MOPADS user trace output options.

3. Handling of Statistics

MSAINT no longer prints out any simulation statistic output. Instead, MSAINT merely collects the statistics in internal arrays. These arrays are then stored in the MOPADS data base either at the end of each run or at the end of the simulation. MOPADS uses these arrays to calculate and print out the statistics requested by the MOPADS user. The MOPADS user will always get standard statistics such as overall system performance measures, but most statistics will be optional. For a complete description of MOPADS statistic output options, see Polito (1983b).



TYPES OF DB ADDRESSES

(-,1-) - Copy directory DB address

(-,2,-) - DB address for messages that require responses

Figure III-19. NDB4 Array Description.

Table III-4. Miscellaneous Variable Definitions.

TSTRTK	Start time of the simulation (minutes)
TFINK	End time of the simulation (minutes)
DELT	Time between checks for updates on aircraft checkpoints.
LASTM	The last message number used. (Message numbers are internally assigned.)
NXMON	Month of the simulation run.
NXDAY	Day of the simulation run.
NXYR	Year of the simulation run.
NXPROJ	8-character project name
NXNAME	8-character MOPADS user name

IV. REFERENCES

Polito, J. The MOPADS data base application programs (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(a).

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(b).

Polito, J. & Goodin II, J. R. MOPADS utility programs (MOPADS/UI) (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Wortman, D. B., Duket, S. D., Seifert, D. J., Hann, R. L. & Chubb, G. P. The SAINT user's manual (AMRL-TR-77-62). Pritsker & Associates, Inc., prepared for Aerospace Medical Research Laboratory, Wright-Patterson AFB, OH, 1978(a).

Wortman, D. B., Duket, S. D., Seifert, D. J., Hann, R. L. & Chubb, G. P. Simulation using SAINT: a user-oriented instruction manual (AMRL-TR-76-61). Pritsker & Associates, Inc., prepared for Aerospace Medical Research Laboratory, Wright-Patterson AFB, OH, 1978(b).

Wortman, D. B., Duket, S. D., Seifert, D. J., Hann, R. L. & Chubb, G. P. Documentation for the SAINT simulation program (AMRL-TR-77-63). Pritsker & Associates, Inc., prepared for the Aerospace Medical Research Laboratory, Wright-Patterson AFB, OH, 1978(c).

Wortman, D. B. & O'Reilly, J. J. New capabilities in the SAINT simulation program. Pritsker & Associates, Inc., West Lafayette, IN, 1982.

DISTRIBUTION LIST

PERI-IB (5)
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.#1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service - Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St. Room 108
Boulder, Colorado 80301

CHANGE NOTICES

MOPADS Volume 4.5

CHANGE
No. 1

Date:

Document Title: MSAINT User's Guide: Changes and Additions

Contract ARI Field Unit, MDA903-81-C-AA06

1. Remove old pages and insert new pages as indicated below.

Remove Pages	Insert Pages
Inside cover page	Inside cover page
DD1473	DD1473
ix,x	ix,x
xi,xii	xi,xii
I-1	I-1
II-3,II-4	II-3,II-4
II-5,II-6	II-5,II-6
II-9,II-10	II-9,II-10
II-11,II-12	II-11,II-12
III-5,III-6	III-5,III-6
III-9,III-10	III-9,III-10
III-13,III-14	III-13,III-14
III-21,III-22	III-21,III-22
III-25,III-26	III-25,III-26
III-27,III-28	III-27,III-28
III-29,III-30	III-29,III-30
III-31,III-32	III-31,III-32
III-33,III-34	III-33,III-34
IV-1	IV-1
	Add this page

2. File this change sheet in the back of the publication under Change Notices.

APPENDIX I

MOPADS FINAL REPORT:

CREATING REFERENCE AIR DEFENSE SYSTEM MODULES

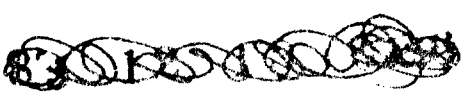


TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

LIST OF FIGURES

LIST OF TABLES

TERMINOLOGY

1-0 STANDARD MOPADS TERMINOLOGY.

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

2-0 OTHER TERMINOLOGY.

ACC	ADSM Code Character.
ADSM	Air Defense System Module.
DATA BASE	The collection of user information and control information that is processed by the DBCS.
DATA BASE FILE	A computer file stored on disk that contains the data base.
DATA LIST	A list of values (character or numeric) that is user specified and is stored in a data base.
DB	Data Base.
DBCS	Data Base Control System.
DBF	Data Base File.

DIRECTORY	A collection of data lists and other directories.
DL	Data List.
DR	Directory.
ID	Identify. DL's and DR's have identifiers that are lists of integers and are stored in their owning directory.

I. INTRODUCTION

1-0 PURPOSE.

This document describes the procedure for entering reference Air Defense System Modules (ADSM) into a MOPADS data base. It's a user manual for the "Create/Edit Reference System Module" subprocess of the MOPADS user interface. The need for this subprocess arises when a MOPADS modeler has developed a MOPADS model of an air defense system. Guidelines for developing system modules is described in MOPADS Volume 4.3 and 4.4 (Walker & Polito (1983a,b).

When the information for the system module has been developed, the MOPADS modeler is ready to enter the information into a MOPADS data base. That is the purpose of the "Create/Edit Reference System Module" subprocess. This subprocess has facilities for entering and editing all of the data associated with a system module that is contained in the data base. In particular, the following information is entered and edited with this subprocess:

1. Operator State Vectors
2. Environment State Vectors
3. Task Data
4. System Resource Data

The system module should be completely designed, the MSAINT model developed, and the user code written before this subprocess is used to enter data in the data base. The next section describes the information needed to create a reference ADSM.

2-0 INFORMATION NEEDED.

The data required to enter a reference ADSM are contained in Figures I-1 through I-6. The operator numbers and labels are from (1) and (2) in Figure I-1. If system hardware resources are included in the model (as opposed to MSAINT resources) these may be found in (3) of Figure I-1 or (5) of Figure I-2, depending upon how the MOPADS modeler has chosen to enter them.

Each operator will be a SAINT entity, and the node where he is created can be found in Figure I-2, (4). Goal defintiions and parameters will have been entered in the forms in Figures I-3 and I-4. Each system operator may have a set of display parameters which will be found in Figure I-5.

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
(1)	(2)		(3)

NAME:
DATE:

AIR DEFENSE SYSTEM MODULE:
PROJECT:

OPERATOR DEFINITIONS

Page____of____

Figure I-1. Operator Data.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
(4)					(5)

Name:
Date:

AIR DEFENSE SYSTEM MODULE:
PROJECT:

SAINT ENTITIES

Page___of___

Figure I-2. Additional Operator Data.

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
NAME: DATE:	AIR DEFENSE SYSTEM MODULE: PROJECT:	
OPERATOR GOALS DEFINITIONS		Page___of___

Figure I-3. Goal Definitions.

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY

OPERATOR:	NUMBER OF GOALS CONSIDERED:
NAME: DATE:	AIR DEFENSE SYSTEM MODULE: PROJECT:
OPERATOR GOAL SPECIFICATION	Page ___of___

Figure I-4. Operator Goal Specifications.

PARAMETER NUMBER	DESCRIPTION	INITIAL VALUE

NAME: DATE:	AIR DEFENSE SYSTEMS MODULE: PROJECT: OPERATOR:

DISPLAY DATA Page___of___

Figure I-5. Display Data.

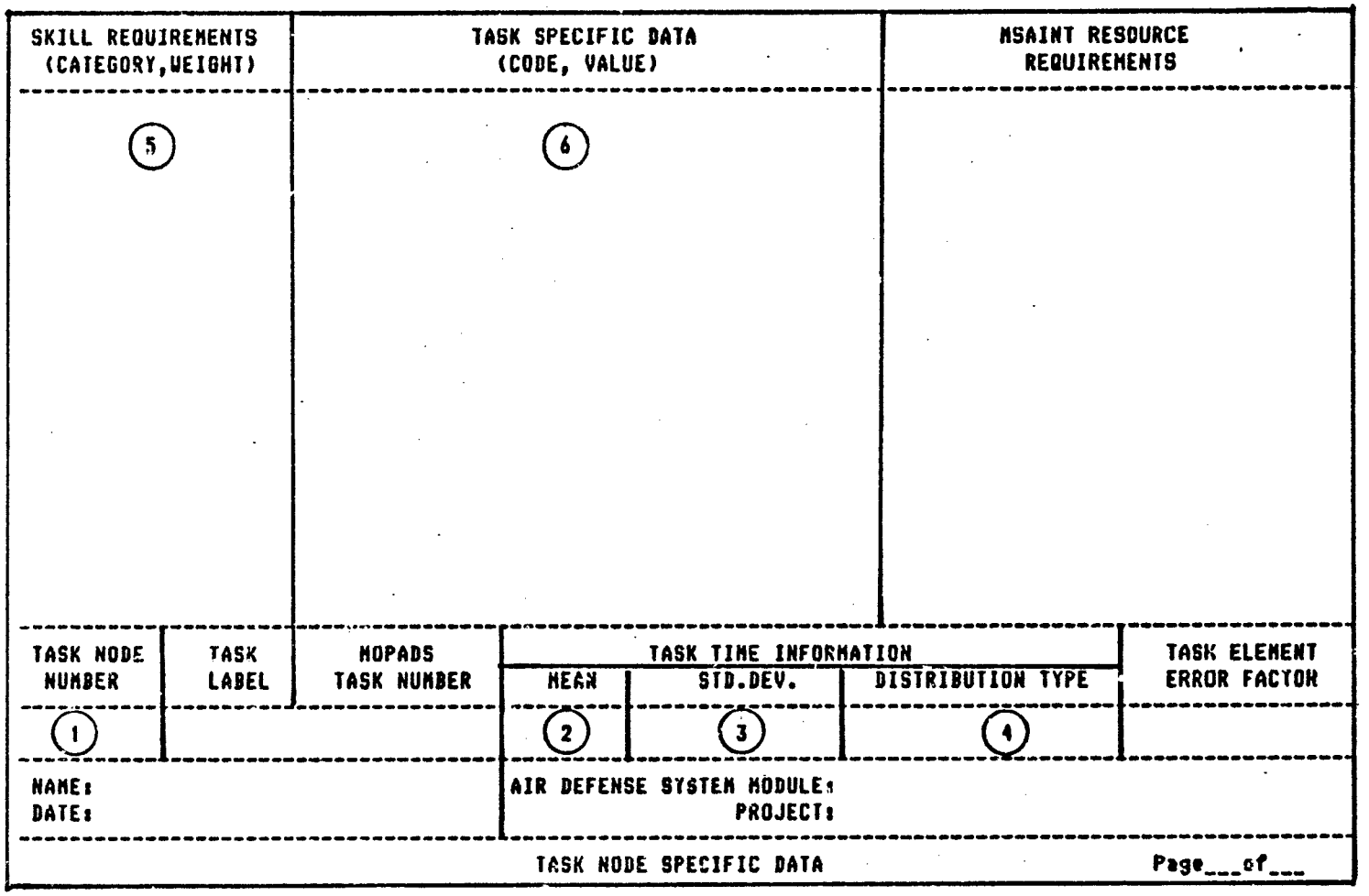


Figure I-6. Task Element Data.

Finally, each Task node will have an associated form shown in Figure I-6 from which Task data will be entered into the data base. The required data are noted by (1) through (6).

When the system module is created, default values for parameters in the operator and environmental state vectors will be loaded into the data base. See Laughery & Gawron (1983). Changes to these defaults may be entered using the editing features of the subprocess.

Tables I-1 and I-2 show the structure of portions of the MOPADS data base of concern. The REFERENCE-ADSM directory, Table I-1, is owned by the master directory and it owns all of the reference ADSM directories. These code 11 directories are shown in Table I-2. The various data lists in these directories are explained below.

TD-TASK-DATA	-	This data list contains information for each task node in the model of the system module. It contains one row per task node.
R-ADS-RESOURCES	-	This data list contains information on each hardware resource defined for the ADSM.
RL-RESOURCE-LABELS	-	This data list contains the label of each defined hardware resource.
OP-OPERATOR-STATE-VECTOR	-	These data lists are the operator state vectors for the operators.
EN-ENVIRONMENT-STATE-VEC	-	This is the environmental state vector for the ADSM.
OT-OPERATOR-TYPES	-	This data list contains the operator labels and the task nodes where they are created.

MOPADS Volume 5.17 (Polito (1983a)) contains details of the contents of these data lists.

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 2

DIRECTORY LABEL: REFERENCE-ADSM

TYPE	CODE	LABEL	NUMBER
DL	-	SKILL-CATEGORIES	1
DL	-	TASK-SPECIFIC-CATEGORIES	1
DL	-	ADSM-LABELS + ACC	1
DR	11	User specified reference ADSM labels	as needed

Table I-1. Contents of REFERENCE-ADSM Directory.

MOPABS/DIRECTORY CONTENTS

DIRECTORY CODE: 11

DIRECTORY LABEL: User Specified Reference ADSN Label

TYPE	CODE	LABEL	NUMBER
DL	-	TB-TASK-DATA	1
DL	-	R-ABS-RESOURCES	1
DL	-	RL-RESOURCE-LABELS	1
DL	-	OP-OPERATOR-STATE-VECTOR	as needed
DL	-	EN-ENVIRONMENT-STATE-VEC	1
DL	-	OT-OPERATOR-TYPES	1

Table I-2. Contents of Code 11 Directories.

II. COMMANDS AVAILABLE

1-0 BASIC DATA BASE COMMANDS.

The basic data base commands that are available in all subprocess of the User Interface are also available in this subprocess. Descriptions of these commands are repeated in Appendix A for easy reference.

2-0 THE ADD COMMAND.

The specification for the ADD command is shown in Table II-1. The ADD command creates a new reference ADSM directory with the label specified by the response to the prompt "ADSM." The label must be unique and must have the form of a one character ACC followed by a dash (-) followed by the ADSM label (e.g., "Q-BATTALION-Q73"). The ADD command will query the user for information to complete the creation of the system module.

The ADD command creates the shell of a system module with all default parameters and no task or resource data entered. This additional information and changes to defaults are entered with the CHANGE command. The ADD command leaves the created ADSM as the current directory.

3-0 THE CHANGE COMMAND.

The CHANGE command is shown in Table II-2. The single prompt requests the type of data in the current reference ADSM to edit. Depending upon the response (TD, R, OP, or EN), the user enters an editor to modify that type of data.

Prior to exiting the CHANGE command, the user is asked if a line printer list is desired of the particular DL. This feature permits hard copy records of the entire reference ADSM to be maintained and cross-checked with desired values from Section I. The specified data list becomes the current data list and remains current on exit from the CHANGE command.

4-0 THE DELETE COMMAND.

The DELETE command is shown in Table II-3. The DELETE command will irrevocably remove an entire reference ADSM from the REFERENCE

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: ADD

MODULE: UI/S

REVISION DATE: 19 MAR 83

PURPOSE: To create a new reference ADSN in the REFERENCE ADSN directory. The new ADSN becomes current.

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
ADSN (Label for the ADSN preceded by a one-character ACC)	3001	S				

Table II-1. ADD Command Specification.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: CHANGE

MODULE: UI/5

REVISION DATE: 9 MAR 83

PURPOSE: To change elements of a specified data list of the current reference ADSM

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DATA-LIST	3001	4		TD, R, OP, EN TD - task data R - resources OP - operator state vector EN - environmental state vector	--	--

Table II-2. CHANGE Command Specification.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: DELETE

MODULE: UI/5

REVISION DATE: 9 MAR 83

PURPOSE: To delete a reference ADSM from the reference or working DB

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
ADSM (ADSM label)	3001	5	--	--	--	--
DB	3001	4	Working	Working Reference	--	--

Table II-3. DELETE Command Specification.

ADSM directory. Use it with caution. The response to the "ADSM" prompt is the label of the reference ADSM to be deleted. The DELETE command will function on either the working or reference data base.

5-0 THE GET COMMAND.

The intent of the MOPADS DBCS system (Polito (1983b)) is that the reference data base be used to store stable copies of information such as reference ADSM's. When needed in a working setting to perform particular simulations, the information can be transferred to a working DB. The GET command (see Table II-4) copies an entire reference ADSM from the reference DB to the working DB.

The response to the single prompt is an ADSM label. It is copied to the working DB with the same name and will replace an existing ADSM with the same name. The user, however, will be prompted for approval to overwrite an existing ADSM.

6-0 THE RENAME COMMAND.

The RENAME command specification is shown in Table II-5. RENAME will change the label of a Reference ADSM on the working DB. Note that the ADSM Code Character (ACC) may not be changed. Only the portion of the ADSM label following the ACC may be changed. See Section II, 2-0 for a description of ADSM labels.

7-0 THE SAVE COMMAND.

The SAVE command (Table II-6) performs the reverse operation of the GET command. It copies an entire reference ADSM from the working to the reference DB. It will also overwrite an ADSM of the same name on the reference DB (but it will ask permission first). The response to the single prompt is the label of the ADSM to save.

8-0 THE USE COMMAND

The USE command (Table II-7) makes a specified reference ADSM the current directory on the specified DB. Certain commands operate on the "current" directory or the "current" data list. For example, the CHANGE command edits the current reference ADSM. Most commands will leave an ADSM current. For example, the ADD command leaves the created ADSM current. The USE command is a way to specify which reference ADSM should be current.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: GET

MODULE: UI/5

REVISION DATE: 9 MAR 83

PURPOSE: To copy a reference ADSM from the reference DB to the working DB (with replacement)

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
ADSM (an ADSM label)	3001	5	--	--	--	--

Table II-4. GET Command Specification.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: RENAME

MODULE: UI/5

REVISION DATE: 9 MAR 83

PURPOSE: To rename (assign a new label) to a reference ADSM on the working DB. ADSM code character (ACC) may not be changed. Only the subsequent part of the label can be changed.

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
OLD (an existing ADSM label)	3001	5	--	--	--	--
NEW (a new ADSM label)	3001	5	--	--	--	--

Table II-5. RENAME Command Specification.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: SAVE

PURPOSE: To copy a reference ADSM from the working DB to the reference DB with replacement.

MODULE: UI/5

REVISION DATE: 9 MAR 8

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
ADSM (ADSM label)	3001	5	--	--	--	--

Table II-6. SAVE Command Specification.

COMMAND DATA SPECIFICATION — Page 1 of ;

COMMAND: USE

PURPOSE: To make a specified ADSM current on the specified DB.

MODULE: UI/5

REVISION DATE: 9 MAR 83

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
DB	3001	1	Working	Working Reference		
ADSM (a defined ADSM label)	3001	5	--	--	--	--

Table II-7. USE Command Specification.

It is always possible to determine which ADSM is current with the CURRENT command (Appendix A) and to see which ADSM's are available with the DIRECTORY (also Appendix A) command. USE is provided as a convenience for this subprocess; it is a specialized version of the more general basic command, SELECT (once again, see Appendix A).

III. USER INSTRUCTIONS AND EXAMPLES

1-0 A SIMPLE EXAMPLE.

The contents of Table III-1 present a brief example to demonstrate creating and editing a reference ADSM. Only those portions of the forms needed to create the system module are filled in and all of this data is hypothetical.

A reference system module labeled "E-EXAMPLE" is described. It's ACC is "E." Two operators with labels "MISSION-DIRECTOR" and "MISSION-COORDINATOR" are defined. All of the required data for these operators and data for three Task nodes are contained in the forms contained in Table III-1.

The subsequent sections contain sample terminal sessions with MOPADS. Input typed by the user is enclosed in boxes to distinguish it from information printed by MOPADS.

2-0 THE ADD COMMAND.

Figure III-1 shows an example of the ADD command. The annotations are explained below.

(1) - This section shows the initial interchange with MOPADS. MOPADS requests the name of the working DB and asks if it is a previously created data base file. If not, the file is initialized to be an empty MOPADS data base.

WARNING

All information on a data base file will be lost if the user answers "NO" to this question.

If a new DB is being created, the user may specify title information.

(2) - The user must select a subprocess. In this case, select subprocess 5, the "CREATE/EDIT REFERENCE SYSTEM MODULE" subprocess.

OPERATOR NUMBER	OPERATOR TITLE	MISSION AND DUTIES	EQUIPMENT
1	MISSION-DIRECTOR		
2	MISSION-COORDINATOR		

NAME: Polito	AIR DEFENSE SYSTEM MODULE: E-EXAMPLE
DATE: MAY 1983	PROJECT: Vol. 4.6

OPERATOR DEFINITIONS Page 1 of 1

Table III-1. Data for Example.

NODE(S) WHERE CREATED	DESCRIPTION	ATTRIBUTE NUMBER	INFORMATION ATTRIBUTES		RESOURCE REQUIREMENTS
			DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
22	MISSION-DIRECTOR				1,DATA-LINK(1) 2,CON-SCREEN(1)
18	MISSION-COORDINATOR				3,DATA-LINK(1)

Name: POLITO Date: MAY 1983		AIR DEFENSE SYSTEM MODULE: E-EXAMPLE PROJECT: Vol. 4	
	SAINT ENTITIES		Page 1 of 1

Table III-1. (continued)

PARAMETER NUMBER	DESCRIPTION	INITIAL VALUE
1	HOOKED-TRACK	
2	PRIMARY-TARGET	
3	SECONDARY-TARGET	

NAME: Polito DATE: MAY 1983	AIR DEFENSE SYSTEMS MODULE: E-EXAMPLE PROJECT: Vol. 4.6 OPERATOR: MISSION-DIRECTOR & MISSION COORDINATOR

DISPLAY DATA Page 1 of 1.

Table III-1. (continued)

GOAL NUMBER	DESCRIPTION	EVALUATION OF GOAL STATE
1	Reduce Number of Unknown Tracks	Number of Unknown Tracks
2	Engage Nearby Targets	Distance to Nearest Unengaged Target
3	Respond to all Messages and Queries	Number of Outstanding Messages and queries

NAME: Pelito
DATE: MAY 1983

AIR DEFENSE SYSTEM MODULE: E-EXAMPLE
PROJECT: Vol. 4.6

OPERATOR GOALS DEFINITIONS Page 1 of 1

Table III-1. (continued)

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
1	(INF,0)	-	-	-	-	1	1	3	3
2	(25,+INF)	0	30	8	10	-	-	-	-

OPERATOR: MISSION-DIRECTOR	NUMBER OF GOALS CONSIDERED: 2
NAME: Polito DATE: MAY 1983	AIR DEFENSE SYSTEM MODULE: E-EXAMINE PROJECT: Vol. 4.6

OPERATOR GOAL SPECIFICATION Page 1 of 1

Table III-1. (continued)

GOAL NO.	RANGE OF SATISFACTION	LOW POINT 1		LOW POINT 2		HIGH POINT 1		HIGH POINT 2	
		STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY	STATE	PRIORITY
3	(-INF,0)	-	-	-	-	1	.2	5	3
OPERATOR: MISSION-COORDINATOR				NUMBER OF GOALS CONSIDERED: 1					
NAME: Polito DATE: MAY 1983				AIR DEFENSE SYSTEM MODULE: E-EXAMPLE PROJECT: Vol. 4.6					
OPERATOR GOAL SPECIFICATION								Page 1 of 1	

Table III-1. (continued)

SKILL REQUIREMENTS (CATEGORY,WEIGHT)		TASK SPECIFIC DATA (CODE, VALUE)		MSAINT RESOURCE REQUIREMENTS
13	.8	5	01	
17	.2	6	5	

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
10			.1	.035	8	

NAME: Polito
DATE: MAY 1983

AIR DEFENSE SYSTEM MODULE: E-EXAMPLE
PROJECT: Vol. 4.6

TASK NODE SPECIFIC DATA Page 1 of 1

Table III-1. (continued)

SKILL REQUIREMENTS (CATEGORY,WEIGHT)		TASK SPECIFIC DATA (CODE, VALUE)		MSAINT RESOURCE REQUIREMENTS
18	.6	3	10	
19	.2	5	10	
21	.2	6	5	

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
14			.7	.25	8	

NAME: Polito
DATE: MAY 1983

AIR DEFENSE SYSTEM MODULE: E-EXAMPLE
PROJECT: Vol. 4.6

TASK NODE SPECIFIC DATA Page 1 of 1

Table III-1. (continued)

SKILL REQUIREMENTS (CATEGORY,WEIGHT)		TASK SPECIFIC DATA (CODE, VALUE)		MSAINT RESOURCE REQUIREMENTS
2	.4	3	10	
8	.4	5	10	
12	.2	6	5	

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
16			1.1	.39	8	

NAME: POLITO
DATE: MAY 1983

AIR DEFENSE SYSTEM MODULE: E-EXAMPLE
PROJECT: Vol. 4.6

TASK NODE SPECIFIC DATA Page 1 of 1

Table III-1. (continued)

```
WELCOME TO MOPADS VERSION DATED 31 JULY 1983, CHANGE 0.

TYPE HELP ANYTIME IN RESPONSE TO 'ENTER COMMAND'
TYPE THE NAME OF THE WORKING DB
VOL46.DBF                                                        (1)
IS THIS AN OLD DB
N
ENTER RECORD LIMIT(ZERO FOR NONE)
0
TYPE TITLE INFORMATION IN 40 COLUMN LINES
TYPE AS MANY LINES AS NEEDED.
ENCLOSE TITLE LINES IN DOUBLE QUOTES(")
START A LINE WITH '.QUIT' TO STOP
'EXAMPLE TO DEMONSTRATE'
'CREATE/EDIT REFERENCE ADSM'
'.QUIT'
DO YOU WANT TO REVIEW THE TITLE?
Y
1 EXAMPLE TO DEMONSTRATE
2 CREATE/EDIT REFERENCE ADSM
DO YOU WANT TO CHANGE THIS BLOCK?
N
DO YOU WANT TO REVIEW THE TITLE?
N
NEW MOPADS DB FILE SUCCESSFULLY CREATED
SELECT OPTION

 0-TERMINATE
 1-CREATE/EDIT SIMULATION DATA SET
 2-SET UP SIMULATION RUN DATA
 3-EXAMINE STATISTICS
 4-CREATE/EDIT SCENARIO DATA
 5-CREATE/EDIT REFERENCE SYSTEM MODULE
TYPE OPTION NUMBER                                               (2)
5
CREATE/EDIT REFERENCE SYSTEM MODULE
ENTER CR/ED ADSM COMMAND                                         (3)
MENU
COMMANDS CURRENTLY AVAILABLE:
ADD             CHANGE          DELETE          GET
QUIT            RENAME          SAVE            USE
CLOSE           CURRENT         DEPOSIT         DIRECTORY
EXAMINE         HELP            OPEN            SELECT
TERMINATE       PLINK           MENU
ENTER CR/ED ADSM COMMAND
HELP
COMMAND [NO DEFAULT]=                                            (4)
ADD

**********:**********************************

COMMAND : ADD

PROMPTS :  ADSM

ENTER THE NAME OF A PROMPT (# TO TERMINATE) =
ADSM

PROMPT : ADSM
   CHARACTER
   LIST OF LENGTH :   1
   NO DEFAULT VALUES
```

Figure III-1. ADD Command Example.

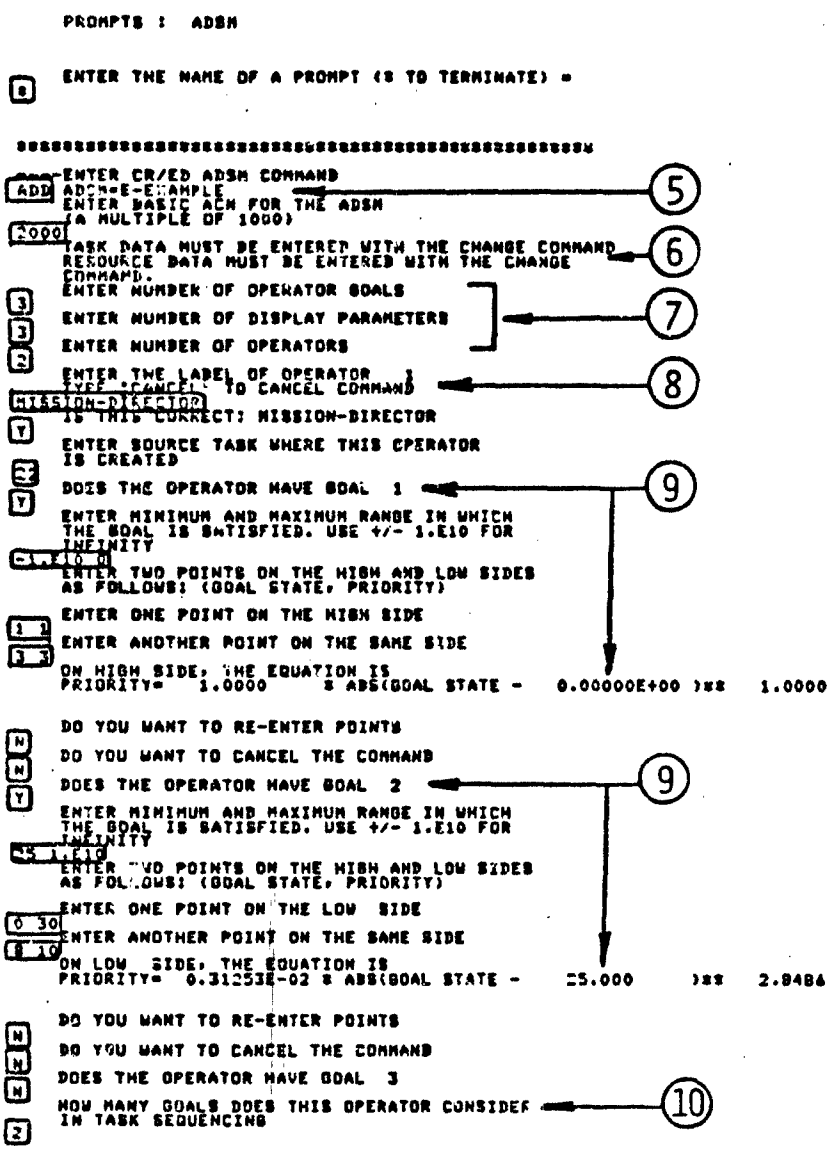


Figure III-1. (continued)

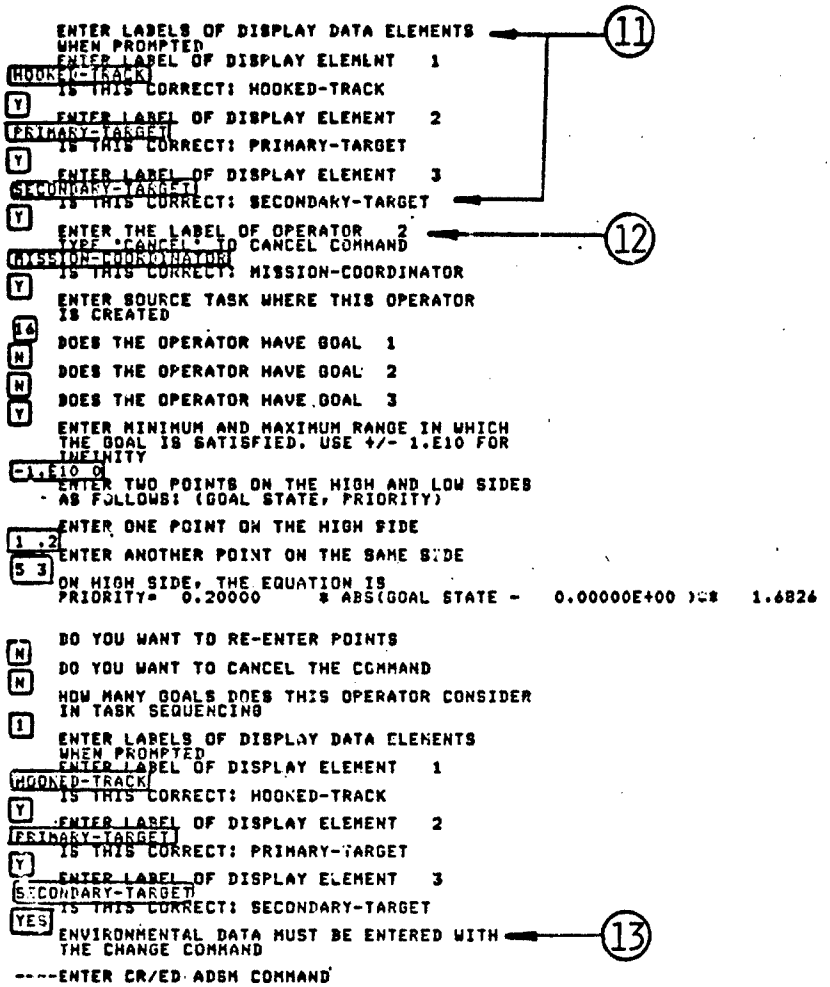


Figure III-1. (continued)

③ - The command prompt is

-----ENTER CR/ED ADSM COMMAND

In any subprocess, the MENU command will list all commands currently available. The subprocess commands are printed (ADD through USE) followed by the basic data base commands that are available in all subprocesses (CLOSE through MENU).

④ - In any subprocess, the HELP command can be issued to print information on the prompts for any command.

⑤ - The ADD command is issued to create the example ADSM. MOPADS queries for the basic ACN. At this time, MOPADS checks for duplicate use of the ACC ("E") and ACN and will re-prompt or cancel the command if needed.

⑥ - Task node data (the "TD" data list) and resource data (the "R" and "RL" data lists) must be entered with the CHANGE command.

⑦ - MOPADS asks for the number of operator goals, the number of display parameters, and the number of operators. A single set of goals is permitted for the ADSM operators. As will be seen, however, each operator need not have all of the goals, so it is possible to have operators with different goal sets in a single ADSM. Similarly, each operator may have different display parameters, but the maximum number of display parameters for any operator must be specified here.

⑧ - Begin input for operator 1. The operator's label and the MSAINT Task node at which he is created must be specified.

⑨ - Goal input for the operator is requested here. Note that the user is queried separately for each goal. Also, since the range of satisfaction of the goals for this example have one end point at $\pm$ infinity, (goal state, goal priority) pairs are requested only for the non-infinity side. MOPADS performs internal consistency checks on the entered values.

⑩ - The operator may only consider the n most dissatisfied goals in task sequencing. Enter n here.

⑪ - The labels of the display parameters for this operator are entered here. The display parameters are those which the MOPADS modeler has represented in his MSAINT model of the ADSM.

⑫ - Repeat for operator 2.

⑬ - Default values for environmental state vectors and operator state vectors have been set by MOPADS. They can be edited with the CHANGE command.

At this point, MOPADS has created the reference ADSM. Some of the data lists have not been populated yet, however. See Table I-2. As discussed above, the "TD," "R", and "RL," data lists are empty, and the "OP" and "EN" data lists have default human factors values.

3-0 THE CHANGE COMMAND.

3-1. CHANGE Task Data.

The CHANGE command has subeditors to change each of the data lists in an ADSM. This section describes editing of the TD-TASK-DATA data list. Figure III-2 is an interactive session to edit this data list. The annotations are explained below.

① - The CHANGE command specifies that the "TD" (task data) data list is to be edited. Recall that the ADD command creates this DL but it is empty. Information for each TASK node that is to have its task time moderated or which specifies system resources must be entered explicitly with this command.

② - Data is entered by specifying a code character for the operation to be performed. The code characters are:

S - show parameters for a node

M - print the menu of code characters

E - enter data for a new node. "E" may be used only once for any node to create it

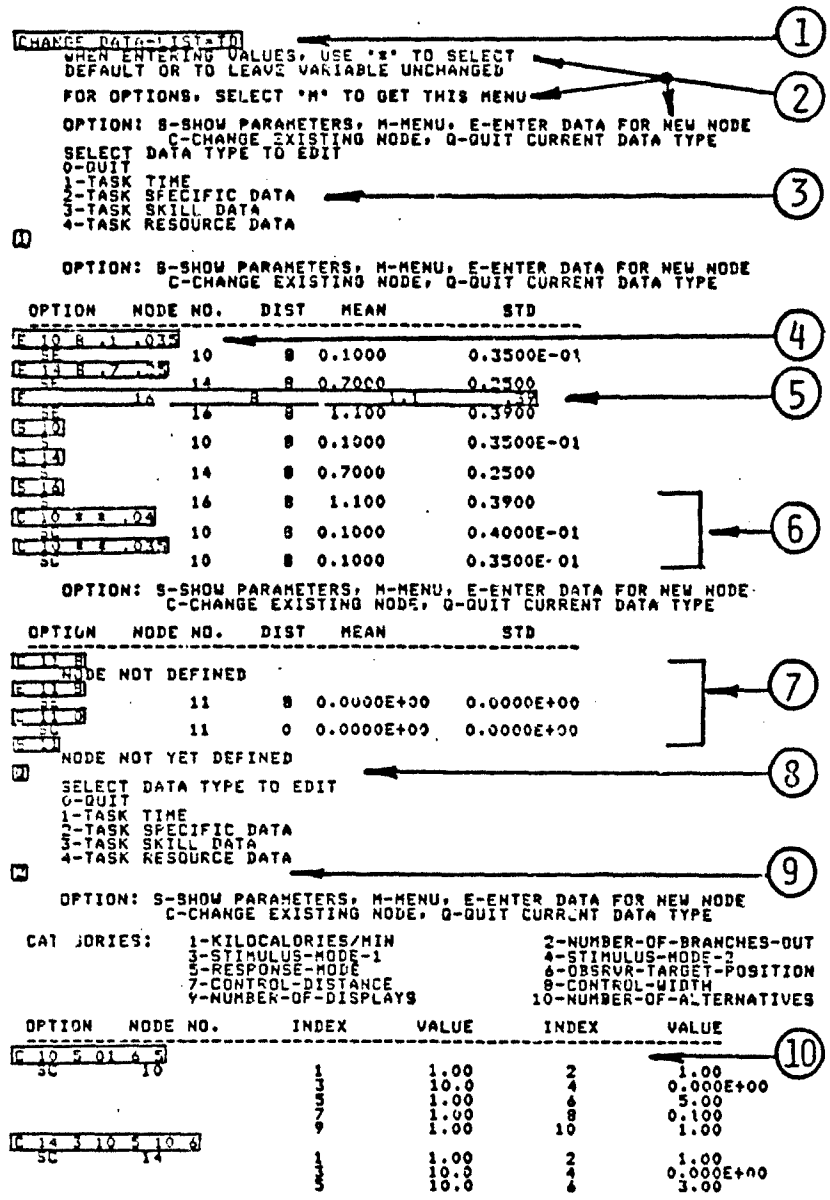


Figure III-2. CHANGE Task Data Example.

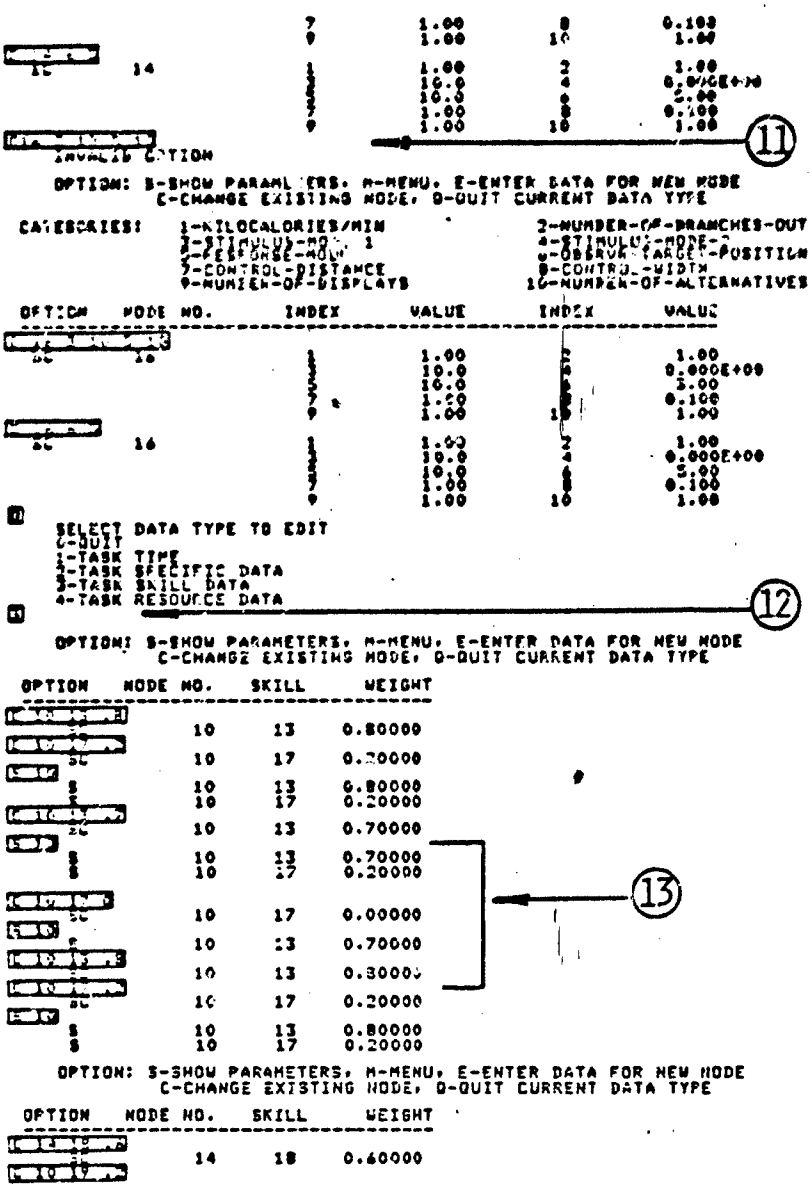


Figure III-2. (continued)

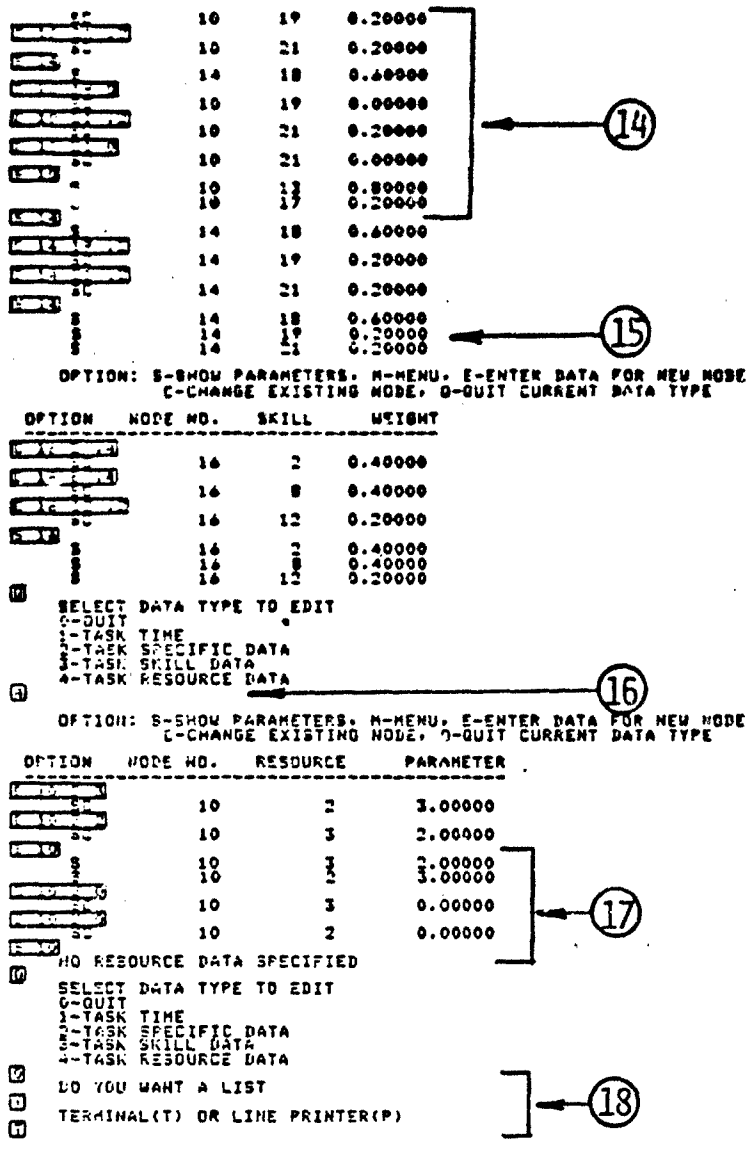


Figure III-2. (continued)

C - change parameters for an existing node. "C" must be used for all edits of a node after it is created with "E".

Q - quit editing the current type of task data

When defaults are appropriate or an old value is to be left unchanged, type an asterisk (*) for the value.

(3) - There are four types of data that may be specified for each node, and they are edited separately. The user has chosen to begin with task time data.

(4) - The data to be entered for task data is node number, distribution type, mean, and standard deviation. The input line creates node 10 with distribution type 10, mean = 0.1 and standard deviation = 0.035. The values entered must be separated by one or more blanks or commas. They need not line up under the labeled columns. MOPADS echoes the command data on the next line. The option echoed is the user option preceded by "S" (for Show). If an error is made that is echoed, the "S" will be replaced by an "X" (e.g., "XE").

(5) - This line demonstrates that the spacing between input data elements is not significant.

(6) - When using the "S" option (e.g., S 10), only the node number is required.

As a demonstration, the standard deviation of node 10 is changed to 0.04 and then back to 0.035. Note that the asterisks caused the distribution type and the mean to be unchanged.

The option menu and column headings have been printed again. They are printed every 15 lines so they will always be visible on most scrolling screen terminals.

(7) - The "C" option is invalid for node 11 because it does not exist. The "E 11 8" line creates node 11 with distribution 8 and default mean and standard deviation.

MCT2: DELETING NODES

If the 'C' option sets the distribution type to zero, the node is deleted. This is the only way a node can be deleted. The "E" option can be used to re-create a node, but previous parameter values (e.g., mean) will be lost.

This capability is used to delete node 11.

(8) - The "Q" terminates editing of task time data and re-prints the data type menu.

(9) - Begin editing task specific human factors parameters. The 10 categories with their associated index values (1 through 10) are printed.

(10) - This "C" command sets parameter 5 to one and parameter 6 to 5. Then MOPADS echoes the parameter values for node 10. When a node is created, it automatically has default values for each task specific parameter. All 10 current values are printed each time the node is referenced. Note that in the echo, parameter 5 has value one and parameter 6 has value 5.

(11) - If an invalid option ("C16") is specified, MOPADS prints the option menu.

(12) - Begin editing skill data. One skill may be specified on each command line. No skills are specified for a task by default. All skill data must be specified explicitly with the CHANGE command.

(13) - Skills 13 and 17 were specified for node 10. The command "C 10 17 0" deletes skill 17 for node 10. This is reflected by the subsequent "S 10" command. Also, existing skills may have there weights changed by issuing a "C" command (e.g., "C 10 13 .8").

(14) - Incorrect skills were entered for node 10, and the above capability was used to remove the skills from node 10 and add them to node 14.

(15) - Examine skills for node 14 and be sure they are correct before moving on to node 16.

(16) - Begin to edit ADSM resource data.

(17) - No ADSM resource data was specified for this example, but for completeness resource data is entered for node 10 and then removed. This removal is accomplished by issuing a "C" command with resource parameters equal to zero.

(18) - Before exiting from the CHANGE command, a listing of task parameters may be obtained. It can be printed at the terminal or on the MOPADS file for non-interactive output. These listings should be scrutinized to ensure correct data entry.

3-2. CHANGE Operator Data.

Figure III-3 shows the CHANGE command to edit Operator State Vectors. The annotations on the figure are explained below.

(1) - Three types of data are contained in the operator state vectors that can be edited with CHANGE.

(2) - Begin editing human factors data. There are 65 human factors data elements. The user can display any of them. Here elements 20 to 25 are selected.

(3) - MOPADS prompts for elements to be edited one at a time. Both the label and value may be changed although normally there is little utility in changing the label since the moderator functions specify human factors data by element number. This however, does not preclude specialized procedures, written into the implementation of a particular system module which might make changing a label desirable.

(4) - Use the asterisk (*) to avoid changing existing values.

(5) - When done editing, type zero for the element number. This brings up the display option again.

(6) - New values for elements 24 and 25 are shown.

(7) - A listing may be obtained when done editing each data type.

(8) - Editing goal parameters is similar to editing human factors parameters except the labels may not be changed. Also, care must be taken in changing this data. See Polito (1983c) for a discussion of goal data.

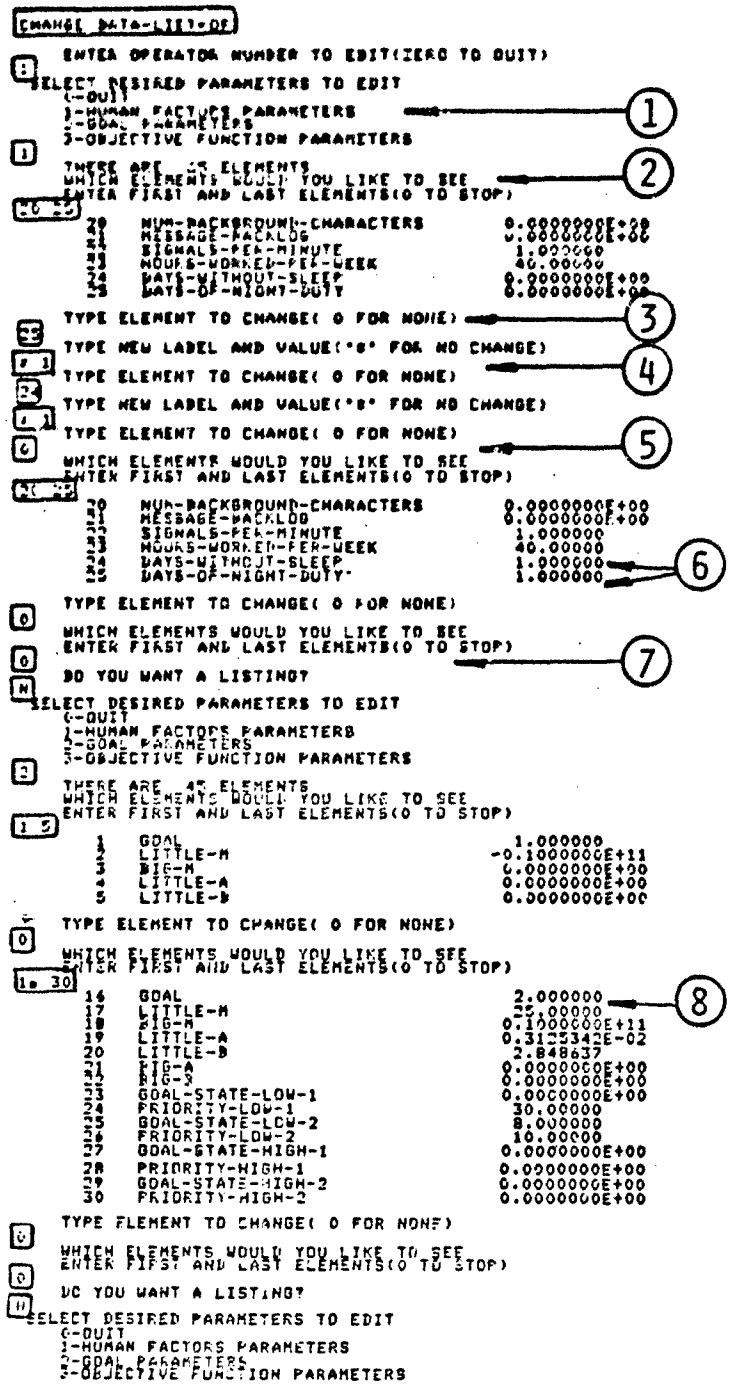


Figure III-3. CHANGE Operator Data Example.

```
3
THERE ARE    2 ELEMENTS
WHICH ELEMENTS WOULD YOU LIKE TO SEE
ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 2
   1   OBJECTIVE-FUNCTION                1.000000
   2   NUMBER-OF-GOALS                   2.000000
TYPE ELEMENT TO CHANGE( 0 FOR NONE)
1
TYPE NEW LABEL AND VALUE('0' FOR NO CHANGE)
0 3
TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
WHICH ELEMENTS WOULD YOU LIKE TO SEE
ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 2
   1   OBJECTIVE-FUNCTION                3.000000
   2   NUMBER-OF-GOALS                   2.000000
TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
WHICH ELEMENTS WOULD YOU LIKE TO SEE
ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
DO YOU WANT A LISTING?
Y
TERMINAL(T) OR LINE PRINTER(P)
T
```

(9) (10)

```
PAGE=    1
                    D A T A   L I S T   R E P O R T

    USER MESSAGE:      OPERATOR-1
    DATA BASE FILE:    VOL46.DBF
    DATA LIST LABEL:   OP-OPERATOR-STATE-VECTOR
    DATA LIST ID:      ( 1- 3)=          5   1516      1
    DATA LIST TYPE:    REAL

    DATA LIST CONTENTS:

    ROW:       1

 COLUMN                    VALUE          LABEL
 ------   ----------------------------   ----------------------------------------
      1                    1.000000       OPERATOR-TYPE
      2           0.0000000E+00           OPERATOR-ID
      3                   32.00000        POINTER-TO-HF-TASK-DATA
      4                   97.00000        POINTER-TO-GOAL-PARAMETER
      5                   142.0000        POINTER-TO-OBJ-FUNCTION
      6                   144.0000        POINTER-TO-DISPLAY-INFO
      7                   147.0000        POINTER-TO-TASK-STACK
      8           0.0000000E+00           UNUSED
      9           0.0000000E+00           UNUSED
     10           0.0000000E+00           UNUSED
     11           0.0000000E+00           UNUSED
     12           0.0000000E+00           UNUSED
     13           0.0000000E+00           UNUSED
     14           0.0000000E+00           UNUSED
     15           0.0000000E+00           UNUSED
     16           0.0000000E+00           UNUSED
     17           0.1000000E+21           START-TRACE-TIME
     18           0.1000000E+21           END-TRACE-TIME
     19           0.1000000E+21           TRACE-INCREMENT
     20           0.0000000E+00           UNUSED
     21           0.0000000E+00           UNUSED
     22           0.0000000E+00           UNUSED
     23           0.0000000E+00           UNUSED
     24           0.0000000E+00           UNUSED
     25           0.0000000E+00           UNUSED
     26           0.0000000E+00           UNUSED
     27           0.0000000E+00           UNUSED
     28           0.0000000E+00           UNUSED
     29           0.0000000E+00           UNUSED
     30           0.0000000E+00           UNUSED
     31           0.0000000E+00           UNUSED
     32                   37.00000        CORE-TEMPERATURE
     33                   1.000000        CID-VALUE
     34           0.0000000E+00           TIME-ON-TASK
     35           0.0000000E+00           DAYS-OF-DUTY
     36                   1.000000        SEARCH-DIMENSIONS
     37           0.0000000E+00           NUMBER-FIRE-UNITS
```

(11) (12)

Figure III-3. (continued)

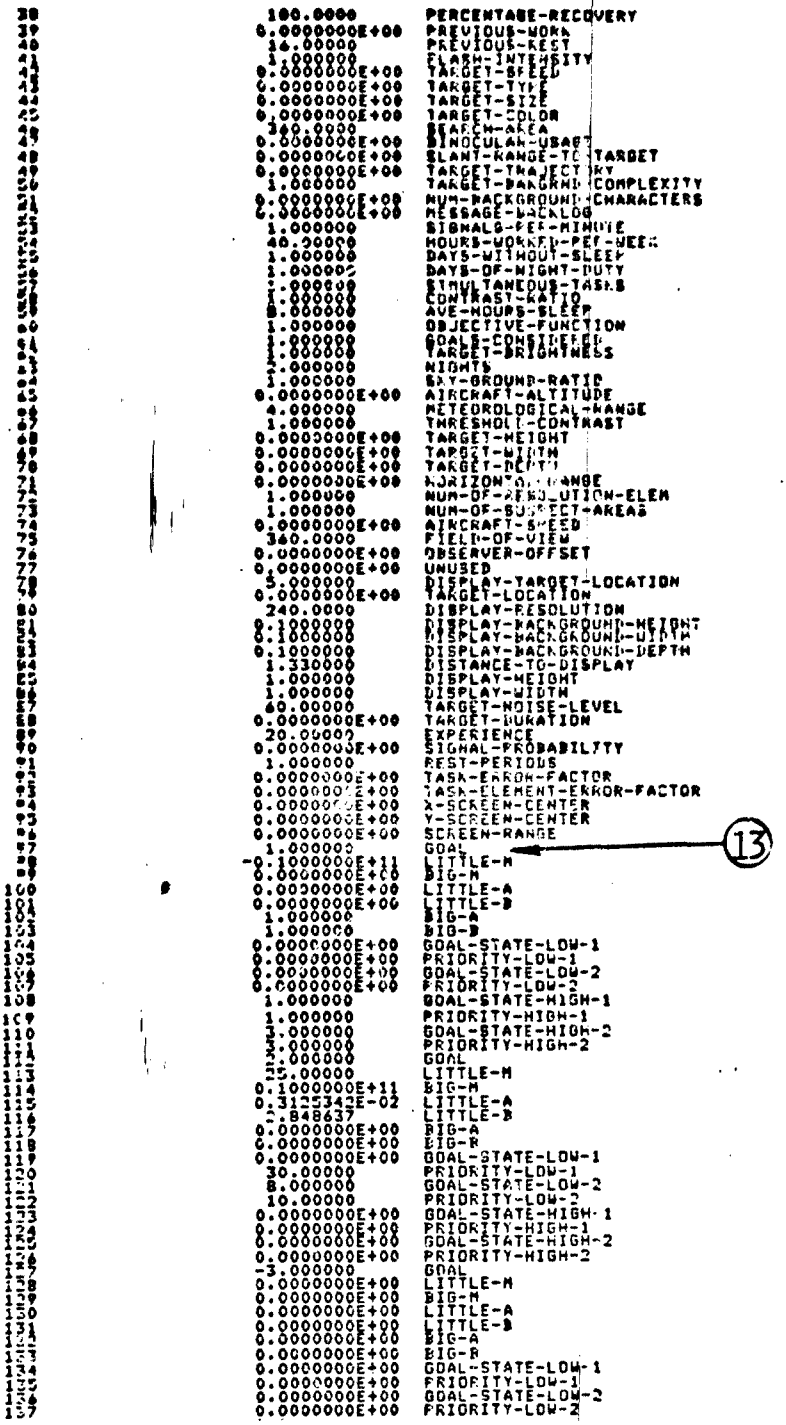


Figure III-3. (continued)

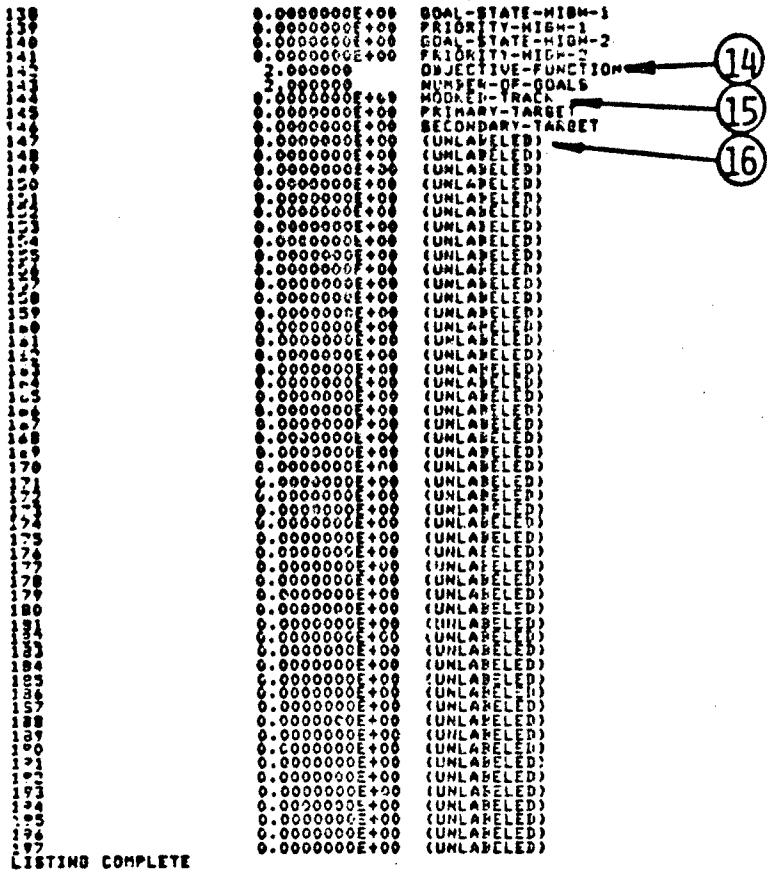


Figure III-3. (continued)

Each operator goal requires 15 goal elements (in the example there are 3 goals, hence 45 goal elements). In Figure III-3, elements 23 to 30 are the points on the goal state vs. goal priority curve input by the user with the ADD command. Elements 17 to 22 are computed from these points to fit exponential curves to the points, Polito (1983c). No automatic computation of elements 17 to 22 is performed by CHANGE. In order to change an operator's goal curves, these elements must be computed manually and then edited with CHANGE.

(9) - Two objective function parameters may be edited. See Polito (1983c) for a discussion of these parameters.

(10) - The operator state vectors are stored with pointers to the various types of data (i.e., human factors, goal parameters, etc.). See (11). Thus, the human factors data begins at element 32 (12).

To find human factor parameter 6, for example, compute its index as follows: 32+6-1 = 37. Column 37 of the operator state vector is human factors parameter 6.

Similarly, the goal parameters start at column 97, (13), and the objective function data begins at column 142, (14).

Finally, the display and task stack information begins at columns 144, (15), and 147, (16), respectively. These data cannot be edited because they change dynamically during simulations.

3-3. CHANGE Environmental Data.

Figure III-4 is an example of editing the Environmental State Vector for the reference ADSM.

(1) & (2) - The environmental state vector contains system and human factors data. Editing is performed in the same way as for human factor parameters in the operator state vectors.

(3) - The listing is also in the same format as the operator state vectors. System parameters begin at element 2, (4), and human factors data starts at element 23, (5).

```
CHANGE DATA-LIST-EN
 SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-SYSTEM PARAMETERS
    2-HUMAN FACTORS PARAMETERS                    <-- (1)
1
    THERE ARE  21 ELEMENTS
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 5
      1   SYSTEM-MODE                          1.000000
      2   OPERATOR-MODE                        0.0000000E+00
      3   DEFCON                               0.0000000E+00
      4   METHOD-OF-CONTROL                    1.000000
      5   WEAPONS-CONTOL-STATUS                2.000000
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
3
    TYPE NEW LABEL AND VALUE('B' FOR NO CHANGE)
0 1
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
    DO YOU WANT A LISTING?
N
 SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-SYSTEM PARAMETERS
    2-HUMAN FACTORS PARAMETERS
2
    THERE ARE  8 ELEMENTS
    WHICH ELEMENTS WOULD YOU LIKE TO SEE          <-- (2)
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
1 8
      1   DRY-BULB-TEMPERATURE                 22.00000
      2   RELATIVE-HUMIDITY                    50.00000
      3   AIR-MOVEMENT-RATE                    4.000000
      4   NOISE-LEVEL                          45.00000
      5   WORKING-AREA-ILLUMINATION            50.00000
      6   NUMBER-ON-DUTY                       1.000000
      7   VIBRATION                            0.0000000E+00
      8   AMBIENT-VAPOR-PRESSURE               10.00000
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
3
    TYPE NEW LABEL AND VALUE('B' FOR NO CHANGE)
0 4.4
    TYPE ELEMENT TO CHANGE( 0 FOR NONE)
0
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
    DO YOU WANT A LISTING?
N
 SELECT DESIRED PARAMETERS TO EDIT
    0-QUIT
    1-SYSTEM PARAMETERS
    2-HUMAN FACTORS PARAMETERS
1
    THERE ARE  21 ELEMENTS
    WHICH ELEMENTS WOULD YOU LIKE TO SEE
    ENTER FIRST AND LAST ELEMENTS(0 TO STOP)
0
    DO YOU WANT A LISTING?
Y
    TERMINAL(T) OR LINE PRINTER(P)
T
    ENTER MESSAGE TO PRINT WITH IT                <-- (3)
.. ..
```

Figure III-4. CHANGE Environmental Data Example.

PAGE= 1

DATA LIST REPORT

DATA BASE FILE: VOL46.DBF
DATA LIST LABEL: EN-ENVIRONMENT-STATE-VEC
DATA LIST ID: , 1- 3)= 5 514 1
DATA LIST TYPE: REAL

DATA LIST CONTENTS:

ROW: 1

COLUMN	VALUE	LABEL
1	23.00000	POINTER-TO-HF-DATA
2	1.000000	SYSTEM-MODE
3	1.000000	OPERATOR-MODE
4	0.0000000E+00	DEFCON
5	1.000000	METHOD-OF-CONTROL
6	3.000000	WEAPONS-CONTOL-STATUS
7	1.000000	WEAPONS-ALERT-DESIGNATION
8	0.0000000E+00	AMMUNITION-HOT
9	0.0000000E+00	AMMUNITION-COLD
10	0.0000000E+00	ALERT-STATUS
11	0.0000000E+00	CONFIGURATION
12	0.0000000E+00	FIRE-UNIT-STATUS
13	0.0000000E+00	PRIMARY-TRACK-ADDRESS
14	0.0000000E+00	PRIME-METHOD-OF-FIRE
15	0.0000000E+00	PRIME-TARGET-CATEGORY
16	0.0000000E+00	SNDRY-TRACK-ADDRESS
17	0.0000000E+00	SNDRY-METHOD-OF-FIRE
18	0.0000000E+00	SNDRY-TARGET-CATEGORY
19	0.0000000E+00	X-POSITION
20	0.0000000E+00	Y-POSITION
21	0.0000000E+00	Z-POSITION
22	1.000000	SAMPLING-OPTION
23	72.00000	DRY-BULB-TEMPERATURE
24	50.00000	RELATIVE-HUMIDITY
25	4.000000	AIR-MOVEMENT-RATE
26	45.00000	NOISE-LEVEL
27	45.00000	WORKING-AREA-ILLUMINATION
28	1.000000	NUMBER-ON-DUTY
29	0.0000000E+00	VIBRATION
30	10.00000	AMBIENT-VAPOR-PRESSURE

LISTING COMPLETE

(4)

(5)

Figure III-4. (continued)

3-4. CHANGE System Resource Data.

System resources were not defined for the example, but an example of defining such data is given in Figure III-5. The data which may be specified for a resource is:

Resource number

Number of Units Available

Mean Time Between Failures (MTBF) (Hrs)

Mean Time To Repair (MTR) (Hrs)

Use Code (0 - non-exclusive use, 1-may be used by only one operator)

Status (1 - green, 2 - amber, 3 - red)

Editing is performed in the same way as for Task data and the listing (not shown) is formatted in the same way.

4-0 THE USE AND RENAME COMMANDS.

Figure III-6 is an example of the USE and RENAME commands. The USE command may be issed at any time in the "CREATE/EDIT REFERENCE SYSTEM MODULE" subprocess to make a specified ADSM current. Similary, RENAME may be issued at any time to rename an ADSM. The renamed ADSM becomes current.

The annotations on Figure III-6 are explained below.

(1) - When entering MOPADS with an "OLD" data base, it is most often done to edit an existing reference ADSM. The CHANGE command requires that an ADSM be current. The USE command is issued to select a particular ADSM.

(2) - The CURRENT command confirms that "E-EXAMPLE" has become current.

(3) - The RENAME command renames "E-EXAMPLE." Note that the ACC ("E" in this case) cannot be changed.

(4) - The CURRENT command confirms the RENAME.

5-0 THE SAVE, GET, AND DELETE COMMANDS.

Figure III-7 shows an example with the SAVE, GET, and DELETE commands. The annctations are explained below.

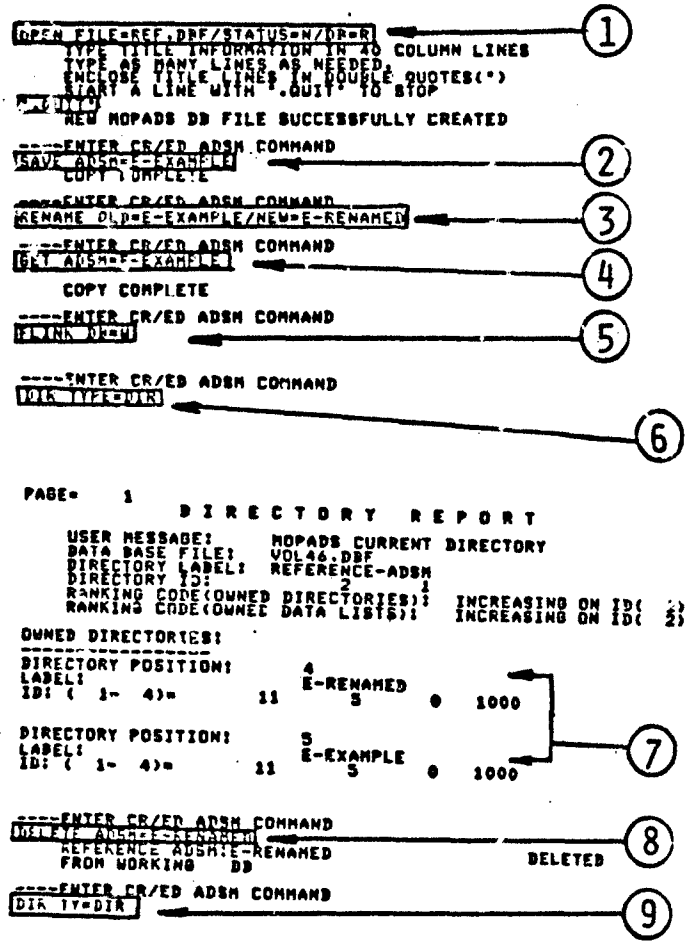
(1) - A reference data base is opened for the example.

```
CHANGE DATA-LIST=R
    WHEN ENTERING VALUES, USE '*' TO SELECT
    DEFAULT OR LEAVE UNCHANGED
    FOR OPTIONS, SELECT 'H' TO GET THIS MENU

    OPTION: S-SHOW PARAMETERS, M-MENU, E-ENTER DATA FOR NEW RESOURCE
            C-CHANGE EXISTING RESOURCE, Q-QUIT RESOURCES
 OPTION  RESOURCE NO.-UNITS      MTBF          MTR      COM   STATUS
E 1 1 100. .8 1 1
   LABEL?
RADIO
   SC          1            1    100.0        0.8000       1       1
  LABEL=RADIO
C 1 = 95
   LABEL?
B
   SC          1            1    95.00        0.8000       1       1
  LABEL=RADIO
C 1 0
   SC          1            0    0.0000E+00   0.0000E+00   0       0
  LABEL=(UNUSED)
S 1
    RESOURCE NOT DEFINED
Q
H   DO YOU WANT A LISTING

----ENTER LR/ED ADSM COMMAND
```

Figure III-5. CHANGE System Resource Data Example

```
    SELECT OPTION

     0-TERMINATE
     1-CREATE/EDIT SIMULATION DATA SET
     2-SET UP SIMULATION RUN DATA
     3-EXAMINE STATISTICS
     4-CREATE/EDIT SCENARIO DATA
     5-CREATE/EDIT REFERENCE SYSTEM MODULE
    TYPE OPTION NUMBER
3   CREATE/EDIT REFERENCE SYSTEM MODULE
----ENTER CR/ED ADSM COMMAND
USE ADSM=E-EXAMPLE                                        (1)
----ENTER CR/ED ADSM COMMAND
CUR TYPE=DIRECTORY

    CURRENT DIRECTORY  LABEL IS:E-EXAMPLE                 (2)
----ENTER CR/ED ADSM COMMAND
RENAME OLD=E-EXAMPLE/NEW=E-RENAMED                        (3)
----ENTER CR/ED ADSM COMMAND
CUR TY=DI

    CURRENT DIRECTORY  LABEL IS:E-RENAMED                 (4)
```

Figure III-6. USE and RENAME Example.




```
PAGE=    1
                D I R E C T O R Y   R E P O R T

    USER MESSAGE:      MOPADS CURRENT DIRECTORY
    DATA BASE FILE:    VOL44.DBF
    DIRECTORY LABEL:   REFERENCE-ADSM
    DIRECTORY ID:                2          1
    RANKING CODE(OWNED DIRECTORIES):   INCREASING ON ID(  2)
    RANKING CODE(OWNED DATA LISTS):    INCREASING ON ID(  2)

OWNED DIRECTORIES:
------------------
DIRECTORY POSITION:          4
LABEL:                       E-RENAMED
ID: (  1-  4)=          11       5          0    1000

DIRECTORY POSITION:          5
LABEL:                       E-EXAMPLE
ID: (  1-  4)=          11       5          0    1000
```



```
----ENTER CR/ED ADSM COMMAND
DELETE ADSM=E-RENAMED
    REFERENCE ADSM:E-RENAMED                    DELETED
    FROM WORKING    DB
----ENTER CR/ED ADSM COMMAND
DIR TY=DIR
```



```
PAGE=    1
                D I R E C T O R Y   R E P O R T

    USER MESSAGE:      MOPADS CURRENT DIRECTORY
    DATA BASE FILE:    VOL44.DBF
    DIRECTORY LABEL:   REFERENCE-ADSM
    DIRECTORY ID:                2          1
    RANKING CODE(OWNED DIRECTORIES):   INCREASING ON ID(  2)
    RANKING CODE(OWNED DATA LISTS):    INCREASING ON ID(  2)

OWNED DIRECTORIES:
------------------
DIRECTORY POSITION:          5
LABEL:                       E-EXAMPLE
ID: (  1-  4)=          11       5          0    1000
```

Figure III-7. SAVE, GET, and DELETE Example.

② - The ADSM "E-EXAMPLE" is saved to the reference DB. After the SAVE, the saved ADSM is current on both the working and reference DB.

③ - Rename "E-EXAMPLE" to "E-RENAMED" on the working DB. (Note that RENAME works only on the working DB.)

④ - Now GET a copy of "E-EXAMPLE" from the reference DB. "E-EXAMPLE" will be current on both the working and reference DB's after the copy.

⑤ - PLINK on the working DB to the "REFERENCE-ADSM" directory so it can be examined.

⑥ - Look at the "REFERENCE-ADSM" directory.

⑦ - Both "E-RENAMED" and "E-EXAMPLE" are present.

NOTE

They both have the same ID! This is not a desirable situation and points out that this is a contrived example. In practice, only one version of a particular ADSM would be used in the working DB.

⑧ - Delete "E-RENAMED." It is not recoverable after DELETE.

⑨ - A DIRECTORY command shows that "E-RENAMED" is gone.

GET, SAVE, and DELETE require substantial computer processing, and the user should be prepared for a delay before the next prompt is printed by the computer.

IV. REFERENCES

Goodin, II, J. R. & Polito, J. MOPADS free-format syntax processor (MOPADS/FFSP) (MOPADS Vol. 5.11). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX. 1983.

Laughery, K. R. and Gawron, V. Human factors moderator functions (MOPADS Vol. 5.10). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. The MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Polito, J. MOPADS data base control system (MOPADS/DBCS)(MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. MOPADS task sequencing structure (MOPADS Vol. 5.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c).

Walker, J. L. & Polito, J. Documentation requirements and development guidelines for MOPADS air defense system modules (MOPADS Vol. 4.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Walker, J. L. & Polito, J. Development methodology for MOPADS air defense system modules (MOPADS Vol. 4.4). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

This page intentionally left blank.

V. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
4327 Spruce St., Room 108
Boulder, CO 80301

This page intentionally left blank.

VI. CHANGE NOTICES

This page intentionally left blank.

APPENDIX A. INTRODUCTION TO THE MOPADS USER INTERFACE

1-0 CONVERSING WITH THE MOPADS USER INTERFACE.

1-1. Organization.

The MOPADS User Interface is hierarchical in structure. It consists of five subprocesses as shown in Figure A-1. Each of the subprocesses has its own set of capabilities and commands to invoke them. The subprocesses are described in separate documents. In addition, a set of Basic Data Base Commands are provided that are available in all subprocesses. The main purpose of this section is to explain the use of these commands.

1-2. Syntax Rules.

The User Interface is primarily a command driven processor that waits for the user to issue instructions. It does, however, have aspects of menu driven systems in that some commands result in menus being presented to the user. Also, the command processor (FFSP described in Goodin and Polito (1983) permits menu-like presentations of commands.

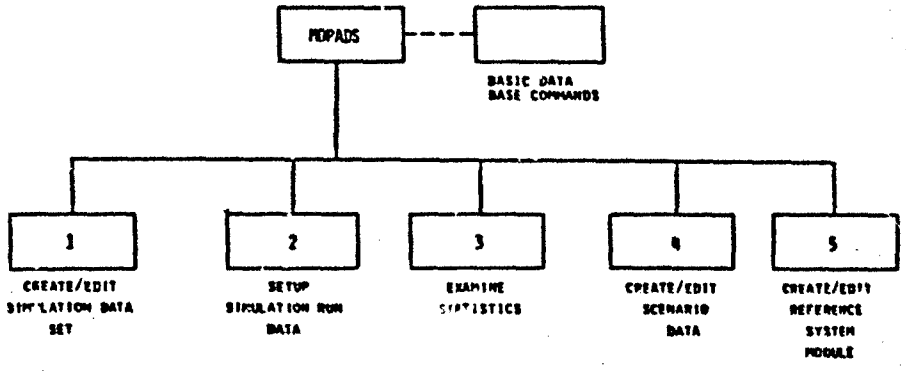


Figure A-1. Organization of the User Interface.

The regular mode for entering commands is shown below.

```
command,prompt1=response1/prompt2=response2/...
```

The commands and prompts are keywords recognized by MOPADS. The responses are particular values for the prompts. For example, consider this.

```
OPEN,FILE=MOPADS.DBF/STATUS=OLD
```

OPEN is the command. FILE and STATUS are prompts recognized by MOPADS, and MOPADS.DBF and OLD are values for the prompts.

Certain prompts for a command may have default values that will be used if the prompt is not entered. In the example above, another prompt, DB, specifies which data base is to be associated with the file. Its default is WORKING, so by not entering it on the command line, WORKING is automatically selected. If the default value is not desired, then the prompt must be explicitly entered on the command line.

If the user fails to enter responses to all required prompts, MOPADS will interactively prompt for them. For example,

```
OPEN,STATUS=OLD

FILE[NO DEFAULT] = MOPADS.DBF
```

After processing the OPEN command, MOPADS found that the required prompt, FILE, was not entered. It printed "FILE[NO DEFAULT]=" to prompt the user for a response. If the last non-blank character on a command line is a dash (-), MOPADS will interactively prompt for all unentered prompts, even those with defaults. For example,

```
OPEN,STATUS=OLD -

DB[WORKING] = REFERENCE

FILE[NO DEFAULT] = MOPADS.DBF
```

The dash caused "DB[WORKING]=" to be printed. The value between the brackets is the default for the prompt. The default can be selected by typing "DEF" as the response. DEF can also be entered on the command line; e.g.

```
OPEN,DB=DEF/STATUS=OLD/FILE=MOPADS.DBF
```

The above demonstrates that the prompt-response groups can be entered in any order.

Also, a command can be cancelled at any time by typing "CANC" as a response or a prompt. For example,

```
OPEN,CANC

OPEN,FILE=CANC
```

Note that DEF and CANC are essentially reserved words. The user interface treats commas, blanks, and equal signs as interchangeable separators. Also, multiple separators are treated as a single separator. This means that the commas in the previous examples could be replaced by any combination of one or more blanks and commas. The same is true of the equal signs, but their use is recommended to make the command lines easy to read. The slashes are required separators between prompt-response groups, but they can be preceded or followed by blanks or commas.

A response may include separators (i.e., commas, blanks, equal signs, and slashes) if it is enclosed in quote marks ("). For example, on some computers file names contain embedded blanks, e.g.,

```
OPEN  FILE="MOPADS DBF"
```

Without the quote marks above, MOPADS will consider MOPADS DBF as two responses when only one is desired. (NOTE: A single prompt may have more than one response if the programmer specified it that way. In such a case, each response would be separated by a blank or comma. In the case above, however, where a single response is required, the quote marks must be used to embed the blank in the response.)

Any response may be enclosed in quotes, although there is no advantage in doing so unless a separator is to be embedded. Blank responses can be entered with " ." where at least one blank appears between the quotes.

A generalization of entering only some of the prompts is to enter only the command name:

```
OPEN
DB[WORKING]=DEF
FILE[NO DEFAULT]=MOPADS.DBF
STATUS[OLD]=DEF
```

The User Interface will prompt for all responses. This method can be selected if the user does not remember the prompts.

For commands which the user issues frequently, a concise mode can be selected by preceding the command with "C-". In this case, the prompt= part of the syntax may be omitted. For example,

C-OPEN DEF/MOPADS.DBF

Responses must be entered in the same order as they are prompted in the command-name-only form. No response may be skipped, except that if all remaining responses have defaults and the defaults are desired, then the command line may be terminated (e.g., the STATUS response was omitted above since OLD was desired). The dash works in the concise mode in the same way as in other modes.

The following rules will formalize the previous discussion of how syntax is processed by FFSP.

- The command-name-only form of a command may be used at any time by typing only the command name.

Blank responses and responses containing separators may be entered by enclosing them in quotes. To enter a blank response type " " (including the quotes). At least one blank must be entered between the quote marks.

A command may be cancelled at any time by typing CANC for any prompt or response. You can not abbreviate CANC.

The user may elect to use the default value(s) by typing DEF for any response in a response list up to one field past the last response in the list.

Slashes (/) must be used to separate one prompt-response group from another. Blanks or commas may be used to separate all other fields. The equal sign should be used to separate prompts from their responses; however, it is not required.

Command and prompt names may be abbreviated to any non-ambiguous string of characters. For example, if there are two commands, DESIGN and DESCRIBE, they can be abbreviated DESI and DESC respectively. The commands may be abbreviated in longer forms. For example, the user may enter DESC, DESCR, DESCRI, DESCRIB, or DESCRIBE for the command DESCRIBE.

If a command line in regular or concise mode is ended with more than one dash, the last dash will signify to the system to prompt the user for all the unentered responses. Other dashes will then be considered as part of a response.

Any multiple combination of commas and blanks is treated as a single separator. For example,

NAME = BILL WOLF and NAME = BILL , WOLF

are equivalent (here the response is a list of two character strings).

If the user enters an incorrect response or misuses the syntax, FFSP will explain the error and prompt interactively for all remaining responses.

Concise mode is signified by preceding the command name with "C-" (without the quotes).

1-3. The Current Directory and Current Data List.

MOPADS data bases are made up of multiple directories, each of which may own one or more directories and data lists. Normally, the User Interface operates on one directory which is designated "current." Similarly, one data list of the current directory can be designated the current data list. This ensures that the action of User Interface commands are unambiguous. Facilities are provided for determining the current DR and DL and for selecting a DR and DL to become current. Note that some commands automatically change the current DR and DL.

2-0 BASIC DATA BASE COMMANDS.

2-1. CLOSE.

The CLOSE command (Figure A-2) will close either the working or reference data base. It can be used to switch to a new data base fil

2-2. CURRENT.

The CURRENT command (Figure A-3) will display label, ID or both of the current directory and/or data list on either data base.

2-3. DEPOSIT.

DEPOSIT (Figure A-4) is a low level editing command that allows any element of the current data list to be changed. DEPOSIT interactively requests element numbers and new values.

2-4. DIRECTORY.

DIRECTORY (Figure A-5) shows the contents (all owned directories and/or data lists) of the current directory on either data base. It shows the labels, ID's, and directory positions of the contents. This information is useful for the SELECT command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: CLOSE — PURPOSE: To close a Data Base file.

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
STATUS	3001	4	KEEP	KEEP DELETE		

Figure A-2. CLOSE Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: CURRENT

PURPOSE: To display the identity of the current Directory or Data List

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
DISPLAY	3001	4	LABEL	LABEL ID ALL		
TYPE	3001	4	DIRECTORY	DIRECTORY DATA-LIST ALL		

Figure A-3. CURRENT Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: DEPOSIT

PURPOSE: To set values of the elements of the current data list

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB interactively prompts for elements and values	3001	4	WORKING	WORKING REFERENCE		

Figure A-4. DEPOSIT Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: DIRECTORY

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: To list selected contents of the current directory on the working or reference DB

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
TYPE	3001	4	DIRECTORIES	DIRECTORY DATA-LIST ALL		
DB	3001	4	WORKING	WORKING REFERENCE		

Figure A-5. DIRECTORY Command Specifications.

2-5. EXAMINE.

EXAMINE (Figure A-6) will display selected contents of the current data list to the terminal or to the MOPADS line printer output file. If the latter is selected, the data list label and other information will also be printed.

2-6. HELP.

HELP (Figure A-7) will print the prompts and options for the prompts for the specified command.

2-7. MENU.

MENU has no prompts. It will print all commands available in the current subprocess.

2-8. OPEN.

OPEN (Figure A-8) will open a data base file as either the working or reference DB. OPEN will not automatically close the current DB. CLOSE must be used explicitly before OPEN to switch DB files. MAXRECORD is the maximum number of records allowed in a data base file. It may not be needed for your computer. Zero implies no limit on the file size. The (MASTER-DIRECTORY) is current after the OPEN command.

2-9. PLINK.

PLINK (Figure A-9) will change the current directory to the owner of the directory which was current when PLINK was issued.

2-10. QUIT.

QUIT has no prompts. It causes the current subprocess to be exited.

2-11. SELECT.

SELECT (Figure A-10) changes the current directory or data list to one that is owned by the directory that is current when SELECT is issued. The desired DR or DL is selected by specifying one (and only one) of the following: 1 - its directory position, 2 - its label, or 3 - its ID. This information is obtained with the DIRECTORY command.

2-12. TERMINATE.

TERMINATE has no prompts. It will close all open data bases and terminate execution. This is the normal way to end a User Interface session.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: EXAMINE

PURPOSE: To display elements of the current data list

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
HEADER	3001	4	NO	NO YES		
DEVICE	3001	4	TERMINAL	TERMINAL PRINTER		

Figure A-6. EXAMINE Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: HELP

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: HELP prints information on the prompts for the command specified in the response

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
COMMAND	3001	5	-	-	-	-

Figure A-7. HELP Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: OPEN — PURPOSE: Open a data base

MODULE: MOPADS/UI

REVISION DATE: 23 NOV 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
FILE	3001	5	-	-		
STATUS	3001	4	OLD	OLD NEW		
DB	3001	4	WORKING	WORKING REFERENCE		
MAXRECORD	1001	3	0	-	0	-

Figure A-8. OPEN Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: PLINK

MODULE: HOPADS/UI

REVISION DATE: 28 FEB 1983

PURPOSE: To make the owner directory of the current directory the new current directory

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001		WORKING	WORKING REFERENCE		

Figure A-9. PLINK Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: SELECT

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: To set the current directory on data list from those in the current directory

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
TYPE	3001	4	DIRECTORY	DIRECTORY DATA-LIST		
POSITION	1001	3	0	-	0	-
LABEL	3001	5	' '	-	-	-
ID	1000	5	0	-	-	-
(enter only one of the above three)						

Figure A-10. SELECT Command Specifications.

3-0 USER INSTRUCTIONS AND EXAMPLES.

Figure A-11 is an example using all of the Basic Data Base Commands. The annotations are explained below.

(1) - The CLOSE commands disassociates the current data base file from the MOPADS software. After close, MOPADS will know no working DB. If STATUS is specified as DELETE, the data base file will be destroyed permanently.

(2) - OPEN associates the data base file VOL46.DBF to the software as the working DB. STATUS=OLD implies that the DBF was previously created as a MOPADS data base file. If STATUS=NEW is specified, MOPADS will create a new MOPADS DB on VOL46.DBF, and any information previously contained on VOL46.DBF will be lost.

(3) - The MENU command is always available to list the commands currently available.

(4) - HELP prints information about a particular command.

(5) - The user has asked for information on the prompts DISPLAY and TYPE. For DISPLAY, MOPADS shows that it expects a response of one character string that must be one of the enumerated values LABEL (display the label), ID (display the ID), or ALL (display both label and ID). The default is LABEL. HELP is terminated by entering "*."

(6) - When OPEN is issued, the (MASTER-DIRECTORY) becomes current. The CURRENT command will not display a directory which has no owner, which of course, the MASTER-DIRECTORY does not.

(7) - The DIRECTORY command will always work, however, even on the (MASTER-DIRECTORY). The contents of the (MASTER-DIRECTORY) are shown.

(8) - The SELECT command makes the REFERENCE-ADSM directory current by specifying its directory position in the (MASTER-DIRECTORY).

(9) - The DIRECTORY command confirms that REFERENCE-ADSM is current and shows the owned reference ADSM's (in this case only one, E-EXAMPLE).

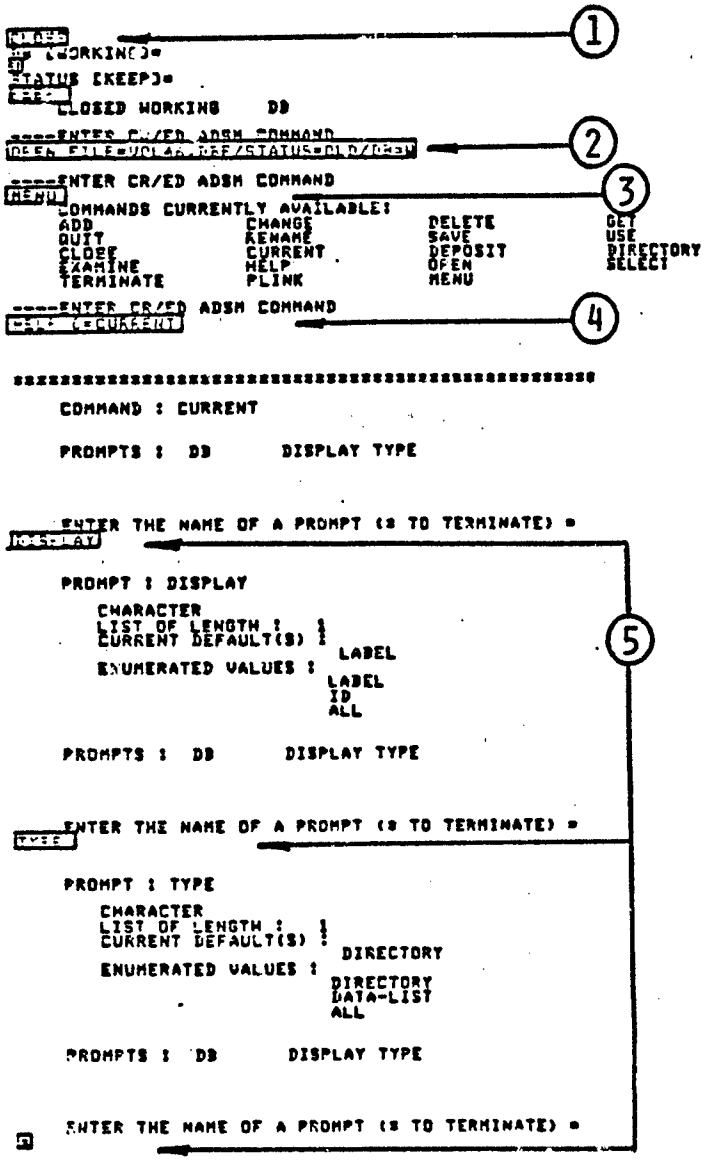


Figure A-11. Example Using Basic Commands.

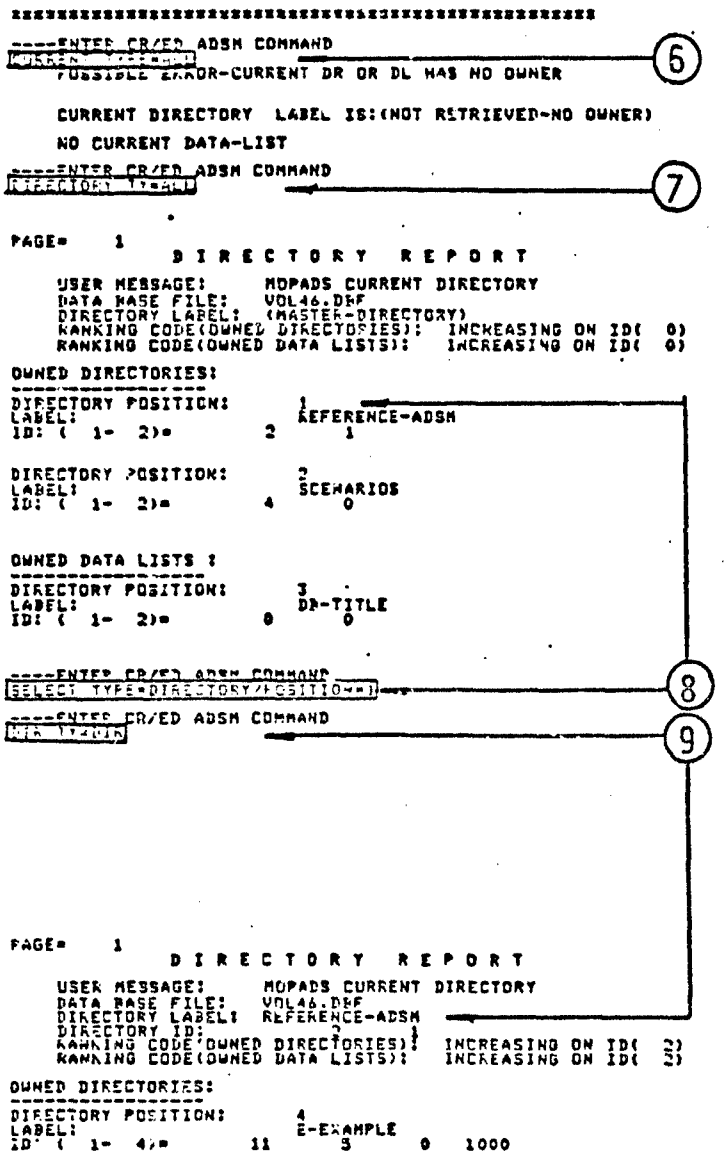


Figure A-11. (continued)

```
----ENTER CR/ED ADSM COMMAND
CRLF TYPE=DIR/FOR=A                                        (10)
----ENTER CR/ED ADSM COMMAND
DIR TYPE=D-RPT                                             (11)
```

```
PAGE=  1
                    D I R E C T O R Y   R E P O R T

    USER MESSAGE:     MOPADS CURRENT DIRECTORY
    DATA BASE FILE:   VOL46.BBF
    DIRECTORY LABEL:  E-EXAMPLE
    DIRECTORY ID:          11          S        0     1000
    RANKING CODE(OWNED DIRECTORIES):(   INCREASING ON ID(  0)
    RANKING CODE(OWNED DATA LISTS):(   INCREASING ON ID(  3)

OWNED DATA LISTS :
-----------------
DIRECTORY POSITION:      1
LABEL:                   R-ADS-RESOURCES
ID: (  1-  3)=       S     10        1

DIRECTORY POSITION:      2
LABEL:                   EN-ENVIRONMENT-STATE-VEC
ID: (  1-  3)=       S     514       1

DIRECTORY POSITION:      3                                 (12)
LABEL:                   OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=       S     1516      1

DIRECTORY POSITION:      4
LABEL:                   OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=       S     1516      2

DIRECTORY POSITION:      5
LABEL:                   OT-OPERATOR-TYPE
ID: (  1-  3)=       S     1520      1

DIRECTORY POSITION:      6
LABEL:                   RL-RESOURCE-LABELS
ID: (  1-  3)=       S     1812      1

DIRECTORY POSITION:      7
LABEL:                   TD-TASK-DATA
ID: (  1-  3)=       S     2004      1

----ENTER CR/ED ADSM COMMAND
```

Figure A-11. (continued).

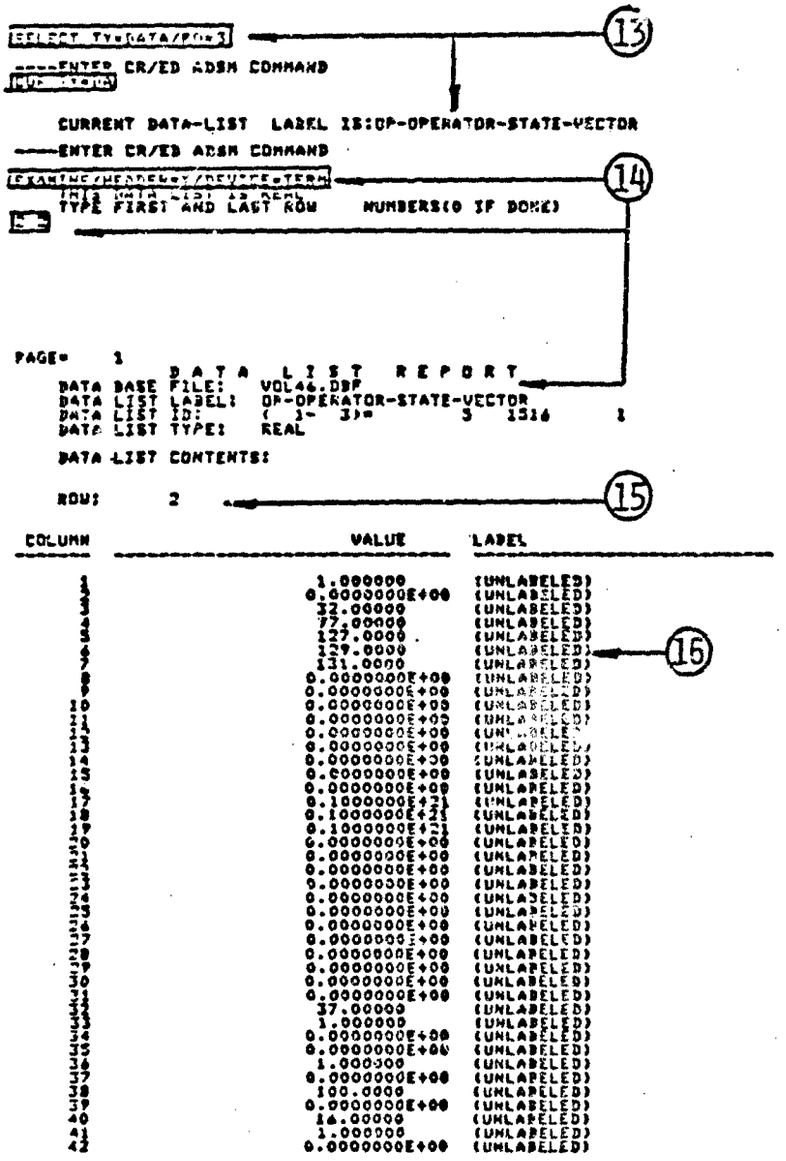


Figure A-11. (continued)

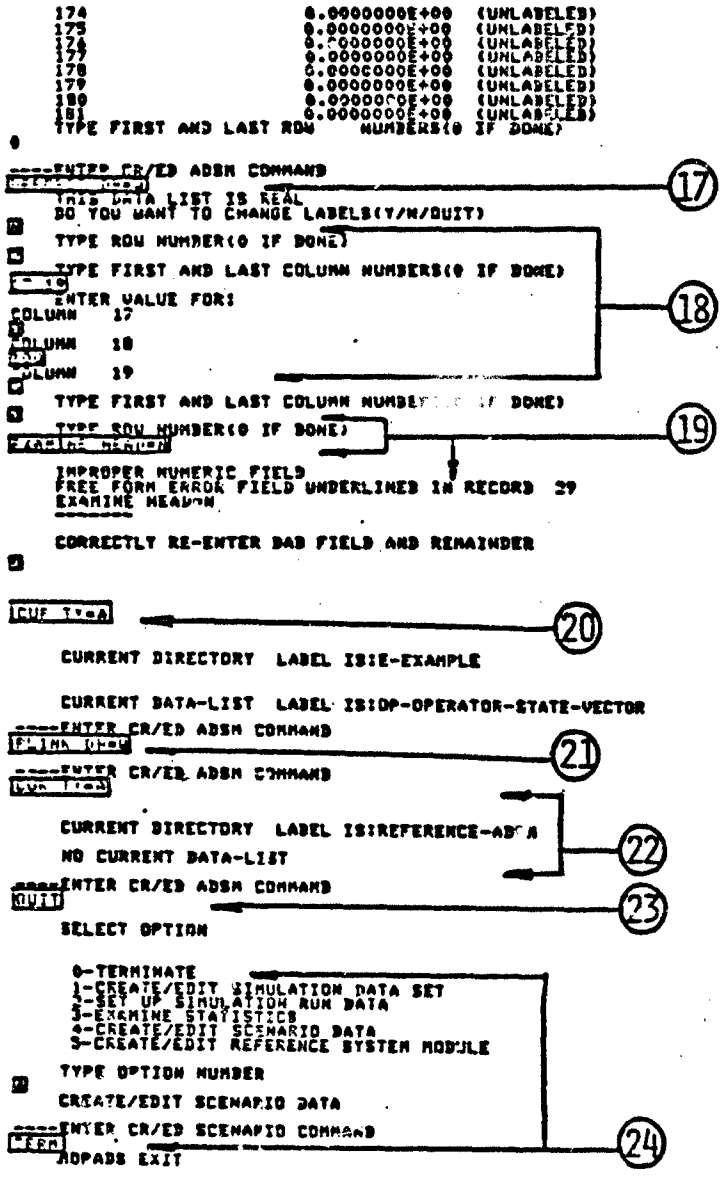


Figure A-11. (continued)

(10) - Select E-EXAMPLE to be current.

(11) - Look at the data list directory of E-EXAMPLE.

(12) - This operator state vector will be edited.

(13) - The SELECT command makes the above operator state vector the current data list. This is confirmed with the following CURRENT command.

(14) - EXAMINE specifies that the data list is to be displayed at the terminal. The user wants row 2. HEADER=Y causes the "DATA LIST REPORT" and the next 6 lines to be printed.

(15) & (16) - This display demonstrates that the basic data base commands are low level commands; they know nothing about the structure or contents of MOPADS data bases. The operator state vectors have two rows. The second row contains the reference values for each column. Prior to a simulation, the second row is copied to the first row, which is dynamically accessed during the simulation. In this way, the reference or starting values are not lost. MOPADS stores column labels only for the first row, however, since it would double the storage requirement to duplicate the labels for each row. The "CREATE/EDIT REFERENCE SYSTEM MODULE" subprocess knows all of this, so when a listing is obtained with the CHANGE command in that subprocess, the labels appear with the reference values.

The EXAMINE command, however, simply operates on a data list without knowledge of the meaning of its contents. Here, all of the elements of row 2 appear to be (UNLABELED). Also, EXAMINE prints the entire row. Only a portion of the listing is shown here for brevity.

(17) - Use the DEPOSIT command to change elements. It too is a low level command, so (row, column) addressing must be used.

(18) - Elements 17, 18, and 19 of row 2 are changed (these are operator trace parameters).

(19) - This demonstrates MOPADS response to input errors. The DEPOSIT command had not yet terminated when the user entered the next EXAMINE. MOPADS is expecting a numeric value for a row number. The free-format input processor examines every input string for

validity before processing it. Since "EXAMINE" is not a valid number, it prints the error message and requests correct input. Without this feature, the program would terminate abnormally from simple typing errors and perhaps damage the data base. MOPADS will protect the user from most such errors.

(20) - CURRENT shows that E-EXAMPLE is the current directory.

(21) - PLINK makes the current directory equal to the owner of the directory that was current when PLINK was issued.

(22) - The CURRENT command confirms that the REFERENCE-ADSM directory is now current. Note that when the current directory is changed, the current data list becomes undefined.

(23) - QUIT exits the current subprocess and brings up the subprocess option menu again.

(24) - The TERM command terminates execution. Selecting the zero option on the subprocess selection menu accomplishes the same thing.

APPENDIX J

MOPADS FINAL REPORT:

CREATING MOPADS AIR SCENARIOS

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

LIST OF FIGURES

LIST OF TABLES

TERMINOLOGY

1-0 STANDARD MOPADS TERMINOLOGY.

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

2-0 OTHER TERMINOLOGY.

ACC	ADSM Code Character.
ADSM	Air Defense System Module.
DATA BASE	The collection of user information and control information that is processed by the DBCS.
DATA BASE FILE	A computer file stored on disk that contains the data base.
DATA LIST	A list of values (character or numeric) that is user specified and is stored in a data base.
DB	Data Base.
DBCS	Data Base Control System.
DBF	Data Base File.

DIRECTORY	A collection of data lists and other directories.
DL	Data List.
DR	Directory.
ID	Identify. DL's and DR's have identifiers that are lists of integers and are stored in their owning directory.

1. INTRODUCTION

1-0 PURPOSE.

This document describes the facilities available in the MOPADS User Interface for creating and editing Critical Asset Configurations and Air Scenarios. These aspects of MOPADS models describe the assets to be protected by the air defense system and the flight paths of aircraft that interact with it.

The User Interface provides a mechanism to create and modify descriptions of critical assets and air scenarios. Asset and track descriptions are prepared on external data files. The user interface has commands which will read the data files and create the appropriate entries in the MOPADS data base. These entries can then be edited if needed.

2-0 INFORMATION NEEDED.

A schematic of the section of the MOPADS data base that concerns air scenarios is shown in Figure I-1. (A More detailed description of the MOPADS data base is contained in Polito, Vol. 5.17.) There is one directory labeled "SCENARIOS" (code 4) that belongs to the master directory. It owns all scenario information. Scenarios are defined which consist of critical asset configurations and one or more air scenarios that interact with it. Thus, the "CRITICAL-ASSET-CONFIGURATION" directories (code 13) own a data list that contains the coordinates of critical assets ("COORDINATE-AND-ASSET-DATA"). More than one "CRITICAL-ASSET-CONFIGURATION" directory may be present. They all have the same label, and are distinguished by user assigned numbers (e.g., 1, 2, 3, etc.).

Each "CRITICAL-ASSET-CONFIGURATION" may contain one or more air scenario directories (code 8). Each air scenario directory will have a unique, user assigned label (signified by lower case letters in the code 8 boxes in Figure I-1). The air scenarios may contain three data lists for hostile, friendly, and other tracks.

When performing a MOPADS simulation, the MOPADS user will select the CRITICAL-ASSET-CONFIGURATION and the air scenario to simulate. The MOPADS software will then access the data from the appropriate data base entries.

The contents of the "COORDINATE-AND-ASSET-DATA" data list are shown in Table I-1. In order to create a CRITICAL-ASSET-CONFIGURATION, the MOPADS analyst must determine the location of each asset. The coordinates are given in relation to a reference position that is

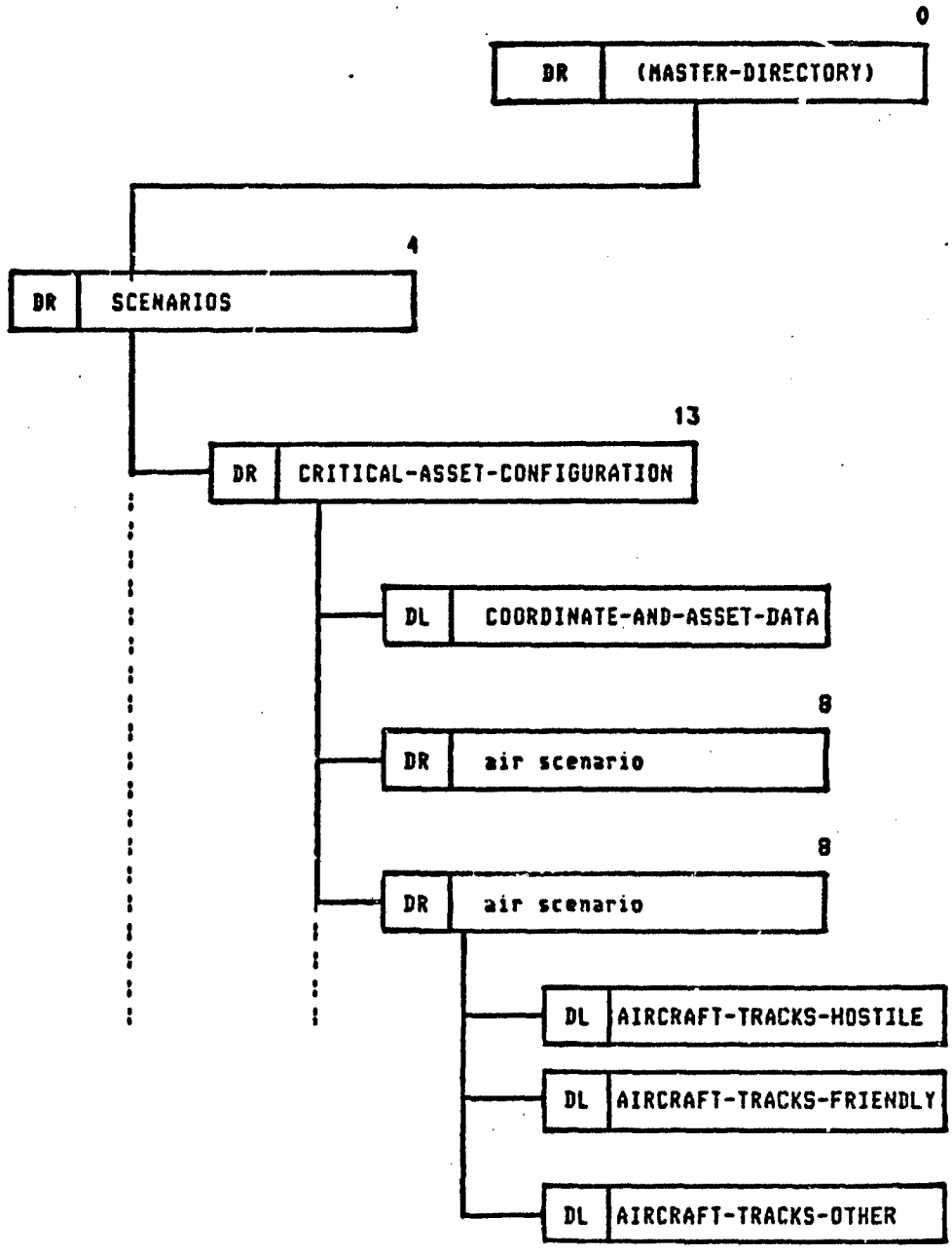


Figure I-1. Scenarios Portion of the MOPADS Data Base.

Table I-1. Contents of the COORDINATE-AND-ASSET-DATA Data List.

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	
Data List Label (25 char.max): COORDINATE-AND-ASSET-DATA		Page 1 of 1
Owner Directory Code: 13/CRITICAL-ASSET-CONFIGURATION DL Id:_ NECC('CD')/1		Rev.Date: 6 MAY 83
DL Type:____IDL/R C DIM =_____ _X_RDL/R C DIM = 1 ___CDL/max char=____		Author: Polito
ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1	REFERENCE-LATITUDE	decimal degrees (negative for south latitude)
2	REFERENCE-LONGITUDE	decimal degrees (negative for east longitude)
3	REFERENCE-ALTITUDE	feet above MSL
4	NUMBER-OF-CRITICAL-ASSETS	The number of critical assets data sets which follow.
5	X-COORDINATE	(n.mi.) for first critical asset (+ is east)
6	Y-COORDINATE	(n.mi.) for first critical asset (+ is north)
7	Z-COORDINATE	(ft.) for first critical asset (+ is up)
8	SITE-TYPE	A code value (currently always zero)
9	asset label (5 char.max)	Zero
10	(Not used)	Zero
		repeat (5) - (10) for each critical asset

specified in decimal degrees and feet above mean sea level. All other positions are specified in nautical miles (for x and y coordinates) and feet (for the z coordinate) from this point. (MOPADS assumes a flat earth.) Each site may be assigned a code (SITE-TYPE) and a five character label. Critical assets are not air defense units that are modeled in the MOPADS system. Air defense units are automatically protected. The sites specified in a CRITICAL-ASSET-CONFIGURATION are non-air defense sites designated to be protected.

The contents of the "AIRCRAFT-TRACKS-?" data lists are shown in Table I-2. Three such data lists exist: one each for hostile, friendly, and other tracks. All three of the data lists need not exist. These data lists have nine columns and as many rows as needed. One track consists of two or more rows.

Aircraft tracks are represented as connected line segments. The first row of a track is its initiation row. It specifies the initiation point (relative to the coordinate reference point) and the initiation time (in minutes from the start time of the simulation). Also, the initiation row has an identifier, the actual primary ID, the aircraft type, and multiplicity. Immediately following the initiation row are one or more segment rows which give the end point of the segment and the speed on the segment. For hostile tracks, an indicator gives whether the end point of the segment is a target (protected site) and whether the aircraft is jamming on the segment. The segment rows are in their correct geographical order. Each track is represented by a group of one initiation row and one or more segment rows, and the groups of track rows are ordered by track initiation time. In other words, the tracks which initiate earliest are in the lower numbered row.

Developing a scenario is the first and most fundamental task of building a MOPADS simulation, since everything including the air defense system will be located with reference to the coordinate reference point. The MOPADS analyst must assemble all of the information in Tables I-1 and I-2 and prepare data files described in the next section before initiating a User Interface session.

NOTE

Aircraft flight paths may not terminate in radar coverages. The MOPADS software will generate an error. Friendly aircraft may be represented as landing within coverage by creating a final flight segment with zero velocity.

Table I-2. Contents of AIRCRAFT-TRACKS Data Lists

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	S-17/DLD - 13
Data List Label (25 char.max): AIRCRAFT-TRACKS-(HOSTILE/FRIENDLY/OTHER)		Page 1 of 2
Owner Directory Code: 0 (user determined)	DL Ids: 1/2/3	Rev.Date: 10 DEC 82
DL Type: IDL/R C DIM = X RDL/R C DIM = ?	CDL/max char=	Authors: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C1		0-implies initiation row of a new track, 1-implies segment row of a track
	If C1 = 0	
C2		Identifying number for the aircraft
C3		Initiation time of the track from the simulation start time (minutes)
C4		Actual primary ID (1-hostile,2-friendly,3-other)
C5		Multiplicity
C6		Aircraft type (see attached)
C7		x - n.miles } initial point relative to the coordinate
C8		y - n.miles } initial point relative to the coordinate
C9		z - ft. } reference point
	If C1 = 1	
C2		x - n.miles } end point of flight segment relative to coordinate reference point
C3		y - n.miles } end point of flight segment relative to coordinate reference point
C4		z - ft. } end point of flight segment relative to coordinate reference point
C5		speed (knots) on this segment
C6		is the end point a target (0-no, 1-yes)(used only if hostile)
C7		is the aircraft jamming on this segment (0-no,1-yes)
C8		Probability of destroying target if C6=1
C9		0 not used

NOTES: 1. All rows for a track are in chronological and geographical order. The first row is the "initiation row" (column 1=0). Each subsequent row is a "segment row" (column 1=1).
2. Tracks must be ordered in chronological order of their initiation times.

This page intentionally left blank.

II. COMMANDS AVAILABLE

1-0 BASIC DATA BASE COMMANDS.

The basic data base commands that are available in all subprocesses of the User Interface are also available in this subprocess. Descriptions of these commands are repeated in Appendix A for easy reference.

2-0 THE ADD-AIR COMMAND.

The ADD-AIR command creates or modifies an air scenario directory of the current "CRITICAL-ASSET-CONFIGURATION" on the working data base. The prompts are shown in Table II-1. LABEL is a label for the air scenario. FILE is an external formatted, sequential data file which contains track data. Its format is shown in Table II-2.

As can be seen from Tables II-2 and II-3 the Track Data File may contain tracks of all three types. Several possibilities exist here:

1. The specified air scenario did not previously exist.

MOPADS will read FILE and create data lists for whichever track types exist on the file.

2. The specified air scenario previously existed.

MOPADS will ask if information in the air scenario may be overwritten.

2a. The answer is YES.

MOPADS will delete an "AIRCRAFT-TRACK" data list and replace it if like kind tracks are found on FILE. For example, an "AIRCRAFT-TRACKS-HOSTILE" data list exists and hostile tracks are found on FILE, the AIRCRAFT-TRACKS data list will be deleted and completely replaced by the hostile tracks on FILE. An existing "AIRCRAFT-TRACKS" data list will not be deleted unless replacements are found in FILE.

2b. The answer is NO.

MOPADS will skip tracks on FILE for which an "AIRCRAFT-TRACKS" data list already exists. For example, if AIRCRAFT-TRACKS-HOSTILE exists and hostile tracks are found on FILE, they will be skipped, and the existing data list will not be disturbed. Other track types on FILE for which no "AIRCRAFT-TRACKS" data list exists will be added to the directory in the normal manner.

Table II-1. The ADD-AIR Command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: ADD-AIR

MODULE: MOPADS/UI 41

REVISION DATE: 9 MAY 1983

PURPOSE: To create an Air Scenario in the current CRITICAL-ASSET-CONFIGURATION Directory on the working directory

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
FILE	3001	5	-	-	-	-
(Data file with track data. It is rewound prior to reading.)						
LABEL	3001	5	-	-	-	-

If the specified LABEL currently exists, the contents of FILE will replace an existing track data list of the same kind.

Table II-2. Track Data File Format.

COLUMNS	DESCRIPTION	FORMAT
Primary ID Record		
1-3	-1	I3
6-15	Primary ID (1-hostile,2-friendly,3-other)	G10.3
Track Initiation Record		
1-3	0	I3
6-15	Track ID number	G10.3
16-25	Initiation time (minutes from start of simulation)	G10.3
26-35	Multiplicity	G10.3
36-45	Aircraft type	G10.3
46-55	x-position (n.mi.)	G10.3
56-65	y-position (n.mi.)	G10.3
66-75	z-position (ft.)	G10.3
Track Segment Record		
1-3	1	I3
6-15	x-position of end of segment (n.mi.)	G10.3
16-25	y-position of end of segment (n.mi.)	G10.3
26-35	z-position of end of segment (n.mi.)	G10.3
36-45	speed (knots)	G10.3
46-55	end point a target (0 - no, 1 - yes)	G10.3
56-65	jamming (0 - no, 1 - yes)	G10.3
66-75	Probability of destroying target if (46-55) is 1	G10.3
End of Data Record		
1-3	2	I3

Table II-3. Track Data File Layout.

-1	xx	Primary ID Record
0	x---	Track Initiation Record
1	x---	Segment Record
1	x---	" "
	▪	
	▪	
	▪	
0	xxx	one group for each track. In chronological order.
1	xx	
	▪	
	▪	
	▪	
1	xxx	
-1	xxx	Primary ID record
		tracks for this primary ID.
1	xxx	
-1	xx	
1	xx	
2		End of Data

NOTE: Only one group for each primary ID is permitted.

The above features allows entire classes of tracks to be added or replaced easily within an air scenario.

3-0 THE ADD-ASSETS COMMAND.

The ADD-ASSETS command will create or modify a "CRITICAL-ASSET-CONFIGURATION" directory on the working data base. The prompts are shown in Table II-4. IDNUMBER Is a positive integer which will identify the asset configuration. FILE is an external, formatted, sequential data file that specifies the assets to be protected. It's format is shown in Table II-5.

ADD-ASSETS will check to see if an asset configuration with the same IDNUMBER already exists. If not, it will create it and read the contents of FILE into the "COORDINATE-AND-ASSET-DATA" data list. If the specified IDNUMBER does exist, MOPADS will ask if it can be over-written. If not, the command will be aborted. If so, the "COORDINATE-AND-ASSET-DATA" data list will be evicted and completely replaced by the contents of FILE. Any air scenario directories that belong to the asset configuration will not be disturbed. This allows different asset configurations to be used with the same air scenarios if desired.

4-0 THE DELETE COMMAND.

The DELETE command will delete an air scenario or asset configuration from the working or reference data base. If an air scenario is to be deleted, the owning asset configuration must be current.

The prompts for the DELETE command request only the data base and the type (asset configuration or air scenario). MOPADS will request the asset configuration number or air scenario label interactively. (See Table II-6.)

If an asset configuration is deleted, all air scenarios which belong to it are also deleted. Deleted entities cannot be recovered.

5-0 THE GET COMMAND.

The GET command will copy an air scenario or asset configuration from the reference to the working data base. The prompts for GET are shown in Table II-7. The single prompt is DR (for directory). It has a two part response: the type of directory and its identifier. In the case of an air scenario, the identifier is its label. For an asset configuration, the identifier is its ID number. Thus, valid GET commands would be

```
GET  DR=AIR-SCENARIO,AIR1
GET  DR=ASSETS,2
```

Table II-4. The ADD-ASSET Command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: ADD-ASSETS

PURPOSE: To create a CRITICAL-ASSET-CONFIGURATION Directory

MODULE: MOPADS/UI4I

REVISION DATE: 9 MAY 1983

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
IDNUMBER	1001	3	-	-	1	-
FILE	3001	5	-	-	-	-

FILE is rewound prior to reading

Table II-5. The ASSETS Data File Format

COLUMNS	DESCRIPTION	FORMAT
Coordinate Reference Record (First Record in File)		
1-5	REFßß	A5
6-15	reference latitude (decimal degrees)	G10.3
16-25	reference longitude (decimal degrees)	G10.3
26-35	reference altitude (feet above MSL)	G10.3
Asset Records (As many as needed)		
1-5	ASSET	A5
6-15	x-coordinate (n.mi.)	G10.3
16-25	y-coordinate (n.mi.)	G10.3
26-35	z-coordinate (ft.)	G10.3
36-45	site type	G10.3
46-55	site label	A10
56-65	not used	G10.3
End of Data Record (Last record in file)		
1-5	DONEß	A5

NOTE: ß means a blank on the data card.

Table II-6. The DELETE Command.

COMMAND DATA SPECIFICATION					Page 1 of 1
COMMAND: DELETE MODULE: MOPADS/UI41 REVISION DATE: 9 MAY 1983				PURPOSE: To delete an Air Scenario or an entire CRITICAL-ASSET-CONFIGURATION Directory	
PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN / MAX
DB	3001	4	WORKING	WORKING REFERENCE	
TYPE	3001	4	-	AIR-SCENARIO ASSETS	

If an air scenario is being gotten, its owning asset configuration must be current on the reference DB, and the asset configuration to own it on the working DB must be current on the working data base.

If an asset configuration is gotten, all air scenarios which belong to it are also copied to the working DB. GET copies with replacement, so if the specified air scenario or asset configuration already exists on the working DB, MOPADS will as permission to overwrite, and, if given, it will replace the existing version on the working DB with the version on the reference DB.

6-0 THE RENAME COMMAND.

RENAME will change the label of an air scenario or the id number of an asset configuration on the working data base. The prompts are shown in Table II-8. If an air scenario is renamed, its owning asset configuration must be the current directory.

7-0 THE SAVE COMMAND.

The SAVE command is shown in Table II-9. It is analogous in every way to the GET command except it copies air scenarios and asset configurations from the working to the reference data base.

Table II-8. The RENAME Command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: REN^ME

MODULE: MOPADS/UI4I

REVISION DATE: 9 MAY 1983

PURPOSE: To rename an air scenario or CRITICAL-ASSET-CONFIGURATION on the working data base

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DR	3001	4	AIR-SCENARIO	AIR-SCENARIO ASSETS		
OLD	3001	5	-			
NEW	3001	5	-			

Table II-9. The SAVE Command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: SAVE

MODULE: MOPADS/U141

REVISION DATE: 9 MAY 1983

PURPOSE: To copy an Air Scenario or CRITICAL-ASSET-CONFIGURATION from the working DB to the reference DB

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DR	4002					
	3001	4	-	AIR-SCENARIO ASSET		
	3001	5		(air scenario label or asset configuration number)		

III. USER INSTRUCTIONS AND EXAMPLES

1-0 A SIMPLE EXAMPLE.

Table III-1 shows a sample data file containing three sites for a Critical Asset Configuration. This file is in the format given in Table II-5. Similarly, Table III-2 is a simple example of a Track data file in the format explained in Tables II-2 and II-3. Table III-2 contains one hostile track, two friendly tracks, and one "other" track.

The data in Tables III-1 and III-2 is hypothetical and will be used only to demonstrate the use of the commands for this subprocess.

Figure III-1 is a sample terminal session with the Create/Edit Scenario Data subprocess. Information entered by the user has been enclosed in boxes for emphasis. The annotations are explained in the sections that follow.

Table III-1. Simple Asset Configuration.

```
REF      30.2     -20.5     103.
ASSET    10.5      12.2     200.      2.     SITE1
ASSET    -2.3      13.       -2.      2.     SITE2
ASSET     0         0         0       3.     SITE3
DONE
```

2-0 THE ADD-ASSETS AND ADD-AIR COMMANDS.

(1) The Create/Edit Scenario Data subprocess is entered by selecting option 4 from the subprocess option menu. The prompt in this subprocess is

"----ENTER CR/ED SCENARIO COMMAND"

Upon entering this subprocess the "SCENARIOS" directory is current.

Table III-2. Sample Track Data File.

```
-1    1.00
 0    1.00       12.0       1.00        999.      20.0      100.     0.240E+05
 1    20.0       80.0     0.120E+05     650.     0.000     0.000     0.000
 1    10.0       50.0     0.550E+04     650.     0.000     0.000     0.000
 1    5.00       10.0     0.300E+04     450.     0.000     0.000     0.000
 1    -50.0      10.0     0.300E+04     450.     0.000     0.000     0.000
-1    2.00
 0    2.00       55.0       1.00        999.      100.     -45.0     0.350E+04
 1    10.0       -45.0    0.350E+04     250.     0.000     0.000     0.000
 1    -75.0      -45.0    0.350E+04     250.     0.000     0.000     0.000
 0    3.00       61.0       1.00        999.      100.     -30.0     0.550E+04
 1    56.0       -15.0    0.550E+04     300.     0.000     0.000     0.000
 1    -40.0      30.0     0.550E+04     300.     0.000     0.000     0.000
-1    3.00
 0    4.00       20.0       1.00        999.      50.0      75.0     0.310E+05
 1    0.000      25.0     0.310E+05     545.     0.000     0.000     0.000
 1    -100.      -100.    0.310E+05     545.     0.000     0.000     0.000
 2
```

One hostile track

Two friendly tracks

One "Other" track

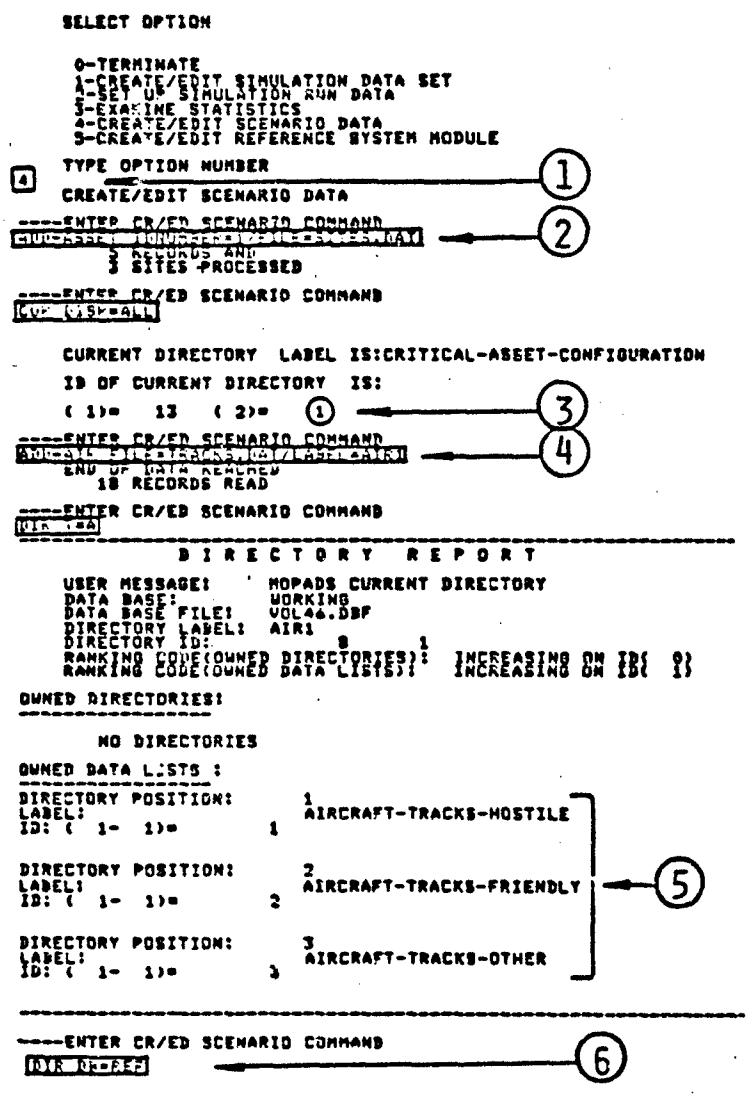


Figure III-1. Example Terminal Session.

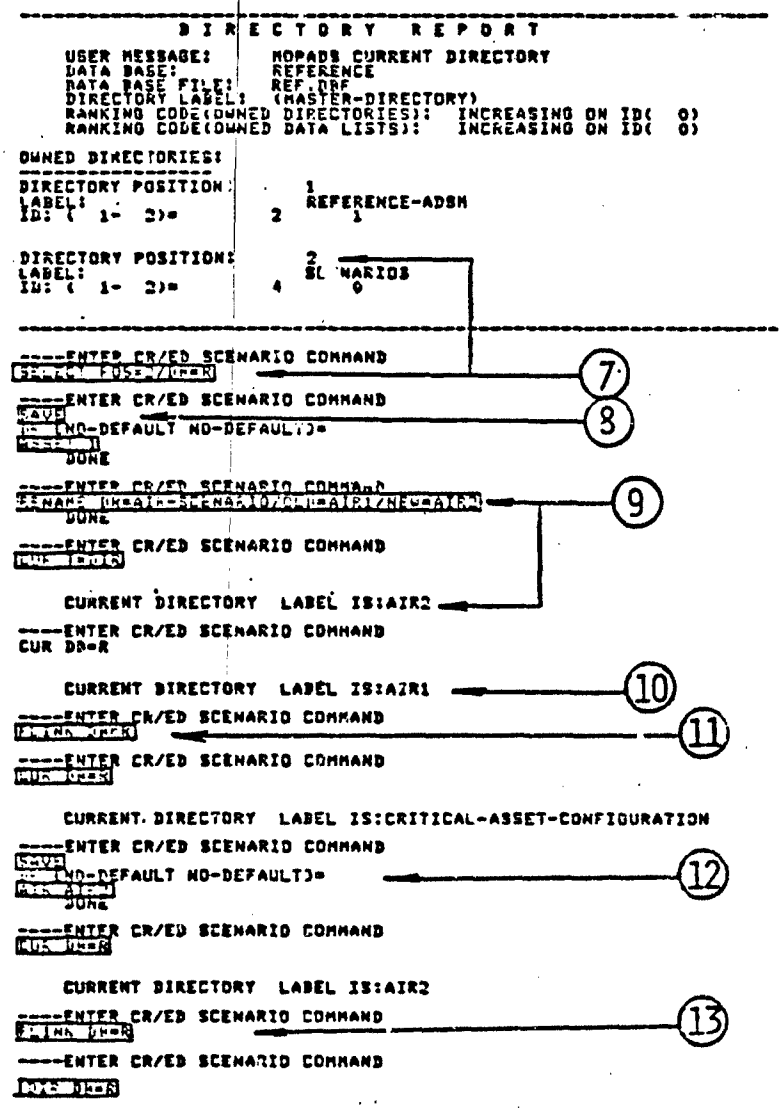


Figure III-1. (continued)

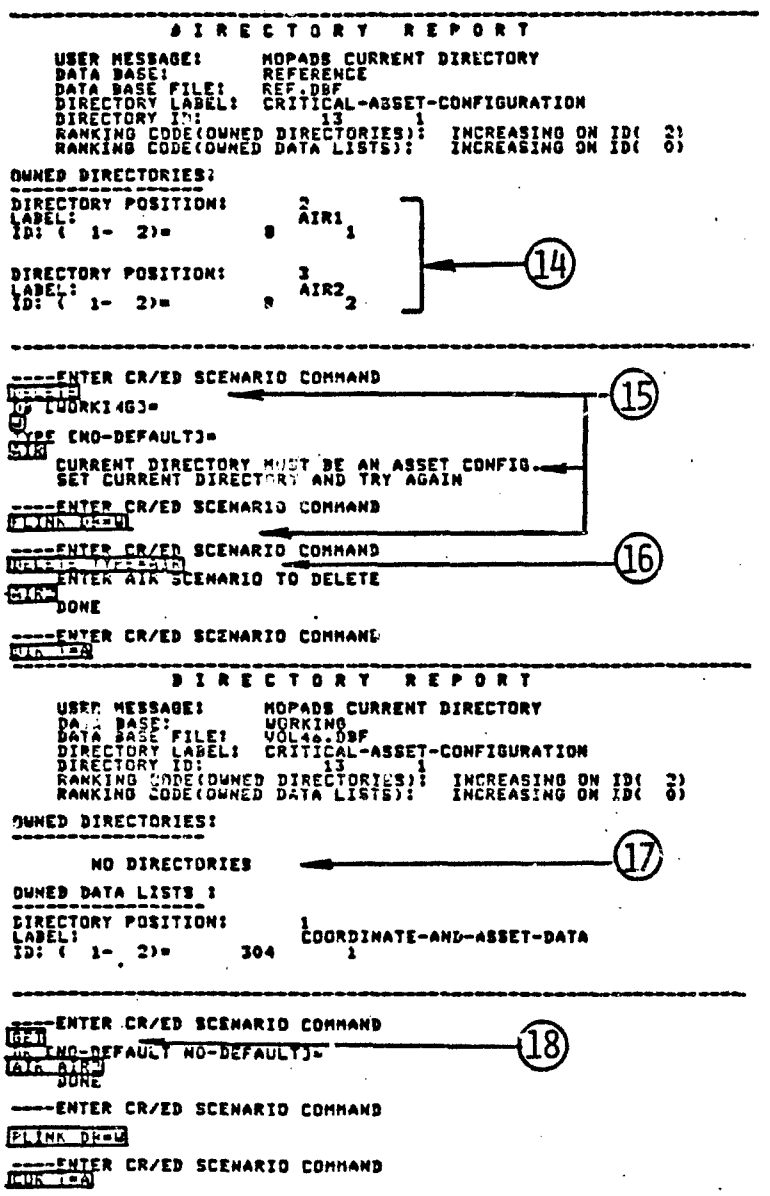


Figure III-1. (continued)

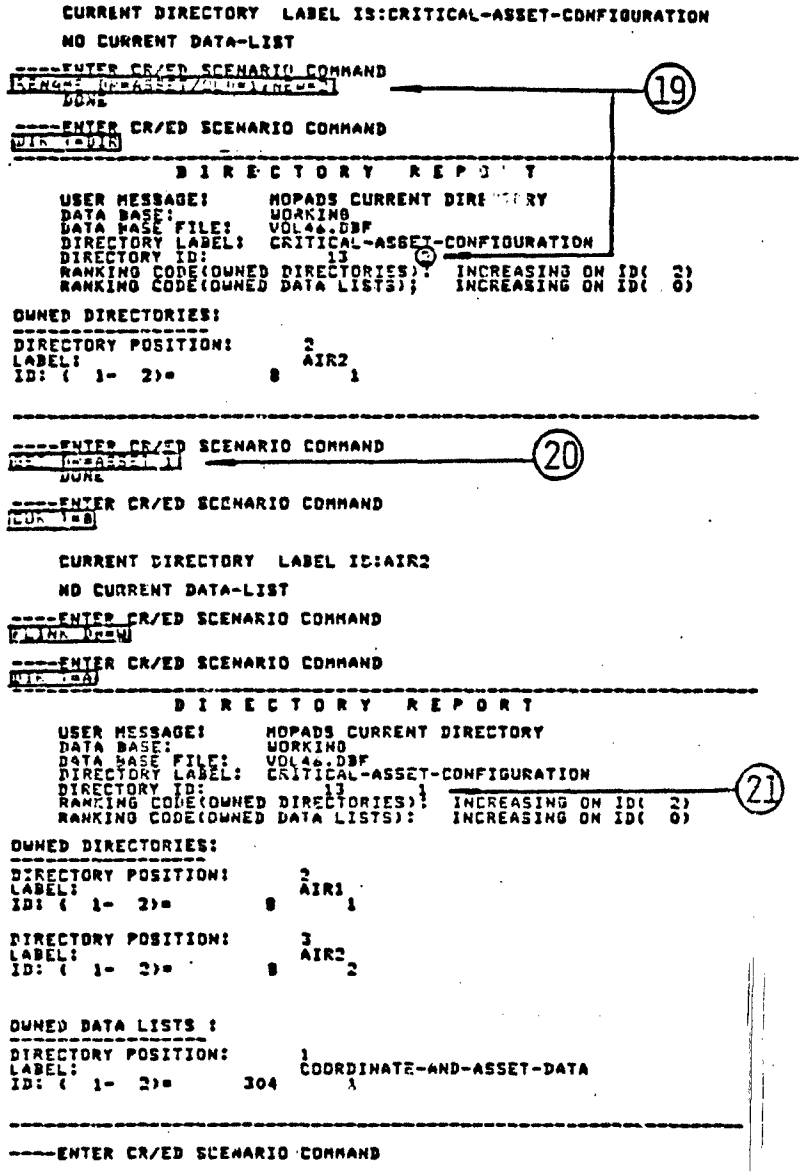


Figure III-1. (continued)

(2) This command specifies that a new CRITICAL-ASSET-CONFIGURATION directory is to be created with ID equal to 1. The asset data is on file SITES.DAT (which is the file shown in Table III-1).

(3) The new asset configuration is current after the ADD-ASSET command. The IDNUMBER of 1 has become the second word of the directory ID. (The first word is the directory code.)

(4) The ADD-AIR command adds a new air scenario directory (with label AIR1) to the newly created asset configuration. The track data is read from the file TRACKS.DAT (which is shown in Table III-2).

(5) The new air scenario, AIR1, is examined with the DIRECTORY command and the three track data lists are present. The contents could be examined with the EXAMINE command. Note that the new air scenario is current after the ADD-AIR command.

3-0 THE SAVE AND RENAME COMMANDS.

(6) A Reference data base has been opened (not shown) and the DIRECTORY command shows that the "(MASTER-DIRECTORY)" is current.

(7) The SELECT command is used to make the "SCENARIOS" directory current on the reference DB.

(8) The SAVE command is used to copy the asset configuration with ID of 1 to the reference DB. This will copy asset configuration 1 and all of its air scenario directories (in this case, only AIR1).

(9) The RENAME command changes the label of the air scenario AIR1 to AIR2 on the working DB. The change is confirmed with the CURRENT command. (See also (19) below.)

(10) We are going to save the air scenario AIR2 to the reference DB. A CRITICAL-ASSET-CONFIGURATION must be current on the reference DB in order to do this. The CURRENT command shows that an air scenario is current, however.

(11) The PLINK command makes the owner of AIR1 on the reference DB the new current directory. This is confirmed by the following CURRENT command. Now the SAVE command can be issued. (If SAVE had been issued when AIR2 was current on the reference DB,

MOPADS would simply have responded with an error message. No harm would have been done.)

(12) This SAVE command copies an air scenario, AIR2, to the reference DB. AIR2 becomes current on the reference DB.

(13) PLINK on the reference DB to make the asset configuration current again.

(14) Both air scenarios are present on the reference DB.

4-0 THE GET AND DELETE COMMANDS.

(15) An asset configuration must be current in order to delete an air scenario. This error message explains this fact. The following PLINK command accomplishes the required action.

(16) The DELETE command deletes AIR2 from the working DB.

(17) The DIRECTORY command confirms that no air scenarios remain.

(18) The GET command copies the air scenario AIR2 from the reference to the working DB.

(19) The RENAME command changes the ID of "CRITICAL-ASSET-CONFIGURATION" 1 to 2. The ID change and the presence of AIR2 is confirmed by the DIRECTORY command.

(20) Now the GET command is used to copy asset configuration 1 from the reference to the working DB.

(21) After using the PLINK command to make the new configuration current, the DIRECTORY command confirms the presence of asset configuration 1 with air scenarios AIR1 and AIR2.

IV. REFERENCES

Goodin, J. R. & Polito, J. MOPADS free-format syntax processor (MOPADS/FFSP) (MOPADS Vol. 5.11). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

V. DISTRIBUTION LIST

Dr. Charles Jorgensen (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 8345
Albuquerque, New Mexico 87198

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.#1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service - Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Ir. K. R. Laughery (2)
Caltpan Corporation
1327 Spruce St. Room 108
Boulder, Colorado 80301

This page intentionally left blank.

VI. CHANGE NOTICES

APPENDIX A. INTRODUCTION TO THE MOPADS USER INTERFACE

1-0 CONVERSING WITH THE MOPADS USER INTERFACE.

1-1. Organization.

The MOPADS User Interface is hierarchical in structure. It consists of five subprocesses as shown in Figure A-1. Each of the subprocesses has its own set of capabilities and commands to invoke them. The subprocesses are described in separate documents. In addition, a set of Basic Data Base Commands are provided that are available in all subprocesses. The main purpose of this section is to explain the use of these commands.

1-2. Syntax Rules.

The User Interface is primarily a command driven processor that waits for the user to issue instructions. It does, however, have aspects of menu driven systems in that some commands result in menus being presented to the user. Also, the command processor (FFSP described in Goodin and Polito (1983) permits menu-like presentations of commands.

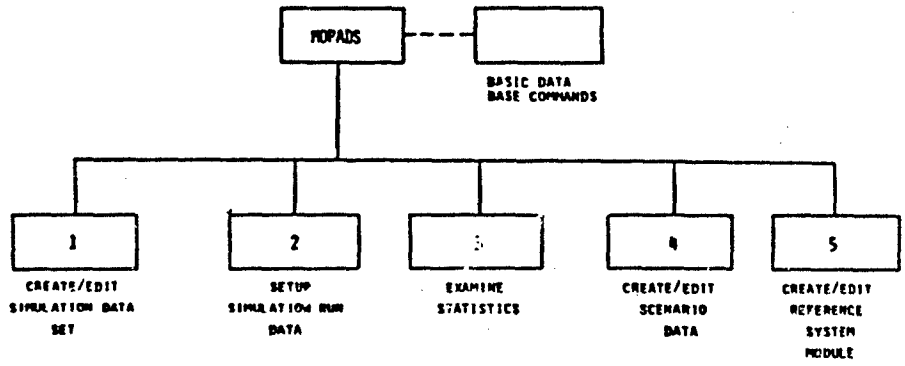


Figure A-1. Organization of the User Interface.

The regular mode for entering commands is shown below.

```
command,prompt1=response1/prompt2=response2/...
```

The commands and prompts are keywords recognized by MOPADS. The responses are particular values for the prompts. For example, consider this.

```
OPEN,FILE=MOPADS.DBF/STATUS=OLD
```

OPEN is the command. FILE and STATUS are prompts recognized by MOPADS, and MOPADS.DBF and OLD are values for the prompts.

Certain prompts for a command may have default values that will be used if the prompt is not entered. In the example above, another prompt, DB, specifies which data base is to be associated with the file. Its default is WORKING, so by not entering it on the command line, WORKING is automatically selected. If the default value is not desired, then the prompt must be explicitly entered on the command line.

If the user fails to enter responses to all required prompts, MOPADS will interactively prompt for them. For example,

```
OPEN,STATUS=OLD

FILE[NO DEFAULT] = MOPADS.DBF
```

After processing the OPEN command, MOPADS found that the required prompt, FILE,was not entered. It printed "FILE[NO DEFAULT]=" to prompt the user for a response. If the last non-blank character on a command line is a dash (-), MOPADS will interactively prompt for all unentered prompts, even those with defaults. For example,

```
OPEN,STATUS=OLD -

DB[WORKING] = REFERENCE

FILE[NO DEFAULT] = MOPADS.DBF
```

The dash caused "DB[WORKING]=" to be printed. The value between the brackets is the default for the prompt. The default can be selected by typing "DEF" as the response. DEF can also be entered on the command line; e.g.

```
OPEN,DB=DEF/STATUS=OLD/FILE=MOPADS.DBF
```

The above demonstrates that the prompt-response groups can be entered in any order.

Also, a command can be cancelled at any time by typing "CANC" as a response or a prompt. For example,

```
OPEN,CANC

OPEN,FILE=CANC
```

Note that DEF and CANC are essentially reserved words. The user interface treats commas, blanks, and equal signs as interchangeable separators. Also, multiple separators are treated as a single separator. This means that the commas in the previous examples could be replaced by any combination of one or more blanks and commas. The same is true of the equal signs, but their use is recommended to make the command lines easy to read. The slashes are required separators between prompt-response groups, but they can be preceded or followed by blanks or commas.

A response may include separators (i.e., commas, blanks, equal signs, and slashes) if it is enclosed in quote marks ("). For example, on some computers file names contain embedded blanks, e.g.,

```
OPEN  FILE="MOPADS DBF"
```

Without the quote marks above, MOPADS will consider MOPADS DBF as two responses when only one is desired. (NOTE: A single prompt may have more than one response if the programmer specified it that way. In such a case, each response would be separated by a blank or comma. In the case above, however, where a single response is required, the quote marks must be used to embed the blank in the response.)

Any response may be enclosed in quotes, although there is no advantage in doing so unless a separator is to be embedded. Blank responses can be entered with " " where at least one blank appears between the quotes.

A generalization of entering only some of the prompts is to enter only the command name:

```
OPEN
DB[WORKING]=DEF
FILE[NO DEFAULT]=MOPADS.DBF
STATUS[OLD]=DEF
```

The User Interface will prompt for all responses. This method can be selected if the user does not remember the prompts.

For commands which the user issues frequently, a concise mode can be selected by preceding the command with "C-". In this case, the prompt= part of the syntax may be omitted. For example,

C-OPEN DEF/MOPADS.DBF

Responses must be entered in the same order as they are prompted in the command-name-only form. No response may be skipped, except that if all remaining responses have defaults and the defaults are desired, then the command line may be terminated (e.g., the STATUS response was omitted above since OLD was desired). The dash works in the concise mode in the same way as in other modes.

The following rules will formalize the previous discussion of how syntax is processed by FFSP.

The command-name-only form of a command may be used at any time by typing only the command name.

Blank responses and responses containing separators may be entered by enclosing them in quotes. To enter a blank response type " " (including the quotes). At least one blank must be entered between the quote marks.

A command may be cancelled at any time by typing CANC for any prompt or response. You can not abbreviate CANC.

The user may elect to use the default value(s) by typing DEF for any response in a response list up to one field past the last response in the list.

Slashes (/) must be used to separate one prompt-response group from another. Blanks or commas may be used to separate all other fields. The equal sign should be used to separate prompts from their responses; however, it is not required.

Command and prompt names may be abbreviated to any non-ambiguous string of characters. For example, if there are two commands, DESIGN and DESCRIBE, they can be abbreviated DESI and DESC respectively. The commands may be abbreviated in longer forms. For example, the user may enter DESC, DESCR, DESCRI, DESCRIB, or DESCRIBE for the command DESCRIBE.

If a command line in regular or concise mode is ended with more than one dash, the last dash will signify to the system to prompt the user for all the unentered responses. Other dashes will then be considered as part of a response.

Any multiple combination of commas and blanks is treated as a single separator. For example,

NAME = BILL WOLF and NAME = BILL , WOLF

are equivalent (here the response is a list of two character strings).

If the user enters an incorrect response or misuses the syntax, FFSP will explain the error and prompt interactively for all remaining responses.

Concise mode is signified by preceding the command name with "C-" (without the quotes).

1-3. The Current Directory and Current Data List.

MOPADS data bases are made up of multiple directories, each of which may own one or more directories and data lists. Normally, the User Interface operates on one directory which is designated "current." Similarly, one data list of the current directory can be designated the current data list. This ensures that the action of User Interface commands are unambiguous. Facilities are provided for determining the current DR and DL and for selecting a DR and DL to become current. Note that some commands automatically change the current DR and DL.

2-0 BASIC DATA BASE COMMANDS.

2-1. CLOSE.

The CLOSE command (Figure A-2) will close either the working or reference data base. It can be used to switch to a new data base file.

2-2. CURRENT.

The CURRENT command (Figure A-3) will display label, ID or both of the current directory and/or data list on either data base.

2-3. DEPOSIT.

DEPOSIT (Figure A-4) is a low level editing command that allows any element of the current data list to be changed. DEPOSIT interactively requests element numbers and new values.

2-4. DIRECTORY.

DIRECTORY (Figure A-5) shows the contents (all owned directories and/or data lists) of the current directory on either data base. It shows the labels, ID's, and directory positions of the contents. This information is useful for the SELECT command.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: CLOSE — PURPOSE: To close a Data Base file.

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
STATUS	3001	4	KEEP	KEEP DELETE		

Figure A-2. CLOSE Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: CURRENT

PURPOSE: To display the identity of the current Directory or Data List

MODULE: NOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
DISPLAY	3001	4	LABEL	LABEL ID ALL		
TYPE	3001	4	DIRECTORY	DIRECTORY DATA-LIST ALL		

Figure A-3. CURRENT Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: DEPOSIT

PURPOSE: To set values of the elements of the current data list

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB interactively prompts for elements and values	3001	4	WORKING	WORKING REFERENCE		

Figure A-4. DEPOSIT Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: DIRECTORY

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: To list selected contents of the current directory on the working or reference DB

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
TYPE	3001	4	DIRECTORIES	DIRECTORY DATA-LIST ALL		
DB	3001	4	WORKING	WORKING REFERENCE		

Figure A-5. DIRECTORY Command Specifications.

2-5. EXAMINE.

EXAMINE (Figure A-6) will display selected contents of the current data list to the terminal or to the MOPADS line printer output file. If the latter is selected, the data list label and other information will also be printed.

2-6. HELP.

HELP (Figure A-7) will print the prompts and options for the prompts for the specified command.

2-7. MENU.

MENU has no prompts. It will print all commands available in the current subprocess.

2-8. OPEN.

OPEN (Figure A-8) will open a data base file as either the working or reference DB. OPEN will not automatically close the current DB. CLOSE must be used explicitly before OPEN to switch DB files. MAXRECORD is the maximum number of records allowed in a data base file. It may not be needed for your computer. Zero implies no limit on the file size. The (MASTER-DIRECTORY) is current after the OPEN command.

2-9. PLINK.

PLINK (Figure A-9) will change the current directory to the owner of the directory which was current when PLINK was issued.

2-10. QUIT.

QUIT has no prompts. It causes the current subprocess to be exited.

2-11. SELECT.

SELECT (Figure A-10) changes the current directory or data list to one that is owned by the directory that is current when SELECT is issued. The desired DR or DL is selected by specifying one (and only one) of the following: 1 - its directory position, 2 - its label, or 3 - its ID. This information is obtained with the DIRECTORY command.

2-12. TERMINATE.

TERMINATE has no prompts. It will close all open data bases and termiante execution. This is the normal way to end a User Interface session.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: EXAMINE

PURPOSE: To display elements of the current data list

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
HEADER	3001	4		NO YES		
DEVICE	3001	4	TERMINAL	TERMINAL PRINTER		

Figure A-6. EXAMINE Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: HELP

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: HELP prints information on the prompts for the command specified in the response

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
COMMAND	3001	5	-	-	-	-

Figure A-7. HELP Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: OPEN — PURPOSE: Open a data base

MODULE: MOPADS/UI

REVISION DATE: 23 NOV 1982

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
FILE	3001	5	-	-		
STATUS	3001	4	OLD	OLD NEW		
DB	3001	4	WORKING	WORKING REFERENCE		
MAXRECORD	1001	3	0	-	0	-

Figure A-8. OPEN Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: PLINK

MODULE: MOPADS/UI

REVISION DATE: 28 FEB 1983

PURPOSE: To make the owner directory of the current directory the new current directory

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		

Figure A-9. PLINK Command Specifications.

COMMAND DATA SPECIFICATION — Page 1 of 1

COMMAND: SELECT

MODULE: MOPADS/UI

REVISION DATE: 6 DEC 1982

PURPOSE: To set the current directory or data list from those in the current directory

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE MIN	RANGE MAX
DB	3001	4	WORKING	WORKING REFERENCE		
TYPE	3001	4	DIRECTORY	DIRECTORY DATA-LIST		
POSITION	1001	3	0	-	0	-
LABEL	3001	5	' '	-	-	-
ID	1000	5	0	-	-	-
(enter only one of the above three)						

Figure A-10. SELECT Command Specifications.

3-0 USER INSTRUCTIONS AND EXAMPLES.

Figure A-11 is an example using all of the Basic Data Base Commands. The annotations are explained below.

(1) - The CLOSE commands disassociates the current data base file from the MOPADS software. After close, MOPADS will know no working DB. If STATUS is specified as DELETE, the data base file will be destroyed permanently.

(2) - OPEN associates the data base file VOL46.DBF to the software as the working DB. STATUS=OLD implies that the DBF was previously created as a MOPADS data base file. If STATUS=NEW is specified, MOPADS will create a new MOPADS DB on VOL46.DBF, and any information previously contained on VOL46.DBF will be lost.

(3) - The MENU command is always available to list the commands currently available.

(4) - HELP prints information about a particular command.

(5) - The user has asked for information on the prompts DISPLAY and TYPE. For DISPLAY, MOPADS shows that it expects a response of one character string that must be one of the enumerated values LABEL (display the label), ID (display the ID), or ALL (display both label and ID). The default is LABEL. HELP is terminated by entering "*."

(6) - When OPEN is issued, the (MASTER-DIRECTORY) becomes current. The CURRENT command will not display a directory which has no owner, which of course, the MASTER-DIRECTORY does not.

(7) - The DIRECTORY command will always work, however, even on the (MASTER-DIRECTORY). The contents of the (MASTER-DIRECTORY) are shown.

(8) - The SELECT command makes the REFERENCE-ADSM directory current by specifying its directory position in the (MASTER-DIRECTORY).

(9) - The DIRECTORY command confirms that REFERENCE-ADSM is current and shows the owned reference ADSM's (in this case only one, E-EXAMPLE).

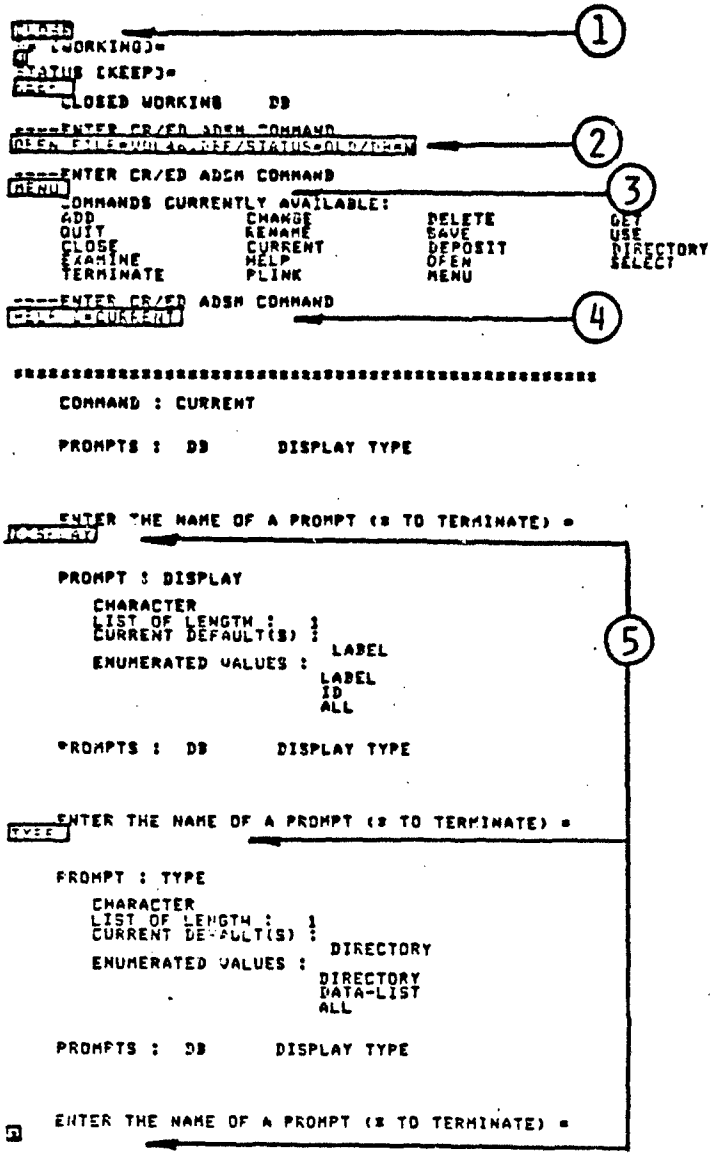


Figure A-11. Example Using Basic Commands.

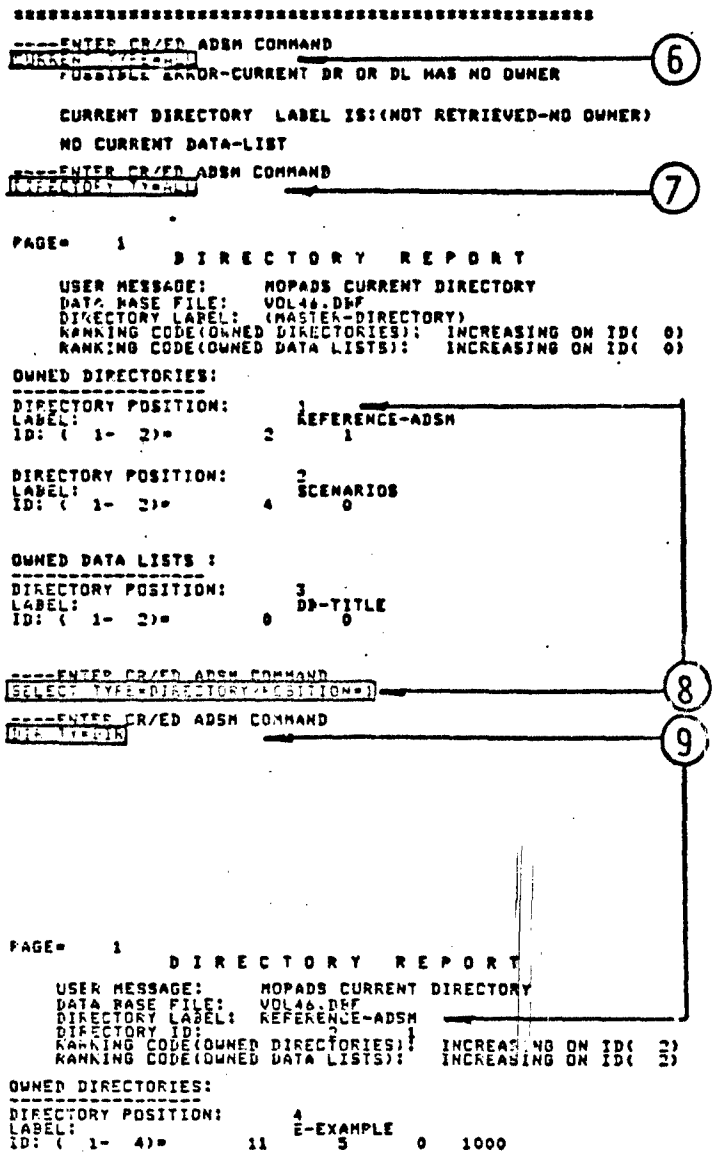
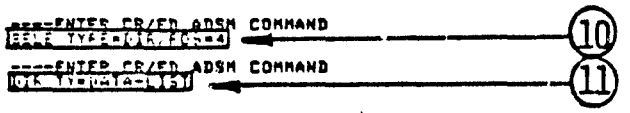
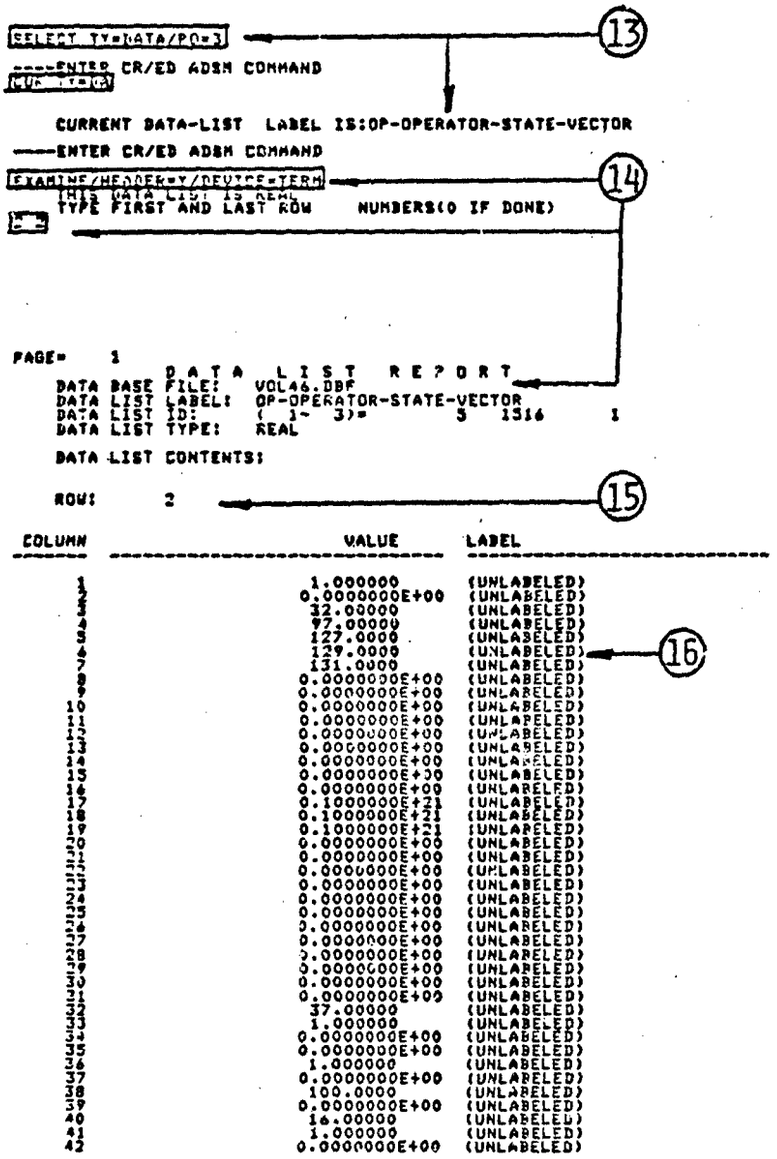


Figure A-11. (continued)




```
PAGE=    1
                    D I R E C T O R Y   R E P O R T

    USER MESSAGE:        MOPADS CURRENT DIRECTORY
    DATA BASE FILE:      VOL46.DBF
    DIRECTORY LABEL:     E-EXAMPLE
    DIRECTORY ID:           11          S        0     1000
    RANKING CODE(OWNED DIRECTORIES):   INCREASING ON ID(  0)
    RANKING CODE(OWNED DATA LISTS):    INCREASING ON ID(  2)

OWNED DATA LISTS :
------------------
DIRECTORY POSITION:        1
LABEL:                     R-ADS-RESOURCES
ID: (  1-  3)=          S    18          1

DIRECTORY POSITION:        2
LABEL:                     EN-ENVIRONMENT-STATE-VEC
ID: (  1-  3)=          S   514          1

DIRECTORY POSITION:        3                                    12
LABEL:                     OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=          S  1516          1

DIRECTORY POSITION:        4
LABEL:                     OP-OPERATOR-STATE-VECTOR
ID: (  1-  3)=          S  1516          2

DIRECTORY POSITION:        5
LABEL:                     OT-OPERATOR-TYPE
ID: (  1-  3)=          S  1520          1

DIRECTORY POSITION:        6
LABEL:                     RL-RESOURCE-LABELS
ID: (  1-  3)=          S  1812          1

DIRECTORY POSITION:        7
LABEL:                     TD-TASK-DATA
ID: (  1-  3)=          S  2004          1

----ENTER CR/ED ADSM COMMAND
```

Figure A-11. (continued).



COLUMN	VALUE	LABEL
1	1.000000	(UNLABELED)
2	0.0000000E+00	(UNLABELED)
3	32.00000	(UNLABELED)
4	97.00000	(UNLABELED)
5	127.0000	(UNLABELED)
6	129.0000	(UNLABELED)
7	131.0000	(UNLABELED)
8	0.0000000E+00	(UNLABELED)
9	0.0000000E+00	(UNLABELED)
10	0.0000000E+00	(UNLABELED)
11	0.0000000E+00	(UNLABELED)
12	0.0000000E+00	(UNLABELED)
13	0.0000000E+00	(UNLABELED)
14	0.0000000E+00	(UNLABELED)
15	0.0000000E+00	(UNLABELED)
16	0.0000000E+00	(UNLABELED)
17	0.1000000E+21	(UNLABELED)
18	0.1000000E+21	(UNLABELED)
19	0.1000000E+21	(UNLABELED)
20	0.0000000E+00	(UNLABELED)
21	0.0000000E+00	(UNLABELED)
22	0.0000000E+00	(UNLABELED)
23	0.0000000E+00	(UNLABELED)
24	0.0000000E+00	(UNLABELED)
25	0.0000000E+00	(UNLABELED)
26	0.0000000E+00	(UNLABELED)
27	0.0000000E+00	(UNLABELED)
28	0.0000000E+00	(UNLABELED)
29	0.0000000E+00	(UNLABELED)
30	0.0000000E+00	(UNLABELED)
31	0.0000000E+00	(UNLABELED)
32	37.00000	(UNLABELED)
33	1.000000	(UNLABELED)
34	0.0000000E+00	(UNLABELED)
35	0.0000000E+00	(UNLABELED)
36	1.000000	(UNLABELED)
37	0.0000000E+00	(UNLABELED)
38	100.0000	(UNLABELED)
39	0.0000000E+00	(UNLABELED)
40	16.00000	(UNLABELED)
41	1.000000	(UNLABELED)
42	0.0000000E+00	(UNLABELED)

Figure A-11. (continued)

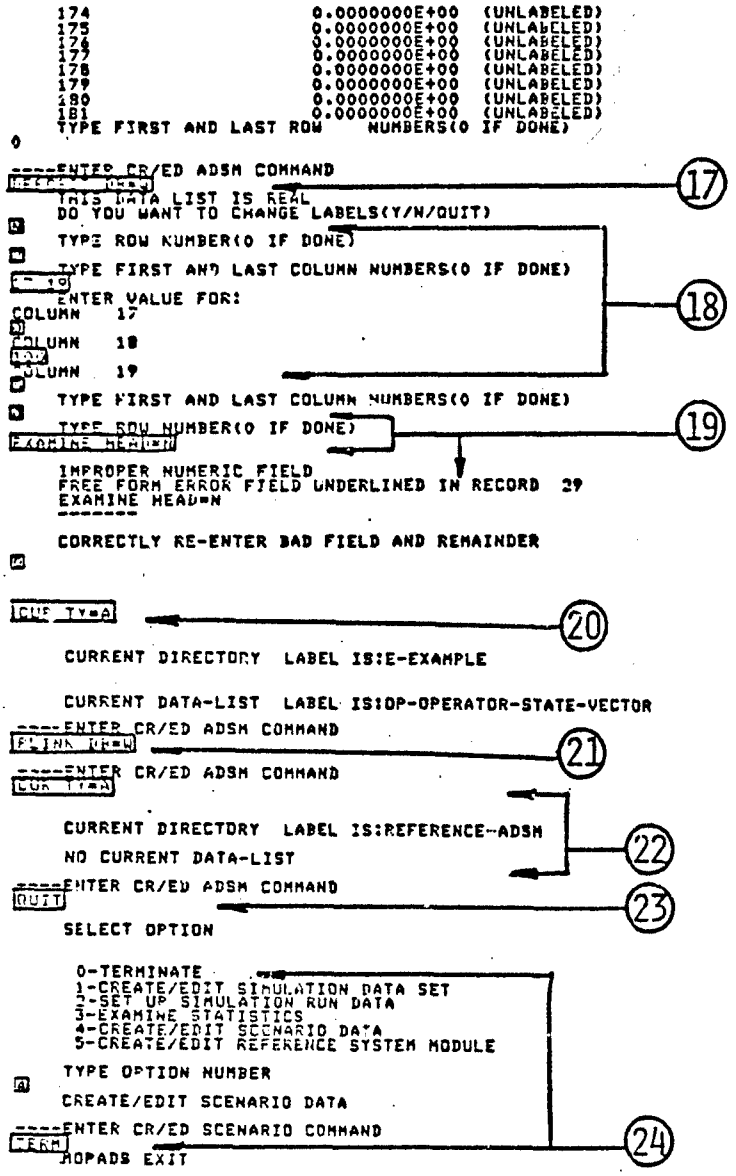


Figure A-11. (continued)

(10) - Select E-EXAMPLE to be current.

(11) - Look at the data list directory of E-EXAMPLE.

(12) - This operator state vector will be edited.

(13) - The SELECT command makes the above operator state vector the current data list. This is confirmed with the following CURRENT command.

(14) - EXAMINE specifies that the data list is to be displayed at the terminal. The user wants row 2. HEADER=Y causes the "DATA LIST REPORT" and the next 6 lines to be printed.

(15) & (16) - This display demonstrates that the basic data base commands are low level commands; they know nothing about the structure or contents of MOPADS data bases. The operator state vectors have two rows. The second row contains the reference values for each column. Prior to a simulation, the second row is copied to the first row, which is dynamically accessed during the simulation. In this way, the reference or starting values are not lost. MOPADS stores column labels only for the first row, however, since it would double the storage requirement to duplicate the labels for each row. The "CREATE/EDIT REFERENCE SYSTEM MODULE" subprocess knows all of this, so when a listing is obtained with the CHANGE command in that subprocess, the labels appear with the reference values.

The EXAMINE command, however, simply operates on a data list without knowledge of the meaning of its contents. Here, all of the elements of row 2 appear to be (UNLABELED). Also, EXAMINE prints the entire row. Only a portion of the listing is shown here for brevity.

(17) - Use the DEPOSIT command to change elements. It too is a low level command, so (row, column) addressing must be used.

(18) - Elements 17, 18, and 19 of row 2 are changed (these are operator trace parameters).

(19) - This demonstrates MOPADS response to input errors. The DEPOSIT command had not yet terminated when the user entered the next EXAMINE. MOPADS is expecting a numeric value for a row number. The free-format input processor examines every input string for

validity before processing it. Since "EXAMINE" is not a valid number, it prints the error message and requests correct input. Without this feature, the program would terminate abnormally from simple typing errors and perhaps damage the data base. MOPADS will protect the user from most such errors.

(20) - CURRENT shows that E-EXAMPLE is the current directory.

(21) - PLINK makes the current directory equal to the owner of the directory that was current when PLINK was issued.

(22) - The CURRENT command confirms that the REFERENCE-ADSM directory is now current. Note that when the current directory is changed, the current data list becomes undefined.

(23) - QUIT exits the current subprocess and brings up the subprocess option menu again.

(24) - The TERM command terminates execution. Selecting the zero option on the subprocess selection menu accomplishes the same thing.

APPENDIX K

MOPADS FINAL REPORT:

A SUMMARY OF THE LITERATURE ON QUANTITATIVE HUMAN PERFORMANCE MODELS

TABLE OF CONTENTS

This report is comprised of two volumes, VOLUME 1 and VOLUME 2.

LIST OF FIGURES

LIST OF TABLES

I INTRODUCTION

This report summarizes the literature which was reviewed as part of the MOPADS effort. In this section, we will describe the literature database search logic used and the way in which the literature was summarized.

1.1 SEARCH LOGIC

Essentially, two search modes were employed. First, a computer literature search was conducted on the DTIC and Psychological Abstracts files of the DIALOG database. The underlying search theme was Quantitative Human Performance Models, although an iterative search procedure yielded a number of key words which might yield literature relevant to this topic. A total of 350 abstracts were reviewed, from which 115 articles were selected as potentially useful for the proposed effort.

During the review, the reference sections of particularly interesting articles were examined. Those references which were not already in the inventory were ordered and also reviewed. The two most noteworthy articles with respect to good reference sources are listed under reference notes at the end of this section.

Each article was given a cursory review to determine its potential utility, and if this decision was positive, a more thorough review was conducted. This thorough review was summarized on a form, see Figure 1.

1.2 LITERATURE SUMMARY FORMAT DESCRIPTION

Each model within an article was summarized on a form as shown in Figure 1. Because some articles included a number of human performance models, it was frequently the case that an article was summarized on a number of forms.

The categories on the forms are defined as follows:

1. Reference - Information regarding author(s), title, and other source data.
2. Appropriateness - A judgement was made by the reviewer regarding the extent to which this article would be useful. These judgements were based upon two criteria. First, did the article present a significant contribution to human performance modeling? If the answer was yes, the article was appropriate. If not, did the article provide data which would be useful for modeling or validating air defense systems? In other words, while the article may not be generally useful for modeling, could it provide some task specific information? If so, the article was appropriate. Appropriateness was treated as a continuous variable, with answers ranging from yes to maybe to no.
3. Dependent Variables - The specific dependent variable(s) defined by the model are described.
4. Independent Variables - Independent variables which were included as mediators of the dependent variables are defined. No units were recorded, but a short description was included.
5. Page - The page(s) where the model was best summarized was recorded. This was intended to simplify future references to the model.
6. Behaviors - The general behavior categories which were modeled were defined. These categories were selected from the taxonomy which is presented in Table 1. Normally, a model would fit under only one behavior category. If this was not the case, all relevant behavior categories were listed.
7. Quantitativeness - The degree of quantitative model development was recorded. The lowest degree of quantitativeness was nominal, or zero quantitativeness. This

implied that only a verbal description of the relationship between the independent and dependent variables was provided. The next level was unscaled graphs, in which the independent and dependent variables were presented as constructs without specific units. The third level was tables, or scaled graphs, in which units were attached to independent and dependent variables. Finally, the highest level was functional in which case specific mathematical models were presented.

8. Supporting Data - The extent to which themodel was supported by data was determined. If not data or references were presented. "No Validation" was indicated. If the data were questionable, this was also indicated. Admittedly, this judgement was largely subjective, however, it may help us identify those models which we wish to incorporate later. At some point, this judgement must be made.

9. Comments - Additional comments were recorded.

Reference Notes

Siegel, A., et al, Human Performance in Continuous Operations, 1979.

Pew, R., et al, A Critical Review and Analysis of Performance Models Applicable to Man-Machine Systems Evaluation, 1977.

Reference

Appropriate

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Figure 1. Literature Survey Form.

Table 1

SKILLS TAXONOMY

GENERAL BEHAVIORS
VIGILANCE
PROBABILITY ESTIMATION
TIME ESTIMATION
HUMAN RELIABILITY ASSESSMENT
MEANINGFUL MEMORY ABILITY
VERBAL KNOWLEDGE
WORD FLUENCY
NUMERICAL ABILITY
CONCEPT FLUENCY
DISCOVERY OF PRINCIPLES
FLEXIBILITY
SYMBOL MANIPULATION
DECISION MAKING
INTELLEGENCE
FINE MANIPULATIVE ABILITIES
POSITION ESTIMATION
GROSS MANIPULATIVE ABILITIES
CONTROL PRECISION
SPEED OF ARM MOVEMENT
MULTILIMB COORDINATION
POSITION REPRODUCTION
MOVEMENT ANALYSIS
MOVEMENT PREDICTION
ACCELERATION CONTROL
REACTION TIME
TRACKING
DETECTION
DISCRIMINATION ABILITIES
TIME SHARING
CLOSURE ABILITIES(PERCEPTUAL)
AUDITORY ABILITIES
SPATIAL ABILITIES-ORIENTATION
SPATIAL ABILITIES-VISUALIZATION
ASSOCIATE MEMORY-ROTE MEMORY
ASSOCIATE MEMORY-MEANINGFUL MEMORY
MEMORY SPAN-SHORT TERM MEMORY
MEMORY SPAN-INTEGRATION OF INFORMATION
VISUAL MEMORY
EXPLOSIVE STRENGTH-GENERAL

Table 1 (continued)

EXPLOSIVE STRENGTH-LEG EMPHASIS
EXPLOSIVE STRENGTH-UPPER BODY EMPHASIS
STATIC STRENGTH-ARM/HAND/SHOULDER
STATIC STRENGTH-LEG/TRUNK
DYNAMIC STRENGTH-ARMS
DYNAMIC STRENGTH-LEGS
TRUNK STRENGTH
EXTENT FLEXIBILITY
DYNAMIC FLEXIBILITY
GROSS BODY EQUILIBRIUM
BALANCE-VISUAL CUES
GROUP COOPERATION
GROUP COMMUNICATION
LEADERSHIP
GENERAL COGNITIVE ABILITIES
GENERAL PHYSICAL ABILITIES

II. LITERATURE SURVEY

Reference

Siegel, A.I., Pfeiffer, M.G., Kopstein, F.F., Wilson, L.G., and Ozkaptan, H. Human performance in continuous operations: Volume I. Human performance guidelines. Alexandria, VA: US Army Research Inst. for the Behavioral Sciences, Research Product 80-4a, December 1979, Contract No. DAHC-19-C-0054. ADA086131

Appropriate

High

Dependent Variables

Effectiveness estimate for armor tasks

Independent Variables

Mission hours;
Platoon leader,
Squad leader
Gunner/Carrier team leader
Maneuver team member

Page

76

Behaviors

General

Quantitativeness

Unscaled Graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Refsrence Siegel, et al (1979)
Contract No. DAHC-19-C-0054

Appropriate High

Dependent Variables Effectiveness estimate for armor tasks

Independent Variables
Mission hours
Squad and platoon

Page 78

Behaviors
General

Quantitativeness
Unscaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel et al (1979)
Contract No. DAHC-19-C-0054

Appropriate Yes

Dependent Variables

Effectiveness estimate for armor tasks

Independent Variables

Mission hours
Best and worst conditions for mechanized infantry

Page 79

Behaviors General

Quantitativeness Unscaled graphs

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, et al (1979)
Contract No. DAHC-19-C-0054

Appropriate High

Dependent Variables

Effectiveness estimate for armor tasks

Independent Variables

Duty position;
Squad vs. platoon; day 1,3,5

Page 77

Behaviors General

Quantitativeness

Table

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, et al (1979)
Contract No. DAHC-19-C-0054

Appropriate

High

Dependent Variables

Projected effectiveness rating for critical combat armor tasks

---

Independent Variables

Specific task
Duty position
Platoon action

Page 82-86

Behaviors General

Quantitativeness

Table

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	High
Dependent Variables	Effectiveness rating for armor tasks
Independent Variables	Duty positions, squads and platoons, various adverse conditions
Page	81
Behaviors	General
Quantitativeness	Table

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054	1
Appropriate	High	
Dependent Variables	Mean time off target; mean # of times off target	
Independent Variables	Hours of work with 15 minutes rest every 5 hours, for 15 hours	
Page	93	
Behaviors	Tracking	
Quantitativeness	Scaled graphs	
Supporting Data	[x] No Validation [ ] Validation with Questionable Results [ ] Validated	
Comments		

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	High
Dependent Variables	Percentage of signals detected
Independent Variables	Test session 10 minute blocks; varied number of signals; varied proportion of target signals, among all signals
Page	97
Behaviors	Detection
Quantitativeness	Scaled Graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0034
Appropriate	High
Dependent Variables	Mean choice RT (seconds)
Independent Variables	Number of alternative responses
Page	103
Behaviors	Reaction Time
Quantitativeness	Scaled Graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	Hick's Law

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	High
Dependent Variables	Pointing error (degrees)
Independent Variables	Practice (4 trials) or familiarity with area; Good vs. poor sense of direction
Page	106, 107
Behaviors	spatial orientation, spatial visualization, position reproduction; meaning full memory ability
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	Graph and nomograph

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	probability of detecting a moving target
Independent Variables	Target distance traveled, moving toward or away from viewer at 9km/hr; Target brightness in foot lamberts
Page	111
Behaviors	Detection
Quantitativeness	Scaled graphs
Supporting Data	☒ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Distance of target travel (meters)
Independent Variables	Luminance of surround (foot lamberts); monocular vs. binocular viewing movement is toward or away from viewer
Page	111
Behaviors	Detection
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Time to detect (sec.)
Independent Variables	Speed of tank (km/hr); Monocular vs. binocular viewing
Page	111
Behaviors	Detection
Quantitativeness	Scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	No
Dependent Variables	Handstrength (0/0)
Independent Variables	Skin temperature (°F) Gross vs. fine manipulation
Page	112, 125
Behaviors	Dynamic and static strength - hand. Manual dexterity
Quantitativeness	Unscaled graphs Nomograph

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

This is extremity strength, not gross body movement ability or fine manipulation.
This may be an approximation to fine manipulation ability

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Number of arithmetic sums done (numerical ability)
Independent Variables	Time of day; days of continuous duty night duty
Page	117
Behaviors	numerical ability
Quantitativeness	unscaled graphs

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0034	1
Appropriate	Yes	
Dependent Variables	Percentage of efficiency	
Independent Variables	Skin temperature of hand Two manipulations and dynamometer	
Page	125	
Behaviors	Fine manipulative abilities	
Quantitativeness	Unscaled graphs	

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Change in time to complete task (sec).
Independent Variables	Hand-skin temperature exposure time (min)
Page	125
Behaviors	Fine manipulative abilities
Quantitativeness	Scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Knot tying task

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	percent detections
Independent Variables	hours of sleep deprivation 1 or 2 nights
Page	127
Behaviors	auditory detection
Quantitativeness	Scaled graphs

Supporting Data

- [*] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	percentage error
Independent Variables	Time per movement (.2 to 1.4 sec) eyes closed (no light) vs. eyes open
Page	130, 131
Behaviors	Gross positioning and movement abilities; perceptual speed
Quantitativeness	Unscaled graphs
Supporting Data	☒ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	Nomograph p. 131

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Words heard perfectly (0/0)
Independent Variables	Noise to speech ratio (ratio and db) with and without seeing speaker
Page	134
Behaviors	Auditory identification abilities
Quantitativeness	scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments	Decrement due to lack of vision under degraded speech conditions. See also p. 177

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	Minimum target detection distance
Independent Variables	Light level; contrast ratio type of target (tank, jeep, person)
Page	141
Behaviors	Detection
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	Nomograph p. 142

Reference	Siegel, et al (1979) Contract No. DAHC-19-C-0054
Appropriate	Yes
Dependent Variables	percent increase in errors
Independent Variables	Time of day; days without sleep
Page	146
Behaviors	perceptual speed, discrimination
Quantitativeness	unscaled graphs
Supporting Data	☒ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	Recommended hours of recovery.
Independent Variables	Hours of sleep loss, starting at 24
Page	146
Behaviors	General
Quantitativeness	Scaled Graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	See p. 93, 117 also

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	hits (shooting personal weapons); variability
Independent Variables	sleepless days, anchor performance before sleeploss and after recovery; 0, 1.5,3 hours sleep/night predictable targets - p. 149 unpredictable targets - p. 150
Page	149, 150
Behaviors	discrimination, perceptual speed; gross positioning fine manipulative abilities
Quantitativeness	scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	See p. 146, 93, 117 also

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	percent of signals detected
Independent Variables	minutes on watch; auditory, visual, both
Page	154
Behaviors	detection, auditory detection
Quantitativeness	unscaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	Nomograph on p. 155

Reference	Siegel, et al (1979)
Appropriate	med.
Dependent Variables	0/0 visual acuity
Independent Variables	degrees from fovea
Page	168
Behaviors	detection, discrimination
Quantitativeness	unscaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	nomograph p. 169

Reference	Siegel et al (1979)
Appropriate	Yes
Dependent Variables	mental performance limit tolerable physical limit
Independent Variables	exposure time effective temperature or wet and dry bulb temp.
Page	171
Behaviors	nominal
Quantitativeness	
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Siegel, et al (1979)
Appropriate	Med.
Dependent Variables	Speech interference level ind b
Independent Variables	Distance between talker and listener
Page	177
Behaviors	auditory identification abilities
Quantitativeness	scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

See also p. 134
Nomograph on p. 178

Reference Siegel, et al (1979)

Appropriate

Yes

Dependent Variables

percentage of errors

Independent Variables

type of noise - unpredictable vs. predictable
loud (108db) vs. soft (56db)

Page 184

Behaviors

general reasoning

Quantitativeness

unscaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

See p. 134, 177

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	probability of correct report
Independent Variables	probability of a false report divided vs. undivided attention
Page	187
Behaviors	time sharing
Quantitativeness	scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference Siegel, et al (1979)

Appropriate No

Dependent Variables

Independent Variables

effects of overlearning and recall

Page 191

Behaviors General

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	mean accuracy
Independent Variables	number of sleepless days of 0, 1.5, 3 hrs sleep per night also, contains data on ability prior to sleeplessness and following recovery
Page	194
Behaviors	discrimination
Quantitativeness	unscaled graphs

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

See also p. 117, 127, 146

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	Mean number of correct computations
Independent Variables	Time of day; days during a 15-day 4 hour on 2 hour off work-rest schedule
Page	199
Behaviors	numerical ability
Quantitativeness	scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	See also p. 117, 127, 146, 194 Nomograph p. 200

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	Decision accuracy
Independent Variables	Sleepless days; 0, 1.5, 3 hours sleep/night Data on performance before sleeplessness and following recovery
Page	206
Behaviors	General reasoning, seeing implications, practical judgment
Quantitativeness	Unscaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	See also p. 117, 127, 146, 194, 199

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	mean number of map reading errors
Independent Variables	Sleepless days; 0, 1.5, 3, hours sleep/night Performance data before sleeplessness and following recovery
Page	209
Behaviors	Spatial abilities
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Siegel, et al (1979)
Appropriate	
Dependent Variables	percent correct recall
Independent Variables	Duration of performance of an irrelevant task; 1 to 5 items to be remembered
Page	212
Behaviors	Meaningful memory ability
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	Nomograph p. 213

Reference	Siegel et al (1979)
Appropriate	Yes
Dependent Variables	Recovery time
Independent Variables	Adapting flash energy; target luminance
Page	218
Behaviors	Vision
Quantitativeness	scaled graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference	Siegel, et al (1979)
Appropriate	Yes
Dependent Variables	Freedom from distraction
Independent Variables	Sleepless days; 0, 1.5, 3 hours sleep/night Performance data before sleeplessness and following recovery
Page	221
Behaviors	Conceptual and thinking abilities
Quantitativeness	Unscaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Roe, W.T. and Finley, D.L. Ergonomic models of human performance; source materials for the analyst. Arlington, VA: Office of Naval Research, Technical Dept. August 1975, Contract No. N00014-74-C-0324 ADAD 20086
Appropriate	Medium
Dependent Variables	Energy expenditure
Independent Variables	Occupations, activities, postures, types of work
Page	31-33
Behaviors	strength
Quantitativeness	Tables
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	

Reference	Roe and Finley (1975)
Appropriate	Moderate
Dependent Variables	Stress effects; heart rate; body temperature; sweat rate
Independent Variables	heat stress
Page	43
Behaviors	Physiological changes
Quantitativeness	Unscaled graphs

Supporting Data

- ☐ **No Validation**
- ☐ **Validation with Questionable Results**
- ☒ **Validated**

Comments

Intervening variable used as a predictor.

Reference	Roe and Finley (1975)
Appropriate	Moderate
Dependent Variables	Production per hour, production per week, production rate
Independent Variables	hours worked per week
Page	63
Behaviors	productivity
Quantitativeness	unscaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	Predictor for a generic behavior

Reference

Roe & Finley (1975)

Appropriate

High

Dependent Variables

Sensing, identifying, interpreting, controlling

Independent Variables

sensing,
identifying;
interpreting;
controlling

Page

52

Behaviors

sensing, identifying, interpreting, controlling

Quantitativeness

nominal

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

structural models

Reference	Pfeiffer, MG and Siegel, A.I. Model development and the assessment of competing models. Santa Monica, CA; Human Factors Society, proceedings of the 18th Annual Meeting, 1974, 425-428
Appropriate	Yes
Dependent Variables	Aggregate utility U_i
Independent Variables	Importance weight of the j^{th} dimension
Page	427
Behaviors	System equalization abilities
Quantitativeness	Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Pfeiffer, M.G., Siegel, A.I., Taylor, S.E., and Shuler, L., Jr. Background data for the human performance in continuous operations guidelines, Alexandria, VA: US Army Research Institute for the Behavioral and Social Sciences, Technical Report 386, July 1979, Contract No. DAHC19-77-C-0054
Appropriate	High
Dependent Variables	Minimum perceptible visual angle
Independent Variables	Background luminance (log foot-lamberts)
Page	11
Behaviors	Discrimination
Quantitativeness	Scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Pfeiffer and Seigel (1979)

Appropriate

Medium

Dependent Variables

Time to recover visual ability from a blinding flash (seconds)

Independent Variables

Log adapting flash energy

Page

12

Behaviors

Discrimination

Quantitativeness

Scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference	Pfeiffer and Seigel (1979)
Appropriate	High
Dependent Variables	probability of detection
Independent Variables	visual angle
Page	11
Behaviors	Discrimination
Quantitativeness	Scaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Data from Blackwell (1946). Actual presentation conditions uncertain

Reference	Pfeiffer, et al (1979)
Appropriate	High
Dependent Variables	Time to recover an intellective-motor ability following a blinding flash
Independent Variables	Visual recovery time; Work area luminance in foot lamberts; Minimum luminance to perceive the work under optimum conditions; energy of flash in ft.l seconds
Page	13
Behaviors	Discrimination
Quantitativeness	Functions
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Medium
Dependent Variables	Speech interference level in decibels
Independent Variables	Distance in feet between talker and listener
Page	14
Behaviors	Scaled graphs
Quantitativeness	
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Med.
Dependent Variables	Rest required following energy expenditure above some standard
Independent Variables	Total working time in minutes; energy cost in Cal/min; energy standard
Page	15, 18 (figure)
Behaviors	Stamina
Quantitativeness	Functions
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	The figure on p. 18 is for several energy standards, and is expressed in Cal/min for energy and minutes rest per hour worked.

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Mean choice Rt
Independent Variables	Number of possible stimuli
Page	17
Behaviors	Reaction time
Quantitativeness	Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Hick's Law may be too theoretical and over simplified for MOPADS. However, it may be as good an approximation as any other formula. K is the problem

Reference	Pfeiffer, et al (1979)
Appropriate	Med.
Dependent Variables	Rest required following energy expenditure above some standard
Independent Variables	Total working time in minutes; energy cost in Cal/min; energy standard
Page	15, 18 (figure)
Behaviors	Stamina
Quantitativeness	Functions
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	The figure on p. 18 is for several energy standards, and is expressed in Cal/min for energy and minutes rest per hour worked.

Reference	Pfeiffer et al (1979)
Appropriate	Med.
Dependent Variables	Percent of omissions
Independent Variables	Speed X load (signals/min) X (number of dials monitored)
Page	17, 19
Behaviors	Time sharing
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	Useful for a guess at the slope of the linear function

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	percent efficiency
Independent Variables	Skin temperature of hand three tasks
Page	21
Behaviors	Fine manipulative abilities
Quantitativeness	Unscaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	See Siegel (1) for same graph

Reference	Pfeiffer et al (1979)
Appropriate	Yes
Dependent Variables	Change in time to complete knot-tying task (seconds)
Independent Variables	Exposure time (to cold) Two hand-skin temperatures (55° and 60°F)
Page	21
Behaviors	Fine manipulative ability
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☒ Validated
Comments	

Reference

Pfeiffer, et al (1979)

Appropriate

Yes

Dependent Variables

Productivity index

Independent Variables

Wet bulb temperature

Page

23

Behaviors

Cross body movement abilities

Quantitativeness

Unscaled graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Error
Independent Variables	Work/rest schedule (5/5 to 60/60 minutes/minutes) Vibration/normal environment
Page	23
Behaviors	Fine manipulative ability
Quantitativeness	Unscaled graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Transmissibility of whole body vibration
Independent Variables	Frequency (Hz) Types of seats
Page	25
Behaviors	General
Quantitativeness	Unscaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	This graph is for conversion use

Reference	Pfeiffer, et al (1979)
Appropriate	
Dependent Variables	Qualitative assessment
Independent Variables	Frequency (Hz) Acceleration (g) due to whole body vibration
Page	25
Behaviors	See comment
Quantitativeness	unscaled graphs

Supporting Data

- ☐ **No Validation**
- ☐ **Validation with Questionable Resuits**
- ☐ **Validated**

Comments

This graph is for assessment of subjective feeling

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Tracking error
Independent Variables	Days of sleep deprivation; Three measures per day; Administration of amphetamine
Page	27
Behaviors	Psychomotor abilities
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Recommended hours recovery
Independent Variables	Cumulative hours of sleep loss
Page	27
Behaviors	General
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference Pfeiffer, et al (1979)

Appropriate

Yes

Dependent Variables

Body temperature

Independent Variables

Days of night duty;
Time of day

Page

28

Behaviors

General

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

For conversion purposes

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Number of sums done
Independent Variables	Days of Night duty Time of day
Page	28
Behaviors	Numerical ability
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	0/0 of failures relative to 24 hour mean
Independent Variables	Time of day (9 A.M. to 9 P.M.) Delay in answering telephone calls Frequency of errors in reading gas meters Frequency of falling asleep while driving Frequency of errors in answering warning signals.
Page	28
Behaviors	Practical judgment, logical evaluation, discriminability Time sharing; visual memory
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Limits for mental performance and physical tolerance
Independent Variables	Effective temperature; Exposure time
Page	30
Behaviors	Conceptual and thinking; psycho-motor
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Comfort zones 0/0 relative humidity Effective Temperature
Independent Variables	Dry-bulb, wet bulb temperature
Page	30
Behaviors	Mediating variables
Quantitativeness	Scaled graphs

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Mean percent correct responses
Independent Variables	Time duration (1-6 sec) following a 1 sec 11 odb 5 PL noise Noisy environment vs. quiet environment
Page	31
Behaviors	perceptual speed, practical judgment, immediate memory
Quantitativeness	scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Average percent of errors while proofreading
Independent Variables	Unpredictable vs. predictable vs. no noise; 108 db vs. 56 db
Page	31
Behaviors	Word fluency, practical judgment, visual memory
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Mean percentage of letters recalled correctly
Independent Variables	Delay of instructions (.5 sec before to 5 sec. after presentation) light vs. dark
Page	33
Behaviors	immediate memory
Quantitative	scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	No
Dependent Variables	Retention after learning
Independent Variables	Days since learning; 100%, 50%, 0% overlearning
Page	36
Behaviors	Rate memory
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Pointing error (degrees)
Independent Variables	Practice (trials 1-4); good vs. poor sense of direction persons
Page	36
Behaviors	Spatial abilities
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	Large variability among persons

Reference	**Pfeiffer - et al (1979)**
Appropriate	**Yes**
Dependent Variables	Obtained B (beta)
Independent Variables	Optimal B (beta); probabilities variable; Values variable
Page	39
Behaviors	General reasoning; seeing implications, practical judgment; memory span, integration
Quantitativeness	Scaled graphs
Supporting Data	[X] No Validation [ ] Validation with Questionable Results [ ] Validated
Comments	This illustrates the suboptimal decision making strategy of humans. Underestimation of high beta and overestimation of low beta.

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Probability of a correct report
Independent Variables	Probability of a false report control vs. divided attention
Page	39
Behaviors	time sharing, memory span, integration
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Words correct (%)
Independent Variables	Speech-to-noise ratio (dB) Auditory plus visual cues vs. auditory cues only
Page	41
Behaviors	Auditory perceptual speed
Quantitativeness	Scaled graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference	Pfeiffer, et al (1979)
Appropriate	Yes
Dependent Variables	Speed of communication Richness of communication
Independent Variables	face to face, voice, typed communication modes
Page	40 and 41
Behaviors	Auditory perceptual speed, communication speed
Quantitativeness	Scaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	Chaponis experiments

Reference Pfeiffer, et al (1979)

Appropriate

Yes

Dependent Variables

Mean shock intensity administered in an anonymous environment

Independent Variables

aggressive, nonaggressive
no model

Page

43

Behaviors

aggression

Quantitativeness

unscaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Aggressive models increase aggression

Reference Pfeiffer, et al (1979)

Appropriate

Yes

Dependent Variables

Aggression

Independent Variables

Depersonalization
Dispersion of responsibility

Page

42

Behaviors

Aggression

Quantitativeness

Nominal

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Both IVs increase aggression

Reference

Appropriate

Yes

Dependent Variables

Mean shock intensity administered

Independent Variables

Aggression aroused vs. nonaroused
Other aggressive stimuli

Page

43

Behaviors

Aggression

Quantitativeness

Unscaled graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Aggressive stimuli and prompting increase aggression

Reference	Pfeiffer (et al) 1979
Appropriate	Yes
Dependent Variables	Mean shock intensity administered
Independent Variables	Total trials Reinforcement vs. nonreinforcement
Page	44
Behaviors	Aggression
Quantitativeness	Unscaled graphs
Supporting Data	☐ No Validation ☐ Validation with Questionable Results ☐ Validated
Comments	Reinforcement for aggression increases aggressive behavior.

Reference Seigel, A.I. Human performance reliability - its measurement and impact on system reliability. Paper presented at the Design Engineering Conference and Show, Chicago, IL: ASME, 1978, Report No. 78-DE-17.

Appropriate Yes

Dependent Variables

Human reliability

Independent Variables

$$\text{Human reliability} = 1 - \frac{\text{No. failures}}{\text{Total attempts}}$$

Page 2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference Seigel, A.I. (1978)

Appropriate

Yes

Dependent Variables

Human availability

Independent Variables

$$\text{Human availability} = 1 - \frac{\text{Time lost or unmanned hours}}{\text{Total mission manhours}}$$

Page

2

Behaviors

"General"

Quantitativeness

Funcations

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference Seigel, A.I. (1978)

Appropriate Yes

Dependent Variables

Human MTTR

Independent Variables

Human MTTR = $\frac{\text{Total time of second try}}{\text{Number of second tries}}$

Page 2

Behaviors "General"

Quantitativeness

Funcations

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference Seigel, A.I. (1978)

Appropriate Yes

Dependent Variables

Equipment reliability

Independent Variables

$$\text{Equipment reliability} = 1 - \frac{\text{Number of failures during mission}}{\text{Number of iterations}}$$

Page 2

Behaviors "General"

Quantitativeness Functions

Supporting Data

- [ ] No Validation
- [X] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Seigel, A.I. (1978)

Appropriate

Yes

Dependent Variables

Equipment Availability

Independent Variables

$$\text{Equipment availability} = \frac{\text{Equipment up time}}{\text{Equipment up time + down time}}$$

Page

2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Seigel, A.I. (1978)

Appropriate

2

Dependent Variables

Equipment MTBF

Independent Variables

$$\text{Equipment MTBF} = \frac{\Sigma \text{ times between failures}}{\text{Number of failures}}$$

Page

2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference Seigel, A.I. (1978)

Appropriate

Yes

Dependent Variables

Equipment MTTR

Independent Variables

$$\text{Equipment MTTR} = \frac{\text{Total } \Sigma \text{ MTTR values over all iterations}}{\text{Number of iterations}}$$

Page 2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Seigel, A.I. (1978)

Appropriate

Yes

Dependent Variables

System reliability

Independent Variables

$$\text{System reliability} = 1 - \frac{\text{Number of equipment failures + Number of second try successes}}{\text{Number of iterations}}$$

Page

2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Seigel, A.I. (1973)

Appropriate

Yes

Dependent Variables

System availability

Independent Variables

$$\text{System availability} = 1 - \left(\frac{\text{Equipment down time X Unmanned hours}}{\text{Mission time}}\right)$$

Page

2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Seigel, A.I. (1978)

Appropriate

Yes

Dependent Variables

System MTTR

Independent Variables

$$\text{System MTTR} = \frac{\Sigma \text{ time for repairs} + \Sigma \text{ time for second try successes}}{\text{Number of repairs} + \text{Number of second try successes}}$$

Page

2

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, A.I. and Federman, P.J. Communications content training as an ingredient in effective team performance, Ergononic, 1973, 16(4), 403-416.

Appropriate

Medium

Dependent Variables

Independent Variables

Contains a taxonomy of categories for military radio communications in an anti-submarine warfare task

---

Page

413

Behaviors "General"

Quantitativeness

Nominal

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, A.I., Leahy, W.R. and Wolf, J.J. Human performance tradeoff curves for use in the design of Navy systems: Final Report. Washington, DC: Naval Sea Systems Command, April 1978, Contract No. N00024-76-C-6126 ADA053332.

Appropriate

Yes

Dependent Variables

Overall Performance

Independent Variables

Number of men at each rank and specialty
Body weight
S.D. of body weight
Summary and secondary specialty profeciency
Average work pace
Average stress tolerance threshold
Average caloric intake
Average aspiration interval
Average physical capability
Average hours since last sleep
Average duration in incapacity (sickness)
Minimum fatigue necessary for sleep
Average short term power output
Average capability after a full work day; equipment: failure rate, avg. repair time, S.D. of repair time, no men required to repair, mental load, consumable use
Event data

Page

4

Behaviors

Generic performance

Quantitativeness

Nominal

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Overall model for this simulation

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Independent Variables

System reliability metrics also spelled out in other Siegel papers

Page

8

Behaviors

"General"

Quantitativeness

Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Mean hours worked

Independent Variables

Primary competence (high - avg. - low)
Secondary competence
Men per shift
Shift length
Pace

Page

17-19

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Percentage of tasks performed successfully on the first attempt

---

Independent Variables

Crew size
Shift length
Primary competence
Secondary competence
Mental load
Pace
Level of aspiration
Leader's expectation

Page

20-22

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1976)

Appropriate

Yes

Dependent Variables

Unmanned station hours - hours of work not performed due to personnel unavilability

Independent Variables

Pace
Primary competence
Level of Aspiration
Leader's expectation
Shift length
Men per shift

Page

23-25

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Percentage of tasks ignored due to time, personnel, or stress constraints

Independent Variables

Crew size
Pace
Primary competence
Secondary competence
Shift length
Leader's expectation
Level of Aspiration
Mental load

Page

26-28

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1973)

Appropriate

Yes

Dependent Variables

System reliability and system availability

Independent Variables

Leader's expectation
Level of aspiration
Mental load
Men per shift
Shift length

Page

29-32

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

System MTTR

Independent Variables

Pace
Shift length
Mental load
Aspiration
Leader's expectation

Page

33-34

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Human reliability

Independent Variables

Primary competence
Pace
Shift length
Level of aspiration
Mental load
Leader's expectation

Page

35-36

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Human availability

Independent Variables

Pace
Men per shift
Shift length
Level of aspiration
Mental load
Leader's expectation

Page

37-38

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Human MTTR

Independent Variables

Pace
Mental load
Level of aspiration
Leader's expectation

Page

39-40

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

8

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Maximum Stress

Independent Variables

Men per shift
Pace

Page

41-42

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Hazard level

Independent Variables

Pace
Primary Competence
Shift length
Men per shift

Page

43-44

Behaviors

"General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al (1978)

Appropriate

Yes

Dependent Variables

Human availability and manpower utilization tradeoff

Independent Variables

Pace
Shift length

Page 45-46

Behaviors "General"

Quantitativeness

Scaled Graphs, Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., Federman, P.J., and Welsand, E.H. Perceptual/psychomotor requirements basic to performance in 35 Air Force specialties. Brooks AFB, TX: Air Force Human Resources Laboratory, Technical Report AFHRL-TR-80-26, Contract No. F33615-78-C-0032 ADA039391

Appropriate

Yes

Dependent Variables

Independent Variables

13 abilities:

- Control precision
- Manual dexterity
- Finger dexterity
- Multilimb coordination
- Rate control (tracking)
- Visual speed and accuracy
- Visual memory
- Position memory
- Auditory memory
- Clerical perception
- Perception of size and form,
- Depth perception

Page

26 and 29

Behaviors

"General"

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Munitions maintenance and fire protection specialities
Other also missile electronic equipment spec. & other support
Radio operator

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for ability in weapons mechanic

Independent Variables

Finger dexterity
Visual memory
Visual speed and accuracy
Position memory

Page

120

Behaviors

see IV

Quantitativeness

Nominal

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Judged by job incumbents AF

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for ability in LOM 25 missile mechanic

Independent Variables

Manual dexterity
Position memory

Page

120

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for ability in information specialist

Independent Variables

Visual memory
Clerical perception

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for ability in missile electronic equipment specialist

Independent Variables

Manual dexterity

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for abilities in missile facilities specialist

Independent Variables

Finger dexterity
Position memory

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for abilities in ground radio equipment repair

Independent Variables

Finger dexterity
Manual dexterity
Control precision
Visual memory
Visual speed and accuracy
Position memory

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for abilities in electrician

Independent Variables

Finger dexterity
Manual dexterity
Vicual memory

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I., et al

Appropriate

Yes

Dependent Variables

Judged need for ability in aircraft fuel system mechanic

Independent Variables

Finger dexterity
Manual dexterity
Position memory

Page

121

Behaviors

See IV

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I., Federman, P.J., and Sellman, W.A. A survey of student measurement and course evaluation procedures within the Air Training Command, Lowry AFB, CO: Air Force Human Resources Laboratory, Technical Report AFHRL-TR-74-5, July 1974, Contract No. F41609-71-C-0025 AD 786041.

Appropriate No

Dependent Variables

Independent Variables

Page

Behaviors

"General"

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Rouse, W.B. Problem solving performance of maintenance trainees in a fault diagnosis task. Human Factors, 1979, 21(2), 195-203.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

"General"

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Rouse, W.B. and Rouse, S.H. Measures of complexity of fault diagnosis -asks. IEEE Transactions on Systems, Man, and Cybernetics, 1979, SMC-9 (11), 720-727.

Appropriate

Yes

Dependent Variables

Complexity of a fault diagnosis task

Independent Variables

Information metric
Number of feasible solutions to find the symptomatic component
Probability of finding a failure when testing a given connection

---

Page

724

Behaviors

Seeing implications and consequences, logical evaluation, general reasoning

---

Quantitativeness

Nominal, Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Rouse, W.B. A model of human decision making in fault diagnosis tasks that include feedback and redundancy. IEEE Transactions on Systems, Man, and Cybernetics, 1979, SMC-9 (4), 237-241.

Appropriate

Dependent Variables

Number of tests to solution

Independent Variables

Psychological distance
Nature of the network under fault test: presence of feedback

Page

Behaviors

Flexibility, decision making, general reasoning, seeing implications and consequences

---

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Fuzzy set model
Feedback use is subject specific

Reference Rouse, W.B. A model of human decision making in a fault diagnosis task. IEEE Transactions on Systems, Man, and Cybernetics, 1978, SMC-8 (5), 357-361.

Appropriate

Yes

Dependent Variables

Number of tests to solution

Independent Variables

Psychological distance

Page

359

Behaviors Decision making, foresight

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Model presented

Reference Rouse, W.B. Problem solving performance in two semester maintenance trainees in two fault diagnosis tasks. Human Factors, 1979, 21 (5), 611-618.

Appropriate
No

Dependent Variables

Independent Variables

Page

Behaviors
General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference: Rouse, W.B. A model of the human in a cognitive prediction task. IEEE Transactions on Systems, Man, and Cybernetics, SMC-3 (5), 1973, 473-477.

Appropriate
Yes

Dependent Variables
System state
Location of a point

Independent Variables
System state at time N
Constants
System state at time N + 1
Last 10 points

Page

Behaviors: Visual memory, decision making, closure abilities, detection

Quantitativeness
Functions

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments
Slow time scale
Cognitive and thought factors only

Reference Baron, S. Kleinman, D.L., Miller, D.C., Levison, W.H., and Elking, J.I. Application of optimae control theory to the prediction of human performance in a complex task. Whight-Patterson Air Force Base, Ohio: Air Force Flight Dynamics Laboratory, AFFDL-TR-69-81, March 1970, F33615-68-C-1192.

Appropriate
No

Dependent Variables
Error
Error rate
Control input
} Observation-noise-ratios

Independent Variables
Control sensitivity
Input signal
Single axis iris full task perf.
Feveal iris peripheral vision

Page 61, 63-66, 74, 76, 78-82, 85, 95

Behaviors
Tracking

Quantitativeness
Table

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments Optimal scanning model (submodel of OCM) is described on pages 14-18. This is based on assumption that well-trained human operator behaves in an optimal manner subject to his inherent limitations and constraints.

Reference Baron, S. and Kleinman, D.L. The human as an optimal controller and information processor. IEEE Transactions on Man-Machine Systems, MMS-10 (1), 1969, 9-17.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I. et al

Reference

Levison, W.H., Baron, S., and Kleinman, D.L. A model for human controller remnant. IEEE Transactions on Man-Machine Systems, MMS-10, Dec. 1979, 101-108.

Appropriate

Yes

Dependent Variables

Normalized observation noise freq.
Normalized error freq.

Independent Variables

Mean-squared input
Vehicle dynamics (order of control)
Input bandwidth
Input-injection point

Page

106-107

Behaviors

Tracking

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

A model for human controller remnant

Reference Kleinman, DL., Baron, S., and Levison, W.H. An optimal control model of human response. Part I: Theory and Validation, Automatica, 1970, 6, 357-369.

Appropriate

No - older article

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Good model specifications

Reference

Baron, S., Kleinman, D.L. and [illegible] II. An optimal control model of human response. Part II: prediction of human performance in a complex task. Automatica, 1970, 6. 371-383.

Appropriate

No - older article

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Kleinman, DL.L., Baron, S., and Levison, W.H. A control theoretic approach to manned-vehicle systems analysis. IEEE Transactions on Automatic Controls, AC-16 (6), 1971, 824-832.

Appropriate

Yes

Dependent Variables

u(t) human control input to the system

Independent Variables

Vehilce dynamics
Display output
Observation noise
Time delay
Kalman estimator
Prediction
Motor noise

Page

827

Behaviors

Gross positioning, system equalization

Quantitativeness

Functions

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments

p 830 "Of course, the model does not tell us whether observed characteristics of the human's response ... are implemented by muscles in the arm or in the head."

Is this the problem of identifiability?

Reference

Baron, S. Application of the optimal control model for the human operator to reliability assessment. IEEE Transactions on Reliability, R-22 (3), 1973, 157-164.

Appropriate

Yes

Dependent Variables

Human performance reliability on a particular mission segment
$R_h(X)$ Probability that a given task will be correctly executed in some increment of time.
X (t) EX

Independent Variables

$$R_h(X) = P_r\{x(t) \text{ EX} | \text{stress}\} = \int_{x(t)\ \text{EX}} \text{pdf}\ (x(t)|\text{stress})\ dx$$

pdf = prob. dist. func.

Page

161

Behaviors

System equalization abilities

Quantitativeness

Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Investigate further

Reference Phatak, A., Weinert, H., Segall, I., and Day, C.N. Identification of a modified optimal control model for the human operator. Automatica, 1976, 12, 31-41.

Appropriate

Yes

Dependent Variables

Tracking

Independent Variables

Operator gain vector
Observation noise covariance

Page

35-38

Behaviors

Tracking

Quantitativeness

Functions

Supporting Data

- [ ] No Validation
- [X] Validation with Questionable Results
- [ ] Validated

Comments

Simplification of the optimal control model
The intent is to increase the identifiability of the model parameters it seems to have worked

Reference Kessler, K.M. and Phatak, A.V. Formulation and validation of a proportional-integral-derivative (P-I-D) structure modified optimal control model for the human gunner in an anti-aircraft artillery (AAA) tracking task. IEEE Conference on Decision and Control/15th Symposium on Adaptive Processes, 1976, 1099-1105.

Appropriate

Yes

Dependent Variables

Tracking error over time (elevation, azimuth)

Independent Variables

Process noise
Target trajectory

Page 1104

Behaviors

System equalization abilities (41-44), tracking

Quantitativeness

Functions

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Reference

Sarma, V.V.S. and Ramchord, K. Air Fleet and Facility Planning via optimal control models. IEEE Transactions on Systems, Man and Cybernetics, 1979, SMC-9 (3), 131-142.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Econometric Model

Reference Johannsen, G. and Govindaraj, T. Optimal control model predictions of system performance and attention allocation and their experimental validation in a display design study. IEEE Transactions on Systems, Man, and Cybernetics, SMC-10 (5), 1980

Appropriate No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

OCM

Reference Baron, S. and Levison, W.H. Display analysis with the optimal control model of the human operator. Human Factors, 1977, 19 (S), 437-457.

Appropriate

Yes

Dependent Variables

Displacement noise (tracking error)
Rate noise (tracking rate error)

Independent Variables

Observation noise
Attention allocation
Display variables

Page

441

Behaviors Time sharing; system equalization abilities, tracking

---

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Only aspects of the model relevant to display analysis

Reference Knoop, P.A. Survey of human operator modeling techniques for measurement applications. Wright-Patterson AFB, OH: Air Force Human Resources Laboratory, Technical Report AFHRL-TR-78-35, July 1978. ADA058327

Appropriate
Yes

Dependent Variables

Independent Variables
Review and Conclusions

Page
25

Behaviors

Quantitativeness
Nominal

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I., Pfeiffer, M.G., Kopstein, F., Wolf, J.J., and Ozkaptan, H. Human Performance in Continuous Operations: Volume III. Technical documentation ADA 088339.

Appropriate

Yes

Dependent Variables

Effectiveness profile by combat unit type and by performance factor

---

Independent Variables

Proficiency/speed factor by unit type
Enemy/friendly material strength ratio
Enemy/friendly personnel strength ratio
Enemy/friendly terrain advantage
Light level
Unit replacement conditions
Time of day that battle starts
Hours since last sleep at battlement
Platoon action and duration for each combat unit

Page

66

Behaviors

Functions

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I. et al

Appropriate Yes

Dependent Variables

Value of light level

Independent Variables

Light conditions (starlight to clear, sunny day)

Page 71

Behaviors General

Quantitativeness

Nominal

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I., et al

Appropriate Yes

Dependent Variables

Effectiveness of performance for a military unit

Independent Variables

Number of adverse impacts
Variability of impacts

Page 54

Behaviors

General

Quantitativeness

Functions

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Siegel, A.I. et al

Appropriate
Yes

Dependent Variables

Effectiveness of a unit

Independent Variables

Fatigue level
Light level
Stress level
Phase of diurnal rhythm

Regression model

Page
58

Behaviors

General

Quantitativeness
Functions

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference Siegel, A.I., Fischl, M.A., and MacPherson, D. The analytic profile system (APS) for evaluating visual displays. Human Factors, 1975, 17 (3), 278-288.

Appropriate No

Dependent Variables

Independent Variables

Page

Behaviors
General

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Siegel, A.I. and Bergman, B.A. A job learning approach to performance prediction. Personnel Psychology, 1975, 28, 325-339.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Miniature job training

Reference

Greening, C.P. Mathematical modeling of air-to-ground target acquisition. Human Factors, 1976, 18, 111-148.

Appropriate No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Involved air-to-ground rather than ground-to-air detection.
Presents 6 models of air-to-ground target acquisition
Author suggests that there is limited evidence of overall validation esp. with field data.

Reference

Greening, C.P. and Wyman, M.J. Experimental evaluation of a visual detection model. Human Factors, 1970, 12 (5), 435-455.

Appropriate

Yes

Dependent Variables

Cumulative probability of having recognized a target by a certain point in the approach

---

Independent Variables

Probability of looking at the target in one glimpse
Available resolution
Object/background contrast
Sky/ground luminance ratio
Aircraft altitude above target
Meteorological range (visibility)
Slant range
Threshold contrast ratio
Target height/width/depth
Horizontal range to target
Number of linear resolution elements to resolve the target

Page

Behaviors

General

Quantitativeness

Supporting Data

☐ No Validation

☒ Validation with Questionable Results

☐ Validated

Comments

Identification problem unsolved
Underlying neural mechanisms not considered
See also #34

Reference Greening, C.P. Target acquisition model evaluation: final summary report. China Lake, CA: Naval Weapons Center, Technical Report NWC TP 5536, 1973, Contract No. N00123-73-C-0250. AD 913918L

Appropriate

No

Dependent Variables

Air-to-ground target acquisition

Independent Variables

Models:		
	VISTRAC	FRANKLIN & WHITTENBURG
	DETECT II and III	CORNELL AERO LAB - RYLL
	MARSAM II	SHAPE
	GRC (A&B)	BOEING
	AUTONETICS	N.A. ROCKWELL - COLUMBUS
	SRI CRESS/SCREEN	VISTARAQ
	RAND	NVI
	FLAG	

Page

Behaviors

Detection, identification

Quantitativeness

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Air-to-ground
Much info. but should be more clearly presented in later Greening articles

Reference Greening, C.P. Modeling visual target acquisition: Inclusion of certain psychological variables. Chine Lake, CA: Naval Weapons Center Technical Report NWC TP 5690, August 1974, Contract No. N00123-74-C-0236. AD923266.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Air to ground target acquisition
Autonetics model

Reference

Bloomfield, J.R. Peripheral acuity with complex stimuli at two viewing distances. In NATO AGARD Conference Proceedings No. 100, AGARD CP-100, Air to ground target acquisition, Nov. 1972, Neuilly-Sur. Seine, France, AD755082

Appropriate

Yes

Dependent Variables

Minimum perceptible visual angle

Independent Variables

Eccentricity from fovea

Page

62

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Bloomfield, J.R. (1972)

Appropriate

Yes

Dependent Variables

Minimum perceptible target-background difference (minutes of angle)

Independent Variables

Eccentricity

Page

62

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference Bloomfield, J.R. (1972)

Appropriate
Yes

Dependent Variables

Variations in viewing distance have no effect on target acquisition or visual search

Independent Variables

Providing that:
a) the search display has angular dimensions of 9° or more;
b) viewing distance is greater than two feet; and
c) viewing distance is not so great as to preclude target resolution then →

Page B6-9

Behaviors
General

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments
As long as visual angle is constant, viewing distance doesn't matter.

Reference AGARD Air to ground target acquisition. AGARD Conference Proceedings No. 100, AGARD CP-100, Neuilly-Sur-Seine, France, November 1972. AD755082

Appropriate

Medium

Dependent Variables

Peripheral acuity (minimum perceptible visual angle)

Independent Variables

Eccentricity from favea

Page 62

Behaviors

Detection

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Jones, D.B., Freitag, M., and Collyer, S.C. Air-to-ground target acquisition source book: a review of the literature. Arlington, VA: Office of Naval Research, Code 455, September 1974, Contract No. N00014-72-C-0389. ADA015079.

Appropriate

yes

Dependent Variables

Independent Variables

Large amounts of data on:
- Properties of the visual system
- Target and environmental factors
- Atmospheric conditions
- Visibility parameters
- Visual search
- Models
- Imagery parameters

Page

Behaviors

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

This is an experimental psychology text on visual detection. 154 pages.

Reference

Jones, D.B., et al., 1974.

Appropriate

Dependent Variables

Log threshold contract
Search time

Independent Variables

Log target area
Contrast and size difference of targets and distractors

Page

Behaviors

Quantitativeness

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

An example of the contents of this document.

Reference

Gordon, J.J., Fean, C.R., and Church, P.V. Visual detection of low altitude aircraft (summer). Bureau of Ships; Technical Report No. NS-714-100, November 1961, Contract No. N00024-68-C-1100. AD512783.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Air-to-air detection, B-52 target, 500 ft.
Motion detection, rather than shape detection, will be the cue.
Recon. at 20,000, 5000, and 3000 ft.

Reference

Camden, R.S. Real-time air defense radar display: operator console simulation. Aberdeen Proving Ground, M.D.: U. S. Army Human Engineering Laboratory, Technical Memorandum 23-76, June 1976. ADA028217.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

An operator simulation of the SAM-D Air Defense System should exist. These are the programs to assess the man-machine interface.

Reference

Hilgendorf, R. and Milenski, J. SEEKVAL Project IA1: effects of target number and clutter on dynamic target acquisition. Wright-Patterson AFB, OH: AMRL, AMD, Technical Report AMRL-TR-74-4, January 1976, Contract No. F33615-73-C-4106. ADA024166

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Air to ground detection.

Reference

Woodward, D.P. and Nelson, P.D. A user oriented review of the literature on effects of sleeploss, work-rest schedules, and recovery on performance. Wash., DC: Office of Naval Research, Technical Report ACR-206, December 1974. ADA009778.

Appropriate

Yes

Dependent Variables

Independent Variables

Type of tank.

Page

1, 7-9

Behaviors

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Sleep loss decrement in performance.

Independent Variables

Work schedules most vulnerable to performance impairment by sleep loss.

Page

1, 1, 14-16

Behaviors

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Performance impairment

Independent Variables

Amount of sleep loss required to impair performance

Page

2

Behaviors

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Types of performance impairment most likely from sleep loss.

Independent Variables

Page

2, 21-22

Behaviors

Quantitativeness

Nominal

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Dependent Variables

Independent Variables

Procedures for reducing impairment risks in continuous operations.

Page

3

Behaviors

Quantitativeness

Nominal

Supporting Data

- [x] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Time required for recovery

Independent Variables

Unusual work schedules
Time required for recovery and adjustment

Page

3, 14-20

Behaviors

Quantitativeness

Nominal

Supporting Data

[X] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Recommended hours of recovery

Independent Variables

Cumulative hours of sleep loss

Page

4

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Woodward, D.P., et al., 1974.

Appropriate

Yes

Dependent Variables

Sleep loss decrement in performance

Independent Variables

Task duration

Page

9-13

Behaviors

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Wright, A.D. The performance of ground observers in detecting, recognizing, and estimating range to low-altitude aircraft. Fort Bliss, TX: Human Resources Research Office, Technical Report 66-19, December 1966, Contract No. DA-44-188-ARO-2. AD645537.

Appropriate

Yes

Dependent Variables

Range of incoming A/C when detected

Independent Variables

Aircraft class (jet, propeller)
Visual aiding (none, unaided detection with aided recognition, aided detection and recognition)
Observer offset (0,650,1400me.)

Page

9

Behaviors

detection

Quantitativeness

Table X

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when tentatively recognized
Range of incoming A/C when positively recognized

Independent Variables

Aircraft class (jet, propeller)
Visual aiding (none, unaided detection with aided recognition, aided detection and recognition)
Observer offset (0,650,1400 meters)

Page

9

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C
Probability of tentatively recognizing incoming A/C
Probability of positively recognizing incoming A/C

Independent Variables

Range (0 to 20,000 meters)

Page

12

Behaviors

detection
discrimination

Quantitativeness

Table 1
Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C

Independent Variables

Range (0 to 20,000 meters)
Type A/C (het A/C, propeller A/C, helicopter A/C)

Page

13

Behaviors

detection

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of tentatively recognizing incoming A/C
Probability of positively recognizing incoming A/C

Independent Variables

Range (0 to 20,000 meters)
Type A/C (jet A/C, propeller A/C, helicopters)

Page

13

Behaviors

discrimination

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C
Probability of tentatively recognizing incoming A/C
Probability of positively recognizing incoming A/C

Independent Variables

Range (0 to 20,000 meters)
Observer offset (0, 650, 1400 meters)

Page

15

Behaviors

detection, discrimination

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C
Probability of tentatively recognizing incoming A/C
Probability of positively recognizing incoming A/C

Independent Variables

Range (0 to 20,000 meters)
Type A/C (F-4C, F-100, T-33)

Page

16

Behaviors

detection discrimination

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C
Probability of tentatively recognizing incoming A/C
Probability of positively recognizing incoming A/C

Independent Variables

Range (0 to 20,000 meters)
Type rotary-wing A/C (4-1A, 4-6A, 0-1A, 0

Page

17

Behaviors

detection
discrimination

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Estimated range of incoming A/C when detected
Estimated range of incoming A/C when tentatively recognized
Estimated range of incoming A/C when positively recognized

Independent Variables

Aircraft class (jet, propeller, helicopter)

Page

20

Behaviors

detection, discrimination, spatial orientation

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Estimated range of incoming A/C when detected
Estimated range of incoming A/C when tentatively recognized
Estimated range of incoming A/C when positively recognized

Independent Variables

Visual aiding (none, binoculars for recog., binoculars for det. and rec.)
Observer offset (none, 650, 1400 meters)

Page

20

Behaviors

detection, discrimination, spatial orientation

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Percent range estimation error of incoming A/C when detected
Percent range estimation error of incoming A/C when tentatively recognized
Percent range estimation error of incoming A/C when positively recognized

---

Independent Variables

Type A/C (jet propeller, helicopter)

Page

21

Behaviors

detection, discrimination, spatial abilities

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wright (1966)

Appropriate

Yes

Dependent Variables

Percent range estimation error of incoming A/C when detected
Percent range estimation error of incoming A/C when tentatively recognized
Percent range estimation error of incoming A/C when positively recognized

---

Independent Variables

Visual aiding (none, binoculars for recog., binoculars for det. and recog.)
Observer offset (none, 650, 1400 meters)

Page

22

Behaviors

detection, discrimination, spatial abilities

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Baldwin, R.D. Relationship between recognition range and the size, aspect angle, and color of aircraft. Washington, D.C.: Human Resources Research Organization, Technical Report 73-2, February 1973, Contract No. DAHC 19-73-C-0004. AD758870.

Appropriate

Yes

Dependent Variables

Recognition range to model A/C

Independent Variables

Type A/C (AF-1E, F-84, F-100, F-102, Mirage, MIG-15, Futer, M.G-21, B-66, B-57, Flashlight, Frebar, Beagle)
Aspect angle (0°-0°, 15°-45°, 15°-20°, 10°-15°, 45°-35°, 0°-90°) [climb-healing]

Page

8

Behaviors

discrimination

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Used models supported on boom mounted on truck--not realistic since truck gave additional cues, etc.

Reference

Baldwin (1973)

Appropriate

Yes

Dependent Variables

Recognition range to model A/C

Independent Variables

Type A/C (AF-1E, F-100, F-102, Mirage, B-66, Beagle)
Color (grey, silver)

Page

9

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

See head page for this article.

Reference

Baldwin (1973)

Appropriate

Yes

Dependent Variables

Percent error in recognizing A/C models.

Independent Variables

Aspect angle [climb-heading] (0°-0°, 10°-15°, 15°-20°, 15°-45°, 45°-35°, 0°-90°)

Page

10

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Baldwin (1973)

Appropriate

Yes

Dependent Variables

A/C size at mean recognition range.

Independent Variables

Type A/C
Aspect angle

Page

11

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Used model A/C

Reference

Baldwin (1973)

Appropriate

Yes

Dependent Variables

Observer reliability for recognizing A/C

Independent Variables

A/C aspect angle

Page

14

Behaviors

human reliability

Quantitativeness

Table

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [x] Validated

Comments

Ised A/C models

Reference

Baldwin (1973)

Appropriate

Yes

Dependent Variables

Mean and SO of recognition ranges to model A/C

Independent Variables

Aspect angle
Type A/C

Page

21

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Baldwin, R.D., Frederickson, E.W., and Hackerson, E.C. Aircraft recognition performance of crew chiefs with and without forward observers. Washington, D.C.: Human Resources Research Organization, Technical Report 70-12, August 1970, Contract No. DAHC 19-70-C-0012. AD714-213.

Appropriate

Yes

Dependent Variables

Percent correct recognition of incoming A/C [real & models]

Independent Variables

Team (crew, single crew chief, single observer)
Range (0 to 20,000 meters)

Page

14

Behaviors

discrimination

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments

Reference

Baldwin, et al. (1970)

Appropriate

Yes

Dependent Variables

Remaining engagement time (time between crew chief's positive recognition judgment and A/C arrival at simulated weapon's crossover point)

Independent Variables

Crew vs. single observer
Observer offset (0, 90°, 2500m)

Page

15-16

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Baldwin, et al. (1970)

Appropriate

Yes

Dependent Variables

Communication score (sum of scale value of sequence codes during observations)

Independent Variables

Crew
Flight offset (0°, 90°, 2500m)

Page

19

Behaviors

Group communication

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Baldwin, et al. (1970)

Appropriate

Yes

Dependent Variables

Percentage recognition accuracy

Independent Variables

Crew
Observer (chief alone, chief in crew, forward observer)

Page

20

Behaviors

discrimination

Quantitativeness

Table

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments

Reference

Baldwin, R.D., Cliborn, R.C., and Foskett, R.J. The acquisition and retention of visual aircraft recognition skills. Arlington, VA: Army Research Institute, ARI Technical Report TR-76-A4, Contract No. DAH-19-75-C-0020.

Appropriate
No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Baldwin, R.D. Capabilities of ground observers to locate, recognize, and estimate distance of low-flying aircraft. Alexandria, VA: Human Resources Research Organization, Technical Report 73-8, March 1973, DAdC 19-73-C-0004. AD758873.

Appropriate

Yes

Dependent Variables

Probability of detection

Independent Variables

Range (0 to 20,000 meters)

Page

8-9

Behaviors

detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Teichner, W.H. and Mocharnuk, J.B. Visual search for complex targets. Human Factors, 1979, 21(3), 259-275.

Appropriate

High

Dependent Variables

Search time
Time per stimulus

Independent Variables

Half number of stimuli
1,2,3 dimensional search

Page

262, 266, 272

Behaviors

Perceptual speed

Quantitativeness

Scaled Graphs

Supporting Data

- [ ] No Validation
- [ ] Validation with Questionable Results
- [X] Validated

Comments

Support for Teichner's two-strategy approach.

Reference

Teichner, W.H., et al. (1979)

Appropriate

High

Dependent Variables

Total information

Independent Variables

Information in each dimension

Page

268

Behaviors

Perceptual speed

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Teichner, W.H., et al. (1979)

Appropriate

High

Dependent Variables

Time per stimulus

Independent Variables

Total stimulus information
Stimulus familiarity

Page

270, 272

Behaviors

Perceptual speed

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Teichner, W. H. and Krebs, M. J. Laws of visual choice reaction time. Psychological Review, 1974, 81(1), 75-98.

Appropriate

High

Dependent Variables

Choice RT

Independent Variables

Log_2 number of alternatives
Low practice
Light/digit stimuli
Key/voice responses

Page

81

Behaviors

Reaction time (45)

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Appropriate

High

Dependent Variables

Choice reaction time

Independent Variables

Stimulus probability
Varying S - R tasks
Critical signal probability

Page

87, 89

Behaviors

Reaction time

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Teichner, W.H., et al. (1974)

Appropriate

High

Dependent Variables

Choice reaction time

Independent Variables

Log_{10} number of trials (practice)
Digit-key/light-key

Page

82, 84

Behaviors

Reaction time (45)

Quantitativeness

Scaled Graphs
Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Teichner, W. H., et al. (1974)

Appropriate

High

Dependent Variables

Choice reaction time

Independent Variables

Log_2 number of alternatives
practice trials

Page

86

Behaviors

Reaction time (45)

Quantitativeness

Scaled Graphs

Supporting Data

[X] **No Validation**

[ ] **Validation with Questionable Results**

[ ] **Validated**

Comments

As practice increases, TR becomes constant as number of stimuli increases. In other words, Hick's Law and the single channel hypothesis vanish at high levels of practice.

Reference

Barber, P. Visual Search and Number of Stimuli Re-examined. Psychological Bulletin, 89 (1), 176 - 132

Appropriate

Slight

Dependent Variables

Search time

Time Per Element

Independent Variables

Half number of stimuli

Page

177 - 178

Behaviors

Reaction time

Quantitativeness

Scaled Graphs

Functions

Supporting Data

☐ No Validation

☒ Validation with Questionable Results

☐ Validated

Comments

Critique of Teichner and Krebs, 1976, #51

Reference

Paskin, H.M. A discrete stochastic optimal control model of the Human Operator. Atlanta, GA: American Society of Mechanical Engineers, 11th Joint Automatic Control Conference with the American Automatic Control Council, 1970, Session Paper 24-B, 640 - 641.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Paskin, H.M. A discrete stochastic optimal control model of the Human Operator. IEEE Proceedings of the National Aerospace Electronics Conference (NAECON '70 Record), 1970, 207 - 276.

Appropriateness Moderate

Dependent Variables

Normalized tracking error

Independent Variables

Input bandwidth
Conpensatory/Pursuit tracking

Page

Behaviors

Movement analysis

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Tracking - specific application of the optimal control model.

Reference

Siegel, A.I., Wolf, J.J., and Lautman, M.R. A family of models for measuring human reliability.

Appropriate

Moderate

Dependent Variables

Human reliability and availability
Equipment reliability and availability
System performance
Man-Machine system effectiveness

Independent Variables

Stress
Proficiency
Aspiration
Fatigue
Speed
Precision
Shifts
Motion Sickness

Page

112 - 114

Behaviors

Reliability Assessment

Quantitativeness

Nominal

Supporting Data

[XX] No Validation

[ ] Validation with Questionable Results

[ ] Validated

Comments

Reference

Bradford, W.H., A mathematical model for determining the probability of visual acquisition of ground targets by observers in low-level high-speed aircraft. Alberqurque, NM Sandia Laboratory Technical Report SC-TM-66-54, 1966.

Appropriate

Medium

Dependent Variables

CDF of acquisition range
Visual air-to-ground

Independent Variables

Slant range to target
Target characteristics (size, orientation, luminence, contrast)
Atmospheric visibility
Aircraft characteristics
Aircraft Altitude
Peripheral vision
Offset of aircraft track from target position
Observer scan patterns
Terrain Masking

Page

22, 30

Behaviors

Detection, Discrimination

Quantitativeness

Functions

Supporting Data

- [XX] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Franklin, M.E., and Whittenburg, J.A. Development of an air-to-ground detection/identification model. U.S. Army Human Engineering Laboratory Technical Report HSR-RR-65/4-D+, 1965. Contract No. DA-31-124-ARO-D-287.

Appropriate

Low

Dependent Variables

Traget detection/identification air-to-ground

Independent Variables

Target size
Target shape
Target/ground contrast
Clutter
Terrain type
Aircraft altitude
Aircraft speed
Range

Page

62

Behaviors

Dection
Discrimination

Quantitativeness

Functions

Supporting Data

- [xx] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Goldstein, I.L., and Durtman, P.W., Speed and load stress as determinents of performance in a time-sharing task. Human Factors, 1978, 20 (5), 603 - 609.

Appropriate

Potential

Dependent Variables

Performance on one (or several) choice RT tasks

Independent Variables

Speed of task stress
Number of simultaneous tasks (timesharing)

Page

606

Behaviors

Reaction time

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Rigney, J.W. and Towne, D.M. TASK TEACH: A method for computer-assisted performance training. Human Factors, 1970, 12 (3), 285 - 296.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

This article addresses task-analytic techniques rather than individual behavior modeling.

Reference

Briggs, G.E., and Johnston, W.A. Influence of a change in system criteria or team performance. Journal of Applied Psychology, 50 (6), 1966, 467 - 472.

Appropriate

Low

Dependent Variables

Team performance on a simulated radar controlling task

Independent Variables

Type of training (performance emphasis on speed or coordination)

Page

496 - 471

Behaviors

Group compatability
Group communication

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☒ Validation with Questionable Results

☐ Validated

Comments

This represents a rather obscure independent variable for this effort. Also, it would be difficult to generalize.

Reference

Johnston, W.A., and Briggs, G.E. Team Performance as a function of team arrangement and workload. Journal of Applied Psychology, 52 (2), 1968, 89 - 94.

Appropriate

High

Dependent Variables

Amount and type of communication among team members
Command and control performance

Independent Variables

Workload
Compensatory vs. non-compensatory team arrangement

Page

92

Behaviors

Movement Analysis
Movement prediction Rate Control

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wortman, D.B., Hixson, A.F., III, and Jorgensen, C.C., A SAINT Model of the AN/TSQ-73 Guided Missle Air Defense System.

Appropriate

Low (specific behavioral components are not individually modelled)

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Frederickson, C.W., Folletie, J.F., and Baldwin, R. Aircraft detection, range estimation, and auditory tracking tests in a desert environment. Fort Bliss, TX: Human Resources Research Office, Technical Report 67-3, March 1967, DA-44-188-ARO-2. AD 650403

Appropriate

Yes

Dependent Variables

Probability of detecting B-52 aircraft or F-4C or A-6

Independent Variables

Range (0 to 20 KM)
Modality (vision, audition)
Visual aiding (unaided, 6 X 30 binoculars)
" " (6 X 30, 7 X 50 binoculars)
Length of early warning (1, 5 minutes)
Degree of terrain masking
Offset from flight path (200, 1400, 2600, 3300M)

Page

10, 15

Behaviors

Detection

Quantitativeness

Tables
Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Frederickson et. al., 1967

Appropriate

Yes

Dependent Variables

Range estimates of fighter and bomber flights

Independent Variables

Flight line distance (1000, 1500, 2000, 2500, 3000, 3500, 4000 meters)
Observer offset from flight path (200, 1400, 2600, 3300 meters)

Page

18, 19

Behaviors

spatial orientation

Quantitativeness

Tables

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Frederickson, et. al., 1967

Appropriate

Yes

Dependent Variables

etection range for A/C structural components

Independent Variables

Observer offset (200, 1400 meters)
Type aircraft (F-4C, A-6, B-52, B-58)
Visual aiding (unaided, 6 X 30 binoculars
Structure

Page

22, 23

Behaviors

Detection
Discrimination

Quantitativeness

Tables

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Frederickson , et.al., 1967

Appropriate

Yes

Dependent Variables

Time delay between detection and identification of first structural component.

Independent Variables

Observer offset (200, 1400 meters)
Visual aiding (unaided, 6 X 30 binoculars)
Type aircraft (F-4C, A-6, B-52, B-58)

Page

25

Behaviors

Detection
Discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Frederickson, et. al., 1967

Appropriate

Yes

Dependent Variables

Order in which A/C structures were detected

Independent Variables

Observer offset (200, 1400 meters)
Type aircraft (F-4C, A-6, B-52, B-58)
Visiual aiding (unaided, 6 X 30 binoculars)
Structure

Page

24

Behaviors

Discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Frederickson, et. al., 1967

Appropriate

Yes

Dependent Variables

Auditory tracking error

Independent Variables

Slant range to A/C

Page

29, 30

Behavior: Auditory detection
Movement analysis

Quantitativeness
Table
Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun, W. Detection of random low-altitude jet aircraft by ground observers. Aberdeen Proving Ground, MD: Human Engineering Laboratory, Technical Memorandum 7-60, June 1960, DA Project S416-04-012. AD 238 341.

Appropriate

Yes

Dependent Variables

Probability of detecting incoming A/C

Independent Variables

Search sector size (1-360°, 2-180°, 4-90°, 8-45°)
Range (0 to 12,000 yds.)
Aircraft altitude (500, 1500 ft.)

Page

23 - 35

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Probability of identifying incoming A/C

Independent Variables

Search sector size (1 - 360°, 2 - 180°, 4 - 90°, 8 - 45°)
Range (0 to 12,000 yds.)
Aircraft altitude (500, 1500 ft.)

Page

37 - 47

Behaviors

Discrimination

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when detected

Independent Variables

Search sector size (1 - 360°, 2 - 180°, 4 - 90°, 8 - 45°)
Time (days of second week)

Page

50

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when detected

Independent Variables

Search sector size (1 - 360°, 2 - 180°, 4 - 90°, 8 - 45°)
Intertrial interval (0 - 15, 16 - 30, 31 - 45, 46+ minutes)

Page

51

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when detected

Independent Variables

Search sector size (1 - 360°, 2 - 180°, 4 - 90°, 8 - 45°)
Course of incoming A/C (1 thru 6)

Page

51

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when identified

Independent Variables

Search sector size (1 - 360°, 2 - 180°, 4 - 90°, 8 - 45°)
Course of incoming A/C (1 thru 6)

Page

52

Behaviors

Discrimination

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when detected

Independent Variables

Cross-over range (445 to 1335 yds.)
Course of incoming A/C (1 thru 6)
Site of observer

Page 53

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Range of incoming A/C when identified

Independent Variables

Cross-Over range (445 - 1335 yards)
Course of incoming A/C (1 thru 6)
Site of observer

Page

53

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wokoun (1960)

Appropriate

Yes

Dependent Variables

Accuracy of A/C identification

Independent Variables

Type of A/C (F-86, F-100, T-33)
Altitude (500, 1500 ft.)

Page

54

Behaviors

Discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Malloy, W.J., and Elliott, T.K. Measuring troubleshooting skills using hardware-free simulative. Lowry AFB, Co. Technical Report AFHRL-TR-78-47, December 1978, Contract No. F33615-77-C-0040. ADA064054

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Discussion of a personnel selection test using simulation.

Reference

Kubala, A.L. Problems in measuring team effectiveness. ARI Professional Paper 2 - 78. DAHC 19-75-C-0025, January 1978. ADA049560.

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Concerned with measuring team effectiveness

Reference

Goldstein, I.G., and Ringel, S. Survey of human factors problems in missle and communication systems. Army Project Research News, APRO-RM- 60-17, October 1960. Project No. SL95-60-001. ABD95127.

Appropriate

Low

Dependent Variables

Performance on various army systems

Independent Variables

Fatugue - stress - monotony
Team integration
Environmental conditions
Relationships with support services
Administrative considerations

Page

13 - 17

Behaviors

No specific behaviors

Quantitativeness

Nominal

Supporting Data

- [xx] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

General survey, no objective data collected. Not likely to provide information useful for quantitative modeling.

Reference

Collins, J.J. A study of potential contributions of small group behavi to team training technology development. Arlington, VA. ONR (code 45 Technical Report, Contract No. N0014-76-C-1076, 1977. ADA043911

Appropriate

NO

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Good discussion of team performance and training. No specific models o data presented.

Reference

Biederman, L.R., Gomer, F.E., Levine, S.H. Dynamic target acquisition: Empirical models of operator performance. Belling AFB, D.C. AFOSR Technical Report TR-80-1177, August 1980, Contract No. F49620-77-CC-0100. ADA092263.

Appropriate

Potentially

Dependent Variables

Response time and probability of target dection and recognition (air-to-g

---

Independent Variables

Target type
Target signature (including FLIR and TV signature)
Background scene complexity
Closure rate
Slant range

Page

78 - 82

Behaviors Detection
Discrimination

Quantitativeness

Functions

Supporting Data

☐ **No Validation**

☐ **Validation with Questionable Results**

☒ **Validated**

Comments

May be most useful for incorporating the effect of using IR recognition the model. Note that it is air-to-ground, however.

Reference

Muray, N., The use of information transmission as a nonparametric correlation in the analysis of complex behavior: a preliminary report. Office of Naval Research Technical Report, May 1980. Contract No. N004-77-C ADA087832.

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Provides an interesting tool for determining interrelationships among system, task, and human performance variable, not using linear correlation techniques. However, no models of human performance are presented, per se

Reference

Leahy, W.R., Lautman, M.R., Bearde, J.L., and Siegel, A.I., A digital simulation model of message handling in the Tactical Operations System: III further extensions of the model for increased interaction. USARI Technical Report, TR-77-A25, October 1977, Contract No. DAHC-19-72-C-0C03.

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

General discussion and flowcharts of MANMODEL. No specific human performa models discussed. Rather, the structure of MANMODEL which incorporates hu performance simulation is discussed.

Reference

Lichtenstein, S, and Fischoff, B. How well do probability experts assess probabilities? Arlington, VA: ONR Technical Report, PTR-1092-80-8, August 1980, Contract No. N00014-80-C-0150. ADA089619.

Appropriate

Medium

Dependent Variables

Percentage of correct responses

Independent Variables

Subjective estimate of percentage of correct responses (confidence)
Training level regarding amount of confidence in decision is warrented
Question difficulty

Page

8

Behaviors

Practical Judgement
Meaningful memory ability

Quantitativeness

Scaled Graphs

Supporting Data

☐ **No Validation**

☐ **Validation with Questionable Results**

☒ **Validated**

Comments

May be useful with respect to training people on how confident they should be for their visual IFFN identifications.

Reference

Kou, R.S., Glass, B.C., and Viknanis, M.M. Reduced-order observer model for AAA tracker response. Wright Patterson AFB, OH. Technical Report AMRL TR-79-79. August 1979. Contract No. F33615-79-C-0500. ADA080932

Appropriate

High

Dependent Variables

Tracking Error for AAA (azimuth and elevation)

Independent Variables

Approaching target angle rate
Approaching target angle acceleration
Observer and controller gains

Page

26 - 30

Behaviors

Tracking

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Compares model to OCM in terms of predictiveness, computer run time, and complexity. Looks very good.

Reference

Harris, R.N. A model of human decision making: preliminary research. Washington Navy Yard, D.C. : NPRDC, Technical Report WTR 73-5, August 1972. AD748596

Appropriate

Medium

Dependent Variables

Decision making efficiency (Boysian)

Independent Variables

Prior odds
Size of likelihood
Amount of Evidence
Data conflict
Response mode
Payoffs
Data sequence characteristics
Complexity of Decisions
Sample size
Amount of experience
Type of subjective estimates
Data Fidelity
Time stress
Non-independence in data
Prior uncertainty

Page

26

Behaviors

Practical judgement
Logical Evaluation
General reasoning

Quantitativeness

Nominal

Supporting Data

☒ XXX **No Validation**

☐ **Validation with Questionable Results**

☐ **Validated**

Comments

Reference

Finiley, D.L., and Muckler, F.A. Human factors research and the development of a manual systems application science: The systems sampling problem and solution. Naval Analysis Programs (code 431), ONR, Technical Report, July 1976, Contract No. N00014-74-C-0324. ADA029417

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

General discussion of taxonomies.

Reference

Baker, J.D. Quantitative modeling of human performance in information systems. Army Behavior and Systems Research Laboratory. Technical Research Note 232, Project No. 20062106A723, June 1972. Ft. Belvoir, VA. AD746096. Reprinted from Ergonomics 1970 13 (6), 645 - 654.

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

General discussion of a modeling approach, no specific performance models presented.

Reference

Wylie, C.D., Dicks, R.A., and Mackie, R.R. Toward a methodology for man-machine function allocation in the automation of surveillance systems Volume 1: summary. Arlington, VA: Office of Naval Research, Technical Report 1722 - F, Volume L, July 1975, Contract No. N00014-71-C-0301. AD-A017 103.

Appropriate

Yes

Dependent Variables

Variability in performance

Independent Variables

Innate abilities
Training
Operational experience
Motivation

Page 29

Behaviors

Quantitativeness

Nominal*

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Descrives model of human information processing in surveillance systems (p. 37 - 52)

*Taxonomy on pages 8 - 10

Reference

Williams, P.J. Kurtz, C.L., and Grubash, J.W. An evaluation of operator performance on the Patriot Display and Central Console. Human Engineering Lab, Aberdeen, MD, LIEL-TM-15-77, May 1977. ADB020664L

Appropriate

Low

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Specific test of design changes to Patriot Console. May be useful during validation because of data presented on operation.

Reference

Teper, G.L. An assessment of the "paper pilot" - An analytical approach to the specification and evaluation of flying qualities. Wright-Patterson AFB, OH: Air Force Flight Dynamics Laboratory, AFFDL-TR-71, 174, June 1972, Contract No. F33615-71-C-1071

Appropriate

No

Dependent Variables

Pilot rating of flying qualities

Independent Variables

System characteristics
Environmental variables, eg. gusts

Page

4

Behaviors

Quantitativeness

Scaled Graphs (6 - 8)
Nominal

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Oatman, L.C. Target detection using black-and-white television. Study 1: The effects of resolution degradation on target detection. Aberdeen Proving Ground, MD: Army Human Engineering Laboratory, Technical Memorandum 9-65, July 1965, Contract No. AD 62523

Appropriate

No (?)

Dependent Variables

Probability of detecting an M-48 tank
(correct detection scores, response time)

Independent Variables

TV resolution (800, 600, 400, 300 lines)

Page

5, 6, 9 - 12.

Behaviors

Detection

Quantitativeness

Scales Graphs
Tables

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Ground-to-ground oriented

Reference

Muralidharan, R., Baron, S., and Feehrer, C. A decision, monitoring, and control model of the human operator applied to an RPV control problem. Final Scientific Report. Bolling AFB, D.C. Air Force Office of Scientific Research, AFOSR-TR-79-0675, March 1969, Contract No. F44620-76-C0029. NTIS AD AO 69880.

Appropriate

?

Dependent Variables

Temporal monitoring behavior

Independent Variables

Strategy
Ground speed (ETA)
Lateral deviation error (LATDEV)
Disturbances
(see input parameters p. 52 for complete list)

Page

39

Behaviors

Quantitativeness

Nominal (pgs. 20, 50) Functions (pgs. 39, 41, 44, 47, 48)
Scaled Graphs (pgs. 61, 63, 65)

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Decision making structure of DEMON (decision-making, monitoring and control model) is based on expected net gain (ENG) (a criterion for national choice among alternatives).

Reference

Muralidharan, et. al., 1969

Appropriate

?

Dependent Variables

Monitoring looks

Independent Variables

Workload (number of controled RPV's)

Page

67, 69

Behaviors

Vigilance
Discrimination

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated by simulation

Comments

Reference

Miller, D.C., Feehrer, C.E., Muralidharan, R., Pew, R.W., and Baron, S. Development of human performance models for man-machine system simulatio Bolling AFB, D.C.: Air Force Office of Scientific Research, AFOSR-TR-79 October 1978, F44620-76-C-0029. AD A069879.

Appropriate

Yes (?)

Dependent Variables

Patching decision
Patch command computation and generation

Independent Variables

Disturbance
Presence or absence of patch control (correct deviation in flight path a

---

Page

See Quantitativeness

Behaviors

Decision-making

Quantitativeness

Nominal (pg. 41) Functions (pgs. 36-39, 52-60)

Supporting Data

☐ **No Validation**

☐ **Validation with Questionable Results**

☒ **Validated** (data not given)

Comments

2nd year report of 3 year effort

"modify optimal control model of human operator by incorporating structu notions that make it suitable for application to problems in which huma control actions are infrequent and in which monitoring and decision-mal are the operator's main activities" (Pg. 32)

Top-down
SAINT compatible (Appendix C has software modifications)

Reference

Hoffman, H.E. Preliminary results of trials to determine the maximum range of perception of fast aircraft. Aircraft Institute, German Aviation and Space Research Center, R.R.E. Translation No. 171, April 1967. N68 2

Appropriate

Yes

Dependent Variables

Maximum perception range

Independent Variables

Horizontal standard visibility
Unaided vs. 10 X 50 binoculars

Page 2 - 4 (unnumbered) unaided; 5 - 7 aided

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Siegel, A.S. and Wolf, J.J. Man-machine simulation models. New York: Wiley - Interscience, 1969.

Appropriate

Dependent Variables

Group performance

Subtask execution time (pg. 25)
Subtask success/failure (pg. 25 - 26) ---- one or two individuals

---

Independent Variables

Urgency (pg. 22)
Team cohesiveness (pg. 23)
Goal aspiration (pg. 26) ---------- one or two individuals
Stress

Individual orientation towards group goal achievement
Communications
Environment ---------- Group performance
Crewmember proficiency
Crew Morale

Page

See above

Behaviors

Across all behaviors

Quantitativeness

Functions

Supporting Data

☐ No Validation

☒ Validation with Questionable Results

☐ Validated

Comments

Describes Siegel Wolf model, looks to be of great value for including some group variables.

Reference

Warner, H.D., and Meimstra, N.W. Effects of noise intensity on visual target-detection performance. Human Factors, 1972, 14, 181 - 185.

Appropriate

Yes

Dependent Variables

Detection time
Detection error

Independent Variables

Noise level (0, 80, 90, 100 dB white noise) (headphones)
Difficulty (8, 16, 32 background letters in display)
Target location (central, peripheral regions of display)

Page 183 - 184

Behaviors

Detection

Quantitativeness

Tables
Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Hammill, H.B. The effect of magnification and profile scaling on the visual tracking system. Buffalo, NY: Cornell Aeronautical Laboratory, Perval Memo No. 3686, May 1972

Appropriate

Yes (?)

Dependent Variables

Tracking error (pg. 3)
Observer reaction time (pg. 3)

Independent Variables

Optical magnification
Aircraft/observer range vector

Page

3

Behaviors

Tracking
Movement prediction Rate control

Quantitativeness

Functions

Supporting Data

- [XX] **No Validation**
- [ ] **Validation with Questionable Results**
- [ ] **Validated**

Comments

Establishes a flight profile rescaling factor

Reference

Sugarman, R.C., Hammill, H.B., and Deutschman, J.N. A modified model for visual detection. Paper presented at the Fifth Naval Training Equipment Center and Industry Conference, February 1972.

Appropriate

Yes

Dependent Variables

Probability of detection

Independent Variables

$$\frac{\text{Target contrast}}{\text{Threshold contrast}} \quad \text{Ratio}$$

Page

2

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

- [XX] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Sugarman, et. al., 1972

Appropriate

Dependent Variables

Threshold difference

Independent Variables

Size of test object
Nasal vs. temporal field

Page

7

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, et. al., 1972

Appropriate

Yes

Dependent Variables

Average detection probability for single glimpse

Independent Variables

Peripheral angle within visual field
" " in the real world

Page

3, 8

Behaviors

Quantitativeness

Functions (pgs. 3, 8)

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, et. al., 1972

Appropriate

?

Dependent Variables

Peripheral angle distributions

Independent Variables

Target location

Page

10

Behaviors

Detection

Quantitativeness

Scaled Grams

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, et. al., 1972

Appropriate

Dependent Variables

Detection probability

Independent Variables

Visibility

Page

12 - 13

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Sugarman, et. al. 1972

Appropriate

Yes

Dependent Variables

Single glimpse detection probability

Independent Variables

Magnition
Difference angle between target position and fixation point

Page

14

Behaviors

Detection

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, R.C., Hammill, H.B., and Deutschman, J.N. Simplifying dynamic visual detection simulations. Paper presented at the Fifth Naval Training Equipment Center and Industry Conference, February 1972.

Appropriate

Yes

Dependent Variables

Probability of detection

Independent Variables

Target subtense angle (taget size)
Contrast
Structure

Page

10

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, et.al., 1972

Appropriate

Yes

Dependent Variables

Detection false alarm rate

Independent Variables

Days performed

Page

8

Behaviors

Detection

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Sugarman, et. al., 1972

Appropriate

Yes

Dependent Variables

Detection hit rate

Independent Variables

Days performed

Page

8

Behaviors

Detection

Quantitativeness

Table

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Hammill, H.B. Visual detection model for structured targets. Paper presented at the Twelfth Annual Pittsburgh Conference of Modeling and Simulation, April/May 1981.

Appropriate

Yes

Dependent Variables

Detection probability

Independent Variables

Threshold contrast
Target contrast

Page

1,2

Behaviors

Detection

Quantitativeness

Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Hammill, 1981

Appropriate

Dependent Variables

Detection time

Independent Variables

Target subtense
Contrast

Page

8

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, R.S., Kidd, J.S., and Ranc, M.P. A study of tactical decision making behavior. Hanscom Freed, MA: Decision Sciences Laboratory, ESD-TR-66-61, November 1965, Contract No. AF 19 (628) - 4792.

Appropriate

Yes

Dependent Variables

Tactical decision making

Independent Variables

Environment
Weapon system
Information system
Perceived environment
Desired environment
Action selection
Action evaluation

Page

8

Behaviors

Decision making

Quantitativeness

Nominal

Supporting Data

- [X] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Dependent Variables

Percent of selection

Independent Variables

Number of alternatives (2,4,6)

Page

53

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Dependent Variables

Error in estimation probability of success

Independent Variables

Number of alternatives (2,4,6)
Probability (low, medium, high)

Page

51

Behaviors

Probability estimation

Quantitativeness

Scaled graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Percent of correct choices

Independent Variables

Probability of success (high, medium, low)
Number of alternatives (2,4,6)

Page

46, 51

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Error in probability estimation

Independent Variables

True probability of success (low, medium, high)
Number of alternatives (2,4,6)

Page

46

Behaviors

Probability estimation

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Frequency of tactic selection

Independent Variables

Number of alternatives

Page

44

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Error in probability estimation

Independent Variables

Feedback (Ps, outcome)
Instructions (correct, incorrect)

Page

39

Behaviors

Probability estimation

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Percent of correct choices

Independent Variables

Block of problem (1 to 4)
Feedback (true Fs, outcome)
Instruction (correct, incorrect)

Page

36

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et al., 1965

Appropriate

Yes

Dependent Variables

Average error

Independent Variables

Correct vs. incorrect choices
Feedback (true Ps, 5 error, 10 error, outcome)

Page

32

Behaviors

Decision making

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Percent correct choices

Independent Variables

Magnitude of probability of success
Feedback (true Ps, 5 error, 10 error, outcome)

P

31

Behaviors

Decision making

Quanti[illegible]veness

Scaled Graphs

S[illegible]ing Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Co[illegible]ts

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Percent of correct choices

Independent Variables

Feedback condition (true, 5 error, 10 error, outcome)

Page

29

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Estimate time

Independent Variables

Blocks of problems

Page

23

Behaviors

Time estimation

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Select time

Independent Variables

Blocks of problems

Page

23

Behaviors

Decision time

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Error in estimating probability of success

Independent Variables

Type feedback (outcome feedback vs. correct answer feedback)

Page

21

Behaviors

Probability estimation

Quantitativeness

Scaled Graphs

Supporting Data

☐ Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Percent correct choices

Independent Variables

Warning vs. no warning

Page

22

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☒ Validated

Comments

Reference

Kinkade, et. al., 1965

Appropriate

Yes

Dependent Variables

Quality of tactical decision performance
(percent of correct choices)

Independent Variables

Frequency of change in system state (1 vs. 3)

Page

21

Behaviors

Decision making

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Lewis, D. and Low, W.F. Retention of skill on the SAM complex coordinato
Academy of Science, 1956, 63, 591 - 59[illegible].

Appropriate

Dependent Variables

Number of matches (red lights to adjacent green lights)

Independent Variables

Practice session
Distributed vs. massed practice

Page

594

Behaviors

Discrimination

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Lewis and Low, 1956

Appropriate

Dependent Variables

Variance of matches

Independent Variables

Practice session

Page

595

Behaviors

Discrimination

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Lewis and Low, 1956

Appropriate

Dependent Variables

Number of matches

Independent Variables

Practice session
Proficiency (high vs. low)

Page

596, 597

Behaviors

Discrimination

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Wainer, H. Estimating coefficients in linear models: it don't make no nevermind. Psychological Bulletin, 1976, 83, 213 - 217.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Presents Equal Weights Theorum: coefficients in multiple regression models can be replaced with equal weights with almost no loss in accuracy on the original data sample.

Reference

Akerman, A., III, and Kinzly, R.E. Predicting aircraft detectability. Human Factors, 1979, 21 (3), 277 - 291.

Appropriate

Yes

Dependent Variables

Contrast threshold

Independent Variables

Daylight only ------- Target size (α)
Retinal position (Θ)

Page

273

Behaviors

Detection

Quantitativeness

Nominal

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

VIDEM model - detectability of aircraft against sky background.

This paper reviews two components of VIDEM - Liminal contrast threshold and frequency-of-seeing curve.

Reference

Akerman, et. al., 1979

Appropriate

Yes

Dependent Variables

Single glimpse detection probability

Independent Variables

Apparent contrast/threshold contrast (C/Ct)

Page

280

Behaviors

Detection

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Koopman's empirical curve

Reference

Akerman, et. al., 1979

Appropriate

Yes

Dependent Variables

Liminal Brightness Contrast Threshold

Hammill/Sloan model

Independent Variables

$\ominus$ Target size

α Retinal position

Page

282

Behaviors

Detection

Quantitativeness

Supporting Data

☐ No Validation

☒ Validation with Questionable Results

☐ Validated

Comments

$\ominus$ is the angle between the foveal fixation axis and the observer-target axis.

Reference

Akerman, et. al., 197

Appropriate

Yes

Dependent Variables

Probability of detection

Independent Variables

Slant range

VIDEM Model

Page

289

Behaviors

Detection

Quantitativeness

Nominal Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

VIDEM uses Hammill/Sloan Ct
Soft shell search ($p(\Theta)$ for angles)
Discrete cumulation over 1/3 second glimpse intervals

Reference

Carter, R.C., and Cahill, M. Regression models of search time for color-coded information displays. Human Factors, 1979, 21 (3), 293 - 302.

Appropriate

Yes

Dependent Variables

Search time

Independent Variables

Number of coding colors
Display density (number of background stimuli)

Page 295

Behaviors

Identification

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Carter, et. al., 1979

Appropriate

Yes

Dependent Variables

Mean search time

Independent Variables

Number of items in each color category

Page

297

Behaviors

Identification

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Good linear fit

Reference

Burmeister, E. Uniqueness of rest points for optimal control models in economics. Automatica, 1978, 14, 157 - 160.

Appropriate

No

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Based on capita production functions economics model.

Reference

Siegel, A.I., Wolf, J.J. and Leahy, W.R. A digital simulation model of message handling in the Tatical Operations System: I. The model, its sensitivity, and user's manual. Alexandria, VA: U.S. Army Research Institute for the Behavioral and Social Sciences, Technical Report TR-77-A23, October 1977, Contract No. DAHC 19-72-C-0003. ADA047104

Appropriate

Yes

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- [XX] No Validation
- [ ] Validation with Questionable Results
- [ ] Validated

Comments

2, 20, and 99 person model versions.
Messages are to computer-based system via typewriter, keyboard, CPT

*If this model is used extensively, see Part II ADA046407 #90 for added improveme responsiveness, effectiveness, multiple input messages, undetected errors and st

Reference

Siegel, et. al., 1977

Appropriate

yes

Dependent Variables

Pace adjustment factor

Independent Variables

Aspiration - Performance difference. This represents an operationalizati of goal discrepancy.

Stress above or below threshold. The stress/discrepancy combinations are broken into four cases. P.29 has cases 1 & 3; in case 2, pace = 1; in ca pace = 1, stress = 90% stress.

Page

28 - 30

Behaviors

Aspiration

Quantitativeness

Nominal Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Pace adjustment factor

Independent Variables

Hours after starting work - up to 10 (12) hours
Over 9 days (fatigue)

Page

30 - 33

Behaviors

Quantitativeness

Scaled Graphs
Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Based on one study of air traffic controllers

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Pace adjustment factor

Independent Variables

Stress due to message backlog

Page

34

Behaviors

Quantitativeness

Unscaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Execution time

Independent Variables

Individual differences due to speed
Stress adjustment factor
Message length

Page

36

Behaviors

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Task element success probability change

Independent Variables

Operation precision

Page

40, 122 (function)

Behaviors

Quantitativeness

Scaled Graphs
Functions

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Task element success probability change

Independent Variablos

Stress

Page

41, 122 (Functions)

Behaviors

Quantitativeness

Unscaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Information loss

Independent Variables

Number of errors made (simulated)
Length of message
Importance of message

Page

42

Behaviors

Quantitativeness

Nominal

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Efficiency - 4 measures

Thoroughness	Responsiveness
Completeness	Accuracy

Independent Variables

Page

45 - 46

Behaviors

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

See these four variables

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Thoroughness

Independent Variables

$$= \frac{\text{Number of message blocks completed}}{\text{Number of possible blocks which could be completed}}$$

Page

45

Behaviors

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Completeness

Independent Variables

$$= E \text{ perf } (M) = \Sigma \frac{\text{Number of successes}}{\text{Number of successes and Number of fai}}$$

Page

45, 118

Behaviors

Quantitativeness

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Responsiveness

Independent Variables

$$= 1 - \left(\frac{\text{Average message processing time}}{\text{600 seconds}} \right)$$

Page

45

Behaviors

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Accuracy

Independent Variables

$$= 1 - \left(\frac{\text{Total information loss}}{\text{Number of messages completed}} \right)$$

Page

45

Behaviors

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Percentage of time worked

Independent Variables

Operator skill

Page

59

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Simulated results

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Performance time per message

Independent Variables

Team skill level
5 segments

Page

60 - 61

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Effectiveness
(4 components and overall efficiency)

Independent Variables

Operator skill

Page

62

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Simulated results

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Mean number of messages carried over to the next hour and completed within the hour

---

Independent Variables

Skill
Two job positions

Page

63 - 64

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Simulation data

Reference

Siegel, et. al., 1977

Appropriate

No

Dependent Variables

Error

Independent Variables

Skill
No relationship

Page

65

Behaviors

Quantitativeness

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Mean time used

Independent Variables

Segment
Operator mix

Page

69

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Dependent Variables

Percent of time worked

Independent Variables

Operator Mix - three officers/ three specs.
two officers/ four specs.

Skill level

Page

67

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Yes

Dependent Variables

Percent of time worked
Time

Independent Variables

Message workload

Page

71 - 73

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

Reference

Siegel, et. al., 1977

Appropriate

Medium

Dependent Variables

Performance adjustment

Independent Variables

Aspiration
Current performance level

Page

74 - 75

Behaviors

Quantitativeness

Scaled Graphs

Supporting Data

- ☐ No Validation
- ☐ Validation with Questionable Results
- ☐ Validated

Comments

Reference

Siegel, A.I., Wolf, J.J., and Liahy, W.R. A digital simulation model of message handling in the Tactical Operations System: II. Extensions of the model for interactivity with subjecst and emperimenters. Alexandria, V. U.S. Arim Research Institute for the Behavioral and Social Sciences, Technical Report TR-77-A24, October 1977, Contract No. DAHC19-72-C00003. ADA046407.

Appropriate

Conditional

Dependent Variables

Independent Variables

Page

Behaviors

Quantitativeness

Supporting Data

- [XX] **No Validation**
- [ ] **Validation with Questionable Results**
- [ ] **Validated**

Comments

If #89 is used, see this for important revisions to responsiveness, effectiveness, multiple input messages, undetected errors, and stress.

Reference

Hendrick, C., Mills, J., and Kiesler, C.A. Decision time as a function of the number and complexity of equally attractive alternatives. Journal of Personality and Social Psychology, 1968, 8, 313 - 318.

Appropriate

Yes

Dependent Variables

Decision time

Independent Variables

Choice set (2, 4)
Number of dimensions (1, many)
First vs. second choice

Page

316

Behaviors

Decision making

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Andrews, R.S. Jr., and Ringel, S. Certitude judgements and accuracy o information assimilation from visual displays. U.S. Army Personnel Research Office, Technical Research Note 145, May 1964

Appropriate

Yes

Dependent Variables

Certitude (certainty rating 1 to 7)
Accuracy of detection

Independent Variables

Number symbols removed (2, 4, 6, 8)
Amount of targets (12, 16, 20, 24)

Page 9

Behaviors Decision making
Discrimination
Visual memory

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

98

Reference

Andrews, et. al., 1964

Appropriate

?

Dependent Variables

Frequency of use of ratings of certitude

Independent Variables

Rating number

Page

20

Behaviors

Decision Making

Quantitativeness

Table

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Andrews, et. al., 1964

Appropriate

Yes

Dependent Variables

Certitude

Independent Variables

Number symbols removed (2, 4, 6, 8)

Page

13

Behaviors Decision Making
Discrimination
Visual Memory

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Andrews, et. al., 1964

Appropriate

Yes

Dependent Variables

Certitude

Independent Variables

Amount of targets (12, 16, 20, 24)

Page

12

Behaviors

Decision Making
Discrimination
Visual Memory

Quantitativeness

Scaled Graphs

Supporting Data

☐ No Validation

☐ Validation with Questionable Results

☒ Validated

Comments

Reference

Williams, E. Human performance simulation. Bolling AFB, Washington, D.C Air Force Office of Scientific Research, Technical Report AFOSR-TR-78-123 August 1977. Contract No. F44620-76-C-0013. ADA058940

Appropriate

Low

Dependent Variables

RT

Independent Variables

S-S Translation
S-R Translation
Response initiation

Page

Behaviors

Reaction time

Quantitativeness

Functions

Supporting Data

☒ No Validation

☐ Validation with Questionable Results

☐ Validated

Comments

This is part of Teichner's RT formulation.

APPENDIX L

MOPADS FINAL REPORT:

A DATA BASE FOR QUANTITATIVE HUMAN PERFORMANCE MODELS

TABLE OF CONTENTS

LIST OF TABLES

1-INTRODUCTION

To facilitate the development and analysis of a data base on models of human performance, a set of computer programs and a computerized data base were developed. Through the use of these programs, human performance models can easily be stored into the data base, and later analyzed in any number of different ways. This approach was believed to be more practical than hand-recording this data base. First, computer input of the data would be less time consuming than hand recording if a well designed input program was used. Second, each computer analysis requires only simple programming, whereas manual manipulation would require slower hand calculations and data restructuring Finally, additions or modifications to the data base would be less cumbersome using a computerized data base.

Currently, three program are available to create, modify, and analyze the data bases. They are currently on an APPLE II plus microcomputer with 48 K RAM. These programs and brief descriptions are as follows:

1. ENTER ARTICLES - This program allows the entry of models from articles in the open literature into the computer data base.
2. DATA BASE ANALYSIS - This program permits an examination of the data base to determine independent variables which are linked to skill categories, number or times the link is referenced, and articles referencing skill categories.
3. IV REF COUNT - This program counts the total number of times that all independent variables are referenced in the data base.

Sample on each of these programs will be provided in the following three sections of this report. Printouts of the three programs are presented in Appendices A, B, and C. Nine data files are required to support these programs. Six of them represent the independent variables, one of them represents the skills taxonomy, and one of them represents the literature data base.

The first five files include all independent variable titles, as noted below, and the sixth file (IND VARS-10) provides some information required by the programs for file manipulation.

IND VARS-1 - Aptitude

IND VARS-2 - Learning or practice

IND VARS-3 - Environment

IND VARS-4 - Task procedures and/or strategies

IND VARS-5 - Human/machine interface

IND VARS-10- This file includes the total number of titles in each of the five other files.

The five files with independent variable titles have been categorized in the above manner to provide an intuitive structure for the analyst. If the independent variables were stored in an arbitrary manner, alphabetically for example, an analyst might look for a variable such as "screen clutter" under any number of titles such as "number of targets per unit area" or "display clutter." The process of searching for the particular variable of interest would be difficult. Given the above structure, however, the analyst can reasonably assume that it is a "human/machine interface" variable and search for it within that file. The number of independent variable titles within each file are relatively small; 21, 12, 12, 33, and 42 for files IND VARS-1,2, 3,4, and 5, respectively. Printouts of the data in each of these files as of this report date are presented in Tables 1 through 5.

These files were structured to be dynamic. In other words, every time a model is encountered in which a new independent variable is presented, this new variable should be added to these data files. Consequently, as the

Table 1

INDEPENDENT VARIABLES- APTITUDE

RISK ACCEPTANCE
INTELLEGENCE
AGE
OPERATOR TYPE
SENSE OF DIRECTION
WILLINGNESS TO TAKE RISKS
OBSERVATION NOISE
PERCEPTUAL TIME DELAY
MOTOR NOISE
OPERATOR GAIN (8&9 COMBINED)
GENERAL PROFICIENCY FACTOR
THRESHOLD CONTRAST
ASPIRATION LEVEL
OPERATOR PRECISION
TEAM SKILL LEVEL
TRACKING ABILITIES
MOTIVATION LEVEL
VISUAL ACUITY
GENERAL APTITUDE SCORE
MORALE LEVEL
LEADERSHIP ABILITY

Table 2

INDEPENDENT VARIABLES-LEARNING/PRACTICE

PRACTICE TRIALS
TIME SINCE LAST PRACTICED
AMOUNT OF OVERLEARNING
TIME SINCE INITIALLY LEARNED
TRAINING EMPHASIS(SPEED VS COORDINATION)
AMOUNT OF TRAINING
PRESENCE OF FEEDBACK
NUMBER OF CORRECT CHOICES
MASSED VS DISTRIBUTED PRACTICE
AMOUNT OF INITIAL TRAINING
TYPE OF FEEDBACK
SUCCESSFUL TRIALS

Table 3

INDEPENDENT VARIABLES-ENVIRONMENTAL

AMBIENT TEMPERATURE
SKIN TEMPERATURE
TIME OF DAY
NOISE LEVEL
LIGHT LEVEL
DISTANCE BETWEEN COMMUNICATORS
TYPE OF NOISE
WET BULB TEMPERATURE
VIBRATION LEVEL
EFFECTIVE TEMPERATURE
GENERAL STRESS
FLASH INTENSITY

Table 4

INDEPENDENT VARIABLES-TASK STRATEGY

INFORMATION QUANTITY (SAMPLE SIZE)
TIME SPENT ON TASK
ORGANIZATION PERFORMING(SQUAD
DAYS SPENT ON TASK
WORK/REST CYCLE
SESSION NUMBER(SHORT SESSIONS)
NUMBER OF POSSIBLE RESPONSES
MONOCULAR VS BINOCULAR VIEWING
SLEEP DEPRIVATION
NUMBER OF TASKS PERFORMED AT ONCE
HOURS WORKED PER WEEK
IMPORTANCE OF TASK PARAMETERS
PHYSICAL ENERGY COST OF TASK
NUMBER OF DISPLAYS MONITORED
SIGNAL RATE
DRUG USAGE
TIME BETWEEN STIMULUS AND RESPONSE
PERCEIVED PROBABILITY OF AN EVENT
PERCEIVED COST OF ALTERNATE DECISIONS
PROBABILITY OF AN EVENT
MATERIAL STRENGTH RATIO-FRIEND/FOE
PERSONNEL STRENGTH RATIO FRIEND/FOE
ENEMY TERRAIN ADVANTAGE
UNIT RESUPPLY CONDITIONS
TEAM STRUCTURE
NUMBER OF OBSERVERS
OBSERVER SCAN PATTERNS
SPEED OF TASK STRESS
BACKLOG OF WORK
CONSISTENCY OF DATA
OBSERVATION TIME PER DISPLAY
HISTORY OF SYSTEM STATES
URGENCY OF TASK

Table 5

INDEPENDENT VARIABLES-TASK MAN/MACHINE INTERFACE

TARGET SIZE
TARGET BACKGROUND TYPE
BACKGROUND LUMINANCE
PERCENT TARGETS AMONG SIGNALS
TARGET BRIGHTNESS
DISTANCE TARGET HAS TRAVELED
RELATIVE DIRECTION OF TARGET MOTION
TARGET SPEED
TARGET TYPE
CONTRAST RATIO (TARGET/BACKGROUND)
TARGET ANGLE RELATIVE TO FOVEA
BLINDING FLASH ENERGY
NUMBER OF CUES FOR A SIGNAL
COMMUNICATION MODE
CONTROL SENSITIVITY
CONTROL DYNAMICS (ORDER)
TARGET MOVEMENT BANDWITH
PROCESS NOISE
SEARCH AREA
TARGET DISTANCE
METEOROLOGICAL RANGE(VISIBILITY)
TARGET HORIZONTAL DISTANCE
VISUAL AIDING
TARGET COLOR
STIMULUS MODE
RESPONSE MODE
LENGTH OF EARLY WARNING
DISPLAY TYPE
DISPLAY RESOLUTION
DISPLAY CLUTTER
NUMBER OF DISPLAY COLORS
MESSAGE LENGTH
MESSAGE IMPORTANCE
NUMBER OF SIGNAL/TARGET DIMENSIONS
SIGNAL DURATION
INTER DISPLAY ANGLE
DIRECTION OF ILLUMINATION SOURCE
SCENE COMPLEXITY
TARGET POSITION ON DISPLAY
TASK TYPE(PROCEDURAL
SIGNAL AMPLITUDE
STEP VS RAMP FAILURE

overall literature data base was being entered into the computer, new independent variables were constantly being added. This was accomplished through the ENTER ARTICLES program.

The SKILLS data file included the titles of all skills in the skill taxonomy. These skills represented the general nature of the dependent variables included in each model. This file was not intended to be dynamic since the skills taxonomy was believed to be comprehensive. There were some general skill titles added to include models with poorly defined dependent variables. Table 6 presents a printout of this file.

The final data file is (Skills by Independent Variables). This incl the information used to identify the models presented in the literature. Thi file is created and added onto by the ENTER ARTICLES program. Each record in the file includes the following information:

1. The number of the skill being modeled.
2. The number of the independent variable file (1 to 5).
3. The number of the independent variable within the file type identified above.
4. The degree of quantitative development of the model (ranging from 0 to 3, 0 = nominal, 3 = specific function identified).
5. The number of skill categories linked to one model.

The fifth piece of information above is relevant when the specific model presented in the literature corresponds to several skills. For example, if a article presented a model for detection and identification time, but did not separate the total time into a detection component and identification component, this model would be stored on the records, once with the skill identified as detection and once with the skill as identification. However, the fifth data point on each record would be a "2", indicating that this mod was stored in another location under a different skill title. This informati was included in the data base so models with multiple skills could be elimina from analysis if desired.

Table 6

PRINTOUT OF SKILLS DATA FILE

GENERAL BEHAVIORS
VIGILANCE
PROBABILITY ESTIMATION
TIME ESTIMATION
HUMAN RELIABILITY ASSESSMENT
MEANINGFUL MEMORY ABILITY
VERBAL KNOWLEDGE
WORD FLUENCY
NUMERICAL ABILITY
CONCEPT FLUENCY
DISCOVERY OF PRINCIPLES
FLEXIBILITY
SYMBOL MANIPULATION
DECISION MAKING
INTELLEGENCE
FINE MANIPULATIVE ABILITIES
POSITION ESTIMATION
GROSS MANIPULATIVE ABILITIES
CONTROL PRECISION
SPEED OF ARM MOVEMENT
MULTILIMB COORDINATION
POSITION REPRODUCTION
MOVEMENT ANALYSIS
MOVEMENT PREDICTION
ACCELERATION CONTROL
REACTION TIME
TRACKING
DETECTION
DISCRIMINATION ABILITIES
TIME SHARING
CLOSURE ABILITIES(PERCEPTUAL)
AUDITORY ABILITIES
SPATIAL ABILITIES-ORIENTATION
SPATIAL ABILITIES-VISUALIZATION
ASSOCIATE MEMORY-ROTE MEMORY
ASSOCIATE MEMORY-MEANINGFUL MEMORY
MEMORY SPAN-SHORT TERM MEMORY
MEMORY SPAN-INTEGRATION OF INFORMATION
VISUAL MEMORY
EXPLOSIVE STRENGTH-GENERAL
EXPLOSIVE STRENGTH-LEG EMPHASIS
EXPLOSIVE STRENGTH-UPPER BODY EMPHASIS
STATIC STRENGTH-ARM/HAND/SHOULDER
STATIC STRENGTH-LEG/TRUNK
DYNAMIC STRENGTH-ARMS

Table 6 (continued)

DYNAMIC STRENGTH-LEGS
TRUNK STRENGTH
EXTENT FLEXIBILITY
DYNAMIC FLEXIBILITY
GROSS BODY EQUILIBRIUM
BALANCE-VISUAL CUES
GROUP COOPERATION
GROUP COMMUNICATION
LEADERSHIP
GENERAL COGNITIVE ABILITIES
GENERAL PHYSICAL ABILITIES

In the following three sections, examples are presented of interactio with the three sets of computer programs, ENTER ARTICLES, DATA BASE ANALYSIS, and IV REF COUNT. In these sections, all input by the analyst are underlined. All other information is printed by the computer.

2-EXAMPLE OF ENTERING ARTICLES

Below is an example of an interaction between the analyst and the computer program ENTER ARTICLES. The example illustrates an analyst attempting to add a model from article number 100, skill number 14 (Decision Making), and an initial attempt to link this model to the independent variable Visual Acuity. After this first attempt, the analyst decides to create a new independent variable, then again changes his mind and decides not to add the model at all.

NOTE: All information entered by the analyst is underlined.

```
]RUN ENTER ARTICLES


DO YOU WANT TO ADD A NEW ARTICLE TO
THE DATA BASE? (Y/N) Y
ENTER ARTICLE NUMBER-100
ENTER QUANTITATIVENESS INDEX (0-3)-2
DO YOU NEED A SKILL CATEGORY MENU? (Y/N) Y
1 GENERAL BEHAVIORS
2 VIGILANCE
3 PROBABILITY ESTIMATION
4 TIME ESTIMATION
5 HUMAN RELIABILITY ASSESSMENT
6 MEANINGFUL MEMORY ABILITY
7 VERBAL KNOWLEDGE
8 WORD FLUENCY
9 NUMERICAL ABILITY
10 CONCEPT FLUENCY
11 DISCOVERY OF PRINCIPLES
12 FLEXIBILITY
13 SYMBOL MANIPULATION
14 DECISION MAKING
15 INTELLEGENCE
16 FINE MANIPULATIVE ABILITIES
17 POSITION ESTIMATION
18 GROSS MANIPULATIVE ABILITIES

HIT 'SPACE' KEY TO CONTINUE
 ANY OTHER KEY TO EXIT MENU
```

(SPACE KEY)

19 CONTROL PRECISION
20 SPEED OF ARM MOVEMENT
21 MULTILIMB COORDINATION
22 POSITION REPRODUCTION
23 MOVEMENT ANALYSIS
24 MOVEMENT PREDICTION
25 ACCELERATION CONTROL
26 REACTION TIME
27 TRACKING
28 DETECTION
29 DISCRIMINATION ABILITIES
30 TIME SHARING
31 CLOSURE ABILITIES(PERCEPTUAL)
32 AUDITORY ABILITIES
33 SPATIAL ABILITIES-ORIENTATION
34 SPATIAL ABILITIES-VISUALIZATION
35 ASSOCIATE MEMORY-ROTE MEMORY
36 ASSOCIATE MEMORY-MEANINGFUL MEMORY

HIT 'SPACE' KEY TO CONTINUE
ANY OTHER KEY TO EXIT MENU

(SPACE KEY)

```
37 MEMORY SPAN-SHORT TERM MEMORY
38 MEMORY SPAN-INTEGRATION OF INFORMATI
39 VISUAL MEMORY
40 EXPLOSIVE STRENGTH-GENERAL
41 EXPLOSIVE STRENGTH-LEG EMPHASIS
42 EXPLOSIVE STRENGTH-UPPER BODY EMPHAS
43 STATIC STRENGTH-ARM/HAND/SHOULDER
44 STATIC STRENGTH-LEG/TRUNK
45 DYNAMIC STRENGTH-ARMS
46 DYNAMIC STRENGTH-LEGS
47 TRUNK STRENGTH
48 EXTENT FLEXIBILITY
49 DYNAMIC FLEXIBILITY
50 GROSS BODY EQUILIBRIUM
51 BALANCE-VISUAL CUES
52 GROUP COOPERATION
53 GROUP COMMUNICATION
54 LEADERSHIP

HIT 'SPACE' KEY TO CONTINUE
 ANY OTHER KEY TO EXIT MENU
          (SPACE KEY)
55 GENERAL COGNITIVE ABILITIES
56 GENERAL PHYSICAL ABILITIES

ALL SKILL CATEGORIES DISPLAYED
ENTER SKILL CATEGORY NUMBER OR
HIT RETURN IF ALL SKILL CATEGORIES
HAVE BEEN ENTERED- 14
ENTER NEXT SKILL CATEGORY
DO YOU NEED A SKILL CATEGORY MENU? (Y/N) N
ENTER SKILL CATEGORY NUMBER OR
HIT RETURN IF ALL SKILL CATEGORIES
HAVE BEEN ENTERED-    (RETURN KEY)
WHAT TYPE OF VARIABLE IS INDEPENDANT
VARIABLE NUMBER 1

 APTITUDE=1
 LEARNING/PRACTICE=2
 ENVIRONMENTAL=3
 TASK-PROCEDURE/STRATEGY=4
 TASK-MAN/MACHINE INTERFACE=5

HIT 'RETURN' IF ALL VARIABLES ARE IN
ENTER '9' TO SKIP THIS ARTICLE   1
```

```
1    RISK ACCEPTANCE
2    INTELLEGENCE
3    AGE
4    OPERATOR TYPE
5    SENSE OF DIRECTION
6    WILLINGNESS TO TAKE RISKS
7    OBSERVATION NOISE
8    PERCEPTUAL TIME DELAY
9    MOTOR NOISE
10   OPERATOR GAIN (8&9 COMBINED)
11   GENERAL PROFICIENCY FACTOR
12   THRESHOLD CONTRAST
13   ASPIRATION LEVEL
14   OPERATOR PRECISION
15   TEAM SKILL LEVEL
16   TRACKING ABILITIES
17   MOTIVATION LEVEL
18   VISUAL ACUITY
HIT RETURN FOR MORE
ENTER VARIABLE NUMBER IF FOUND
ENTER 'A' IF NEW VARIABLE IS REQUIRED
ENTER 'M' TO  RETURN TO VAR TYPE MENU
ENTER 'R' IF MENU MUST BE REVIEWED
?          16

SKILL=14 IND VAR=18 ARTICLE=100
ALL INFORMATION CORRECT? (Y/N)  N
ENTER VARIABLE NUMBER IF FOUND
ENTER 'A' IF NEW VARIABLE IS REQUIRED
ENTER 'M' TO  RETURN TO VAR TYPE MENU
ENTER 'R' IF MENU MUST BE REVIEWED   M

WHAT TYPE OF VARIABLE IS INDEPENDANT
VARIABLE NUMBER 1

 APTITUDE=1
 LEARNING/PRACTICE=2
 ENVIRONMENTAL=3
 TASK-PROCEDURE/STRATEGY=4
 TASK-MAN/MACHINE INTERFACE=5

HIT 'RETURN' IF ALL VARIABLES ARE IN
ENTER '9' TO SKIP THIS ARTICLE    3
```

```
1    AMBIENT TEMPERATURE
2    SKIN TEMPERATURE
3    TIME OF DAY
4    NOISE LEVEL
5    LIGHT LEVEL
6    DISTANCE BETWEEN COMMUNICATORS
7    TYPE OF NOISE
8    WET BULB TEMPERATURE
9    VIBRATION LEVEL
10   EFFECTIVE TEMPERATURE
11   GENERAL STRESS
12   FLASH INTENSITY

ALL VARIABLES DISPLAYED

ENTER VARIABLE NUMBER IF FOUND
ENTER 'A' IF NEW VARIABLE IS REQUIRED
ENTER 'M' TO  RETURN TO VAR TYPE MENU
ENTER 'R' IF MENU MUST BE REVIEWED  A

ENTER NAME OF NEW INDEPENDANT VARIABLE
ENTER 'SPACE' TO RETURN TO MENU
          (SPACE BAR)
1    AMBIENT TEMPERATURE
2    SKIN TEMPERATURE
3    TIME OF DAY
4    NOISE LEVEL
5    LIGHT LEVEL
6    DISTANCE BETWEEN COMMUNICATORS
7    TYPE OF NOISE
8    WET BULB TEMPERATURE
9    VIBRATION LEVEL
10   EFFECTIVE TEMPERATURE
11   GENERAL STRESS
12   FLASH INTENSITY

ALL VARIABLES DISPLAYED

ENTER VARIABLE NUMBER IF FOUND
ENTER 'A' IF NEW VARIABLE IS REQUIRED
ENTER 'M' TO  RETURN TO VAR TYPE MENU
ENTER 'R' IF MENU MUST BE REVIEWED  M
```

```
WHAT TYPE OF VARIABLE IS INDEPENDANT
'/ARIABLE NUMBER 1

 APTITUDE=1
 LEARNING/PR\CTICE=2
 ENVIRONMENTAL=3
 TASK-PROCEDURE/STRATEGY=4
 TASK-MAN/MACHINE INTERFACE=5

HIT 'RETURN' IF ALL VARIABLES ARE IN
ENTER '9' TO SKIP THIS ARTICLE (RETURN KEY)
DO YOU WANT TO ADD A NEW ARTICLE TO
THE DA' A BASE? (Y/N)       N
DO YOU  JANT TO QUIT? (Y/N)Y

]
```

NOTE: In this example, no articles, models, or new independent variables were added to the data base. These additions were avoided by respondidg "NO" to the question "IS THIS CORRECT?". New data could have been added simply by responding "YES" to this question.

3-EXAMPLE OF DATA BASE ANALYSIS

Below is an example of the interaction between the analyst and the DATA BASE ANALYSIS computer program. This example shows the analyst looking at the combination of skills DETECTION and DISCRIMINAT Additionally, he only wants to review those models which were tied exclusively (i.e., unambiguously) to either detection or discriminatio

```
]RUN DATA BASE ANALYSIS,D1



DO YOU NEED A SKILL CATEGORY MENU? (Y/N)  Y
1 GENERAL BEHAVIORS
2 VIGILANCE
3 PROBABILITY ESTIMATION
4 TIME ESTIMATION
5 HUMAN RELIABILITY ASSESSMENT
6 MEANINGFUL MEMORY ABILITY
7 VERBAL KNOWLEDGE
8 WORD FLUENCY
9 NUMERICAL ABILITY
10 CONCEPT FLUENCY
11 DISCOVERY OF PRINCIPLES
12 FLEXIBILITY
13 SYMBOL MANIPULATION
14 DECISION MAKING
15 INTELLEGENCE
16 FINE MANIPULATIVE ABILITIES
17 POSITION ESTIMATION
18 GROSS MANIPULATIVE ABILITIES

HIT 'SPACE' KEY TO CONTINUE
 ANY OTHER KEY TO EXIT MENU
```

(SPACE KEY)

19 CONTROL PRECISION
20 SPEED OF ARM MOVEMENT
21 MULTILIMB COORDINATION
22 POSITION REPRODUCTION
23 MOVEMENT ANALYSIS
24 MOVEMENT PREDICTION
25 ACCELERATION CONTROL
26 REACTION TIME
27 TRACKING
28 DETECTION
29 DISCRIMINATION ABILITIES
30 TIME SHARING
31 CLOSURE ABILITIES(PERCEPTUAL)
32 AUDITORY ABILITIES
33 SPATIAL ABILITIES-ORIENTATION
34 SPATIAL ABILITIES-VISUALIZATION
35 ASSOCIATE MEMORY-ROTE MEMORY
36 ASSOCIATE MEMORY-MEANINGFUL MEMORY

HIT 'SPACE' KEY TO CONTINUE
ANY OTHER KEY TO EXIT MENU

(SPACE KEY)

37 MEMORY SPAN-SHORT TERM MEMORY
38 MEMORY SPAN-INTEGRATION OF INFORMATI
39 VISUAL MEMORY
40 EXPLOSIVE STRENGTH-GENERAL
41 EXPLOSIVE STRENGTH-LEG EMPHASIS
42 EXPLOSIVE STRENGTH-UPPER BODY EMPHAS
43 STATIC STRENGTH-ARM/HAND/SHOULDER
44 STATIC STRENGTH-LEG/TRUNK
45 DYNAMIC STRENGTH-ARMS
46 DYNAMIC STRENGTH-LEGS
47 TRUNK STRENGTH
48 EXTENT FLEXIBILITY
49 DYNAMIC FLEXIBILITY
50 GROSS BODY EQUILIBRIUM
51 BALANCE-VISUAL CUES
52 GROUP COOPERATION
53 GROUP COMMUNICATION
54 LEADERSHIP

HIT 'SPACE' KEY TO CONTINUE
ANY OTHER KEY TO EXIT MENU

(SPACE KEY)

```
55 GENERAL COGNITIVE ABILITIES
56 GENERAL PHYSICAL ABILITIES

ALL SKILL CATEGORIES DISPLAYED

ENTER SKILL CATEGORY NUMBERS
TO BE REVIEWED
TYPE 'END'WHEN LIST IS COMPLETE
? 28
?29
?END
ARE YOU INTERESTED ONLY IN
EXCLUSIVE ARTICLES? (Y/N)Y


ARE YOU INTERESTED IN THE ARTICLES
REFERENCING THESE SKILL CATEGORIES? Y
ARTICLE NUMBER      #TIMES REFERENCED
--------------      -----------------

1                    17
5                    3
38                   1
45                   8
46                   3
47                   5
49                   1
63                   6
64                   15
79                   1
82                   2
84                   3
86                   5
87                   5
88                   4
HIT ANY KEY TO CONTINUE
```

(SPACE KEY)

ARTICLE NUMBER	#TIMES REFERENCED
90	3
92	6
93	3
101	1
113	2
116	4
117	3
121	1
124	5
133	6
142	4
143	3
147	2
148	1
149	6

HIT ANY KEY TO CONTINUE

(SPACE KEY)

ARTICLE NUMBER	#TIMES REFERENCED
152	2
157	2
161	9
182	1
189	2
190	3
196	10

TOTAL NUMBER OF ARTICLES=37
TOTAL NUMBER OF REFERENCES=158

ARE YOU INTERESTED IN THE INDEPENDANT VARIABLES WHICH WERE REFERENCED? Y

INDEPENDENT VARIABLE NAME	TIMES
OPERATOR TYPE	1
GENERAL PROFICIENCY FACTOR	1
THRESHOLD CONTRAST	3
VISUAL ACUITY	2
GENERAL APTITUDE SCORE	1
# PRACTICE TRIALS	2
AMOUNT OF TRAINING	1
PRESENCE OF FEEDBACK	1
MASSED VS DISTRIBUTED PRACTICE	1
TYPE OF FEEDBACK	1
TIME OF DAY	2
NOISE LEVEL	3
LIGHT LEVEL	1
FLASH INTENSITY	1
TIME SPENT ON TASK	2

HIT ANY KEY TO CONTINUE
(SPACE KEY)

INDEPENDENT VARIABLE NAME	TIMES
DAYS SPENT ON TASK	2
SESSION NUMBER(SHORT SESSIONS)	1
NUMBER OF POSSIBLE RESPONSES	1
MONOCULAR VS BINOCULAR VIEWING	1
SLEEP DEPRIVATION	2
NUMBER OF DISPLAYS MONITORED	2
SIGNAL RATE	4
PERCEIVED PROBABILITY OF AN EVENT	2
TEAM STRUCTURE	1
NUMBER OF OBSERVERS	1
TARGET SIZE	7
TARGET BACKGROUND TYPE	5
BACKGROUND LUMINANCE	2
PERCENT TARGETS AMONG SIGNALS	2
TARGET BRIGHTNESS	3

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDENT VARIABLE NAME	TIMES
DISTANCE TARGET HAS TRAVELED	1
RELATIVE DIRECTION OF TARGET MOTION	12
TARGET SPEED	2
TARGET TYPE	12
CONTRAST RATIO (TARGET/BACKGROUND)	8
TARGET ANGLE RELATIVE TO FOVEA	7
NUMBER OF CUES FOR A SIGNAL	2
SEARCH AREA	7
TARGET DISTANCE	16
METEOROLOGICAL RANGE(VISIBILITY)	3
TARGET HORIZONTAL DISTANCE	4
VISUAL AIDING	4
TARGET COLOR	4
STIMULUS MODE	2
LENGTH OF EARLY WARNING	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDENT VARIABLE NAME	TIMES
DISPLAY RESOLUTION	2
DISPLAY CLUTTER	6
NUMBER OF DISPLAY COLORS	1
SIGNAL DURATION	1
DIRECTION OF ILLUMINATION SOURCE	1
SCENE COMPLEXITY	1
TARGET POSITION ON DISPLAY	1
STEP VS RAMP FAILURE	1

DO YOU WISH TO REVIEW THE RESULTS? N

]

4-Example of IV Ref Count

This is an example of the execution of the program IV REF COUNT. It prints the total number of times each Independent Variable is referenced in the data base.

```
]RUN IV REF COUNT
```

INDEPENDANT VARIABLE NAME	#TIMES
RISK ACCEPTANCE	0
INTELLEGENCE	2
AGE	0
OPERATOR TYPE	4
SENSE OF DIRECTION	2
WILLINGNESS TO TAKE RISKS	1
OBSERVATION NOISE	4
PERCEPTUAL TIME DELAY	3
MOTOR NOISE	3
OPERATOR GAIN (8&9 COMBINED)	2
GENERAL PROFICIENCY FACTOR	10
THRESHOLD CONTRAST	3
ASPIRATION LEVEL	6
OPERATOR PRECISION	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
TEAM SKILL LEVEL	1
TRACKING ABILITIES	1
MOTIVATION LEVEL	1
VISUAL ACUITY	2
GENERAL APTITUDE SCORE	1
MORALE LEVEL	1
LEADERSHIP ABILITY	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
# PRACTICE TRIALS	15
TIME SINCE LAST PRACTICED	5
AMOUNT OF OVERLEARNING	1
TIME SINCE INITIALLY LEARNED	3
TRAINING EMPHASIS(SPEED VS COORDINAT	2
AMOUNT OF TRAINING	6
PRESENCE OF FEEDBACK	3
NUMBER OF CORRECT CHOICES	1
MASSED VS DISTRIBUTED PRACTICE	1
AMOUNT OF INITIAL TRAINING	2
TYPE OF FEEDBACK	1
# SUCCESSFUL TRIALS	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
AMBIENT TEMPERATURE	2
SKIN TEMPERATURE	3
TIME OF DAY	10
NOISE LEVEL	9
LIGHT LEVEL	5
DISTANCE BETWEEN COMMUNICATORS	3
TYPE OF NOISE	2
WET BULB TEMPERATURE	1
VIBRATION LEVEL	1
EFFECTIVE TEMPERATURE	2
GENERAL STRESS	6
FLASH INTENSITY	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
INFORMATION QUANTITY (SAMPLE SIZE)	1
TIME SPENT ON TASK	25
ORGANIZATION PERFORMING(SQUAD	1
DAYS SPENT ON TASK	9
WORK/REST CYCLE	9
SESSION NUMBER(SHORT SESSIONS)	1
NUMBER OF POSSIBLE RESPONSES	16
MONOCULAR VS BINOCULAR VIEWING	1
SLEEP DEPRIVATION	18
NUMBER OF TASKS PERFORMED AT ONCE	8
HOURS WORKED PER WEEK	1
IMPORTANCE OF TASK PARAMETERS	4
PHYSICAL ENERGY COST OF TASK	4
NUMBER OF DISPLAYS MONITORED	5

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
SIGNAL RATE	8
DRUG USAGE	1
TIME BETWEEN STIMULUS AND RESPONSE	3
PERCEIVED PROBABILITY OF AN EVENT	11
PERCEIVED COST OF ALTERNATE DECISION	4
PROBABILITY OF AN EVENT	7
MATERIAL STRENGTH RATIO-FRIEND/FOE	1
PERSONNEL STRENGTH RATIO FRIEND/FOE	1
ENEMY TERRAIN ADVANTAGE	1
UNIT RESUPPLY CONDITIONS	1
TEAM STRUCTURE	7
NUMBER OF OBSERVERS	5
OBSERVER SCAN PATTERNS	2
SPEED OF TASK STRESS	1
BACKLOG OF WORK	2

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
CONSISTENCY OF DATA	1
OBSERVATION TIME PER DISPLAY	2
HISTORY OF SYSTEM STATES	1
URGENCY OF TASK	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
TARGET SIZE	15
TARGET BACKGROUND TYPE	13
BACKGROUND LUMINANCE	4
PERCENT TARGETS AMONG SIGNALS	2
TARGET BRIGHTNESS	6
DISTANCE TARGET HAS TRAVELED	1
RELATIVE DIRECTION OF TARGET MOTION	26
TARGET SPEED	12
TARGET TYPE	24
CONTRAST RATIO (TARGET/BACKGROUND)	16
TARGET ANGLE RELATIVE TO FOVEA	12
BLINDING FLASH ENERGY	3
NUMBER OF CUES FOR A SIGNAL	8
COMMUNICATION MODE	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
CONTROL SENSITIVITY	1
CONTROL DYNAMICS (ORDER)	4
TARGET MOVEMENT BANDWITH	7
PROCESS NOISE	5
SEARCH AREA	12
TARGET DISTANCE	35
METEOROLOGICAL RANGE(VISIBILITY)	7
TARGET HORIZONTAL DISTANCE	10
VISUAL AIDING	7
TARGET COLOR	4
STIMULUS MODE	5
RESPONSE MODE	1
LENGTH OF EARLY WARNING	2
DISPLAY TYPE	2
DISPLAY RESOLUTION	2

HIT ANY KEY TO CONTINUE

(SPACE KEY)

INDEPENDANT VARIABLE NAME	#TIMES
DISPLAY CLUTTER	8
NUMBER OF DISPLAY COLORS	1
MESSAGE LENGTH	1
MESSAGE IMPORTANCE	1
NUMBER OF SIGNAL/TARGET DIMENSIONS	1
SIGNAL DURATION	2
INTER DISPLAY ANGLE	1
DIRECTION OF ILLUMINATION SOURCE	1
SCENE COMPLEXITY	1
TARGET POSITION ON DISPLAY	1
TASK TYPE(PROCEDURAL	1
SIGNAL AMPLITUDE	3
STEP VS RAMP FAILURE	1

HIT ANY KEY TO CONTINUE

(SPACE KEY)

REPEAT? (Y/N)N

]

Appendix A

PRINTOUT OF
ENTER ARTICLES
COMPUTER PROGRAM

```
 LOAD ENTER ARTICLES
]LIST

10 LP = 18
20  DIM NI(5),ID$(5,1000),SK(10),
     TI(100)
30 D$ =  CHR$ (4)
42  PRINT D$;"OPEN SKILLS X IND V
     ARS,D2"
46  PRINT D$;"APPEND SKILLS X IND
     VARS"
50  PRINT D$;"OPEN SKILLS,D1"
70  PRINT D$;"READ SKILLS": INPUT
     SC: DIM ST$(SC): FOR I = 1 TO
     SC: INPUT ST$(I): NEXT I
72  PRINT D$;"CLOSE SKILLS"
73  PRINT D$;"OPEN IND VARS-10"
75  FOR I = 1 TO 5
80  PRINT D$;"READ IND VARS-10"
90  INPUT NI(I)
100  PRINT D$;"OPEN IND VARS-";I:
      PRINT D$;"READ IND VARS-";I
     : FOR J = 1 TO NI(I): INPUT
     ID$(I,J)
105  NEXT J: PRINT D$;"CLOSE IND
     VARS-";I
110  NEXT I
120  PRINT D$;""
500  HOME : PRINT "DO YOU WANT TO
      ADD A NEW ARTICLE TO ": PRINT
     "THE DATA BASE? (Y/N)": GET
     Q$
510  IF  LEFT$ (Q$,1) = "Y" GOTO
     1000
512  INPUT "DO YOU WANT TO QUIT?
     (Y/N)";Q$: IF  LEFT$ (Q$,1) <
      > "Y" GOTO 500
513  PRINT D$;"CLOSE"
515  END
1000  INPUT "ENTER ARTICLE NUMBER
     -",A: PRINT "ENTER QUANTITAT
     IVENESS INDEX (0-3)-": GET Q
     $
1005  IF  ASC (Q$) = 13 THEN  PRINT
     : PRINT " ERROR": GOTO 1000
1007 QU =  VAL (Q$)
1010 SN = 0
1100  PRINT "DO YOU NEED A SKILL
     CATEGORY MENU? (Y/N)": GET Q
     $: IF Q$ <  > "Y" GOTO 1500
1110 I =  INT (SC / LP):J = 1
1115  HOME : FOR K = (J - 1) * LP
      + 1 TO J * LP: PRINT K;" ";
      LEFT$ (ST$(K),36): NEXT K: PRINT

1120  PRINT "HIT 'SPACE' KEY TO C
     ONTINUE": PRINT " ANY OTHER
     KEY TO EXIT MENU": GET Q$: IF
     Q$ <  > " " THEN  PRINT : GOTO
     1500
1125  IF  LEN (Q$) < 1 GOTO 1500
1130  IF J < I THEN J = J + 1: GOTO
     1115
1150  IF I * LP = SC GOTO 1400
```

```
1160  HOME : FOR K = LP : I + 1 TO
      SC: PRINT K;" "; LEFT$ (ST$(
      K),36): NEXT K
1400  INVERSE : PRINT : PRINT "AL
      L SKILL CATEGORIES DISPLAYED
      ": NORMAL
1500  PRINT "ENTER SKILL CATEGORY
       NUMBER OR": PRINT "HIT RETU
      RN IF ALL SKILL CATEGORIES":
       INPUT "HAVE BEEN ENTERED-";
      Q$
1505  IF Q$ = "" AND SN = 0 THEN
       INVERSE : PRINT "NO SKILL C
      ATAGORIES ENTERED FOR": PRINT
      "THIS ARTICLE-START AGAIN": NORMAL : GOTO 1100
1510  IF Q$ = "" GOTO 1700
1515  IF  VAL (Q$) < 1 OR  VAL (Q
      $) > SC THEN  INVERSE : PRINT
      "VALUE OUT OF RANGE-START AG
      AIN": NORMAL : GOTO 1100
1520 SN = SN + 1:SK(SN) =  VAL (Q
      $): HOME : PRINT "ENTER NEXT
       SKILL CATEGORY": GOTO 1100
1700 IV = 1
1710  HOME : PRINT "WHAT TYPE OF
      VARIABLE IS INDEPENDANT": PRINT
      "VARIABLE NUMBER ";IV
1720  PRINT : PRINT " APTITUDE=1"
      : PRINT " LEARNING/PRACTICE=
      2": PRINT " ENVIRONMENTAL=3"
      : PRINT " TASK-PROCEDURE/STR
      ATEGY=4"
1730  PRINT " TASK-MAN/MACHINE IN
      TERFACE=5": PRINT
1740  PRINT "HIT 'RETURN' IF ALL
      VARIABLES ARE IN": PRINT "EN
      TER '9' TO SKIP THIS ARTICLE
      "
1900  GET Q$: IF  ASC (Q$) = 13 GOTO
      500
1910 IN =  VAL (Q$): IF IN = 9 GOTO
      500
1920  IF IN > 0 AND IN < 6 GOTO 2
      000
1930  INVERSE : PRINT "OUT OF RAN
      GE-REENTER": NORMAL : GOTO 1
      720
2000 N = LP
2010  HOME
2020  IF N < NI(IN) THEN  FOR I =
      N - LP + 1 TO N: PRINT I; TAB(
      5);ID$(IN,I): NEXT I: PRINT
      "HIT RETURN FOR MORE": GOTO
      2500
2050  FOR I = N - LP + 1 TO NI(IN
      ): PRINT I; TAB( 5);ID$(IN,I
      ): NEXT I
2080  INVERSE : PRINT : PRINT "AL
      L VARIABLES DISPLAYED": NORMAL
      : PRINT
2500  PRINT "ENTER VARIABLE NUMBE
      R IF FOUND": PRINT "ENTER 'A
      ' IF NEW VARIABLE IS REQUIRE
      D"
2505  PRINT "ENTER 'M' TO  RETURN
      TO VAR TYPE MENU"
```

```
2510  PRINT "ENTER 'R' IF MENU MU
      ST BE REVIEWED": INPUT Q$
2515  IF Q$ = "M" GOTO 1710
2600  IF Q$ <  > "" GOTO 3000
2620  IF N >  = NI(IN) GOTO 2080
2630 N = N + LP: GOTO 2010
3000  IF Q$ <  > "A" GOTO 4000
3010 NI(IN) = NI(IN) + 1: PRINT "
      ENTER NAME OF NEW INDEPENDAN
      T VARIABLE": PRINT "ENTER 'S
      PACE' TO RETURN TO MENU": INPUT
      ID$(IN,NI(IN))
3012  IF  LEFT$ (ID$(IN,NI(IN)),1
      ) = " " OR  LEN (ID$(IN,NI(I
      N))) < 1 THEN NI(IN) = NI(IN
      ) - 1: GOTO 2010
3013  PRINT D$;"OPEN IND VARS-";I
      N;",D1"
3015  PRINT D$;"POSITION IND VARS
      -";IN;",R";NI(IN) - 1
3020  PRINT D$;"WRITE IND VARS-";
      IN: PRINT ID$(IN,NI(IN))
3025  PRINT D$;"CLOSE IND VARS-";
      IN
3027  PRINT D$;"CLOSE IND VARS-10
      ": PRINT D$;"OPEN IND VARS-1
      0"
3030  PRINT D$;"WRITE IND VARS-10
      ": FOR I = 1 TO 5: PRINT NI(
      I): NEXT I: PRINT D$;""
3040 IO = NI(IN): GOTO 4500
4000  IF Q$ = "R" GOTO 2010
4005  IF  VAL (Q$) < 1 OR  VAL (Q
      $) > NI(IN) THEN  PRINT "INC
      ORRECT ENTRY": GOTO 2500
4010 IO =  VAL (Q$)
4500  FOR I = 1 TO SN: PRINT : PRINT
      "SKILL=";SK(I);" ";"IND VAR=
      ";IO;" ARTICLE=";A: NEXT I
4502  PRINT "ALL INFORMATION CORR
      ECT? (Y/N)": GET Q$: IF  LEFT$
      (Q$,1) <  > "Y" GOTO 2500
4503  PRINT D$;"APPEND SKILLS X I
      ND VARS"
4505  FOR I = 1 TO SN
4507  PRINT D$;"WRITE SKILLS X IN
      D VARS"
4510  PRINT SK(I);" ";IN;" ";IO;"
      ";A;" ";QU;" ";SN
4520  NEXT I
4530  PRINT D$;""
5000 IV = IV + 1: GOTO 1710
5010  END

]
```

Appendix B

PRINTOUT OF
DATA BASE ANALYSIS
COMPUTER PROGRAM

```
]LOAD DATA BASE ANALYSIS
]LIST

5  HOME
8 D$ = " "
10 LP = 18
20  DIM NI(5),ID$(5,1000),SK(10),
     TI(100),SR(60),NA(5,150),AR(
     300)
30 D$ =  CHR$ (4)
50  PRINT D$;"OPEN SKILLS,D1"
70  PRINT D$;"READ SKILLS": INPUT
     SC: DIM ST$(SC): FOR I = 1 TO
     SC: INPUT ST$(I): NEXT I
72  PRINT D$;"CLOSE SKILLS"
73  PRINT D$;"OPEN IND VARS-10"
75  FOR I = 1 TO 5
80  PRINT D$;"READ IND VARS-10"
90  INPUT NI(I)
100  PRINT D$;"OPEN IND VARS-";I:
      PRINT D$;"READ IND VARS-";I
     : FOR J = 1 TO NI(I): INPUT
     ID$(I,J)
105  NEXT J: PRINT D$;"CLOSE IND
     VARS-";I
110  NEXT I
120  PRINT D$;""
1100  PRINT "DO YOU NEED A SKILL
     CATEGORY MENU? (Y/N)": GET Q
     $: IF Q$ <  > "Y" GOTO 1500
1110 I =  INT (SC / LP):J = 1
1115  HOME : FOR K = (J - 1) * LP
      + 1 TO J * LP: PRINT K;" ";
      LEFT$ (ST$(K),36): NEXT K: PRINT

1120  PRINT "HIT 'SPACE' KEY TO C
     ONTINUE": PRINT " ANY OTHER
     KEY TO EXIT MENU": GET Q$: IF
     Q$ <  > " " THEN  PRINT : GOTO
     1500
1125  IF  LEN (Q$) < 1 GOTO 1500
1130  IF J < I THEN J = J + 1: GOTO
     1115
1150  IF I * LP = SC GOTO 1400
1160  HOME : FOR K = LP * I + 1 TO
     SC: PRINT K;" "; LEFT$ (ST$(
     K),36): NEXT K
1400  INVERSE : PRINT : PRINT "AL
     L SKILL CATEGORIES DISPLAYED
     ": NORMAL
1500  PRINT : PRINT "ENTER SKILL
     CATEGORY NUMBERS": PRINT "TO
      BE REVIEWED"
1510  PRINT "TYPE 'END' WHEN LIST
     IS COMPLETE":NS = 0
1520  INPUT Q$: IF  LEFT$ (Q$,1) =
     "E" GOTO 2000
1530  IF  VAL (Q$) < 1 OR  VAL (Q
     $) > SC THEN  PRINT : PRINT
     "OUT OF RANGE-REENTER": GOTO
     1520
1540 SR( VAL (Q$)) = 1:NS = NS +
     1: GOTO 1520
```

```
2000  PRINT D$;"OPEN SKILLS X IND
      VARS-MOD,D2"
2010  PRINT "ARE YOU INTERESTED O
     NLY IN ": INPUT "EXCLUSIVE A
     RTICLES? (Y/N)";Q$: IF  LEFT$
     (Q$,1) = "Y" THEN EX = 1
2100  PRINT D$;"READ SKILLS X IND
      VARS-MOD"
2200  ONERR  GOTO 3000
2300  FOR K = 1 TO 6: INPUT X(K):
      NEXT K
2310  IF SR(X(1)) < 1 GOTO 2300
2320  IF EX = 1 AND X(6) > 1 GOTO
     2300
2330 AR(X(4)) = AR(X(4)) + 1:NA(X
     (2),X(3)) = NA(X(2),X(3)) +
     1: GOTO 2300
3000  IF  PEEK (222) <  > 5 THEN
      PRINT "### ERROR ###": END

3010  POKE 216,0: PRINT : PRINT "
     ARE YOU INTERESTED IN THE AR
     TICLES ": INPUT "REFERENCING
      THESE SKILL CATEGORIES? ";Q
     $: IF  LEFT$ (Q$,1) = "N" GOTO
     4000
3020 I = 1:TA = 0:TR = 0
3030 PL = 1: HOME : PRINT "ARTICL
     E NUMBER      #TIMES REFERENC
    ED": PRINT "---------------
       ----------": PRINT

3040  IF AR(I) > 0 THEN  PRINT I;
      TAB( 21);AR(I);:PL = PL + 1:
     TA = TA + 1:TR = TR + AR(I)
3050 I = I + 1: IF I > 300 GOTO 4
     000
3060  IF PL > 15 THEN  PRINT "HIT
      ANY KEY TO CONTINUE": GET Q
     $: GOTO 3030
3070  GOTO 3040
4000  PRINT : PRINT "TOTAL NUMBER
      OF ARTICLES=";TA: PRINT "TO
     TAL NUMBER OF REFERENCES="TR

4005  PRINT : PRINT "ARE YOU INTE
     RESTED IN THE INDEPENDANT": INPUT
     "VARIABLES WHICH WERE REFERE
     NCED? ";Q$: IF  LEFT$ (Q$,1)
      = "N" GOTO 5000
4010 I = 1:J = 1
4020 PL = 1: HOME : PRINT "INDEPE
     NDENT VARIABLE NAME
      TIMES": PRINT "------------
     ------------              -----
     -
4030  IF NA(I,J) > 0 THEN  PRINT
      LEFT$ (ID$(I,J),35); TAB( 3
     6);NA(I,J);:PL = PL + 1
4035 J = J + 1
4040  IF J > NI(I) THEN J = 1:I =
     I + 1: IF I > 5 GOTO 5000
4060  IF PL > 15 THEN  PRINT "HIT
      ANY KEY TO CONTINUE": GET Q
     $: GOTO 4020
4070  GOTO 4070
```

```
.... .... ....
5000  PRINT : INPUT "DO YOU WISH
      TO REVIEW THE RESULTS? ";Q$:
       IF  LEFT$ (Q$,1) = "Y" GOTO
      3000
5010  END

]
```

Appendix C

PRINTOUT OF
IV REF COUNT
COMPUTER PROGRAM

```
 LOAD IV REF COUNT
]LIST

5  HOME
8 B$ = " "
10 LP = 18
20  DIM NI(5),ID$(5,1000),SK(10),
    TI(100),SR(60),NA(5,150),AR(
    300)
30 D$ =  CHR$ (4)
73  PRINT D$;"OPEN IND VARS-10,D1
    "
75  FOR I = 1 TO 5
80  PRINT D$;"READ IND VARS-10"
90  INPUT NI(I)
100  PRINT D$;"OPEN IND VARS-";I:
     PRINT D$;"READ IND VARS-";I
    : FOR J = 1 TO NI(I): INPUT
    ID$(I,J)
105  NEXT J: PRINT D$;"CLOSE IND
    VARS-";I
110  NEXT I
120  PRINT D$;""
200  PRINT D$;"OPEN SKILLS X IND
    VARS-MOD,D2": PRINT D$;"READ
     SKILLS X IND VARS-MOD"
220  ONERR  GOTO 500
230  INPUT A,B,C,D,E,F
240 NA(B,C) = NA(B,C) + 1: GOTO 2
    30
500  IF  PEEK (222) <  > 5 THEN  STOP

510  POKE 216,0
550  FOR I = 1 TO 5
555 PL = 0: PRINT : HOME : PRINT
    "INDEPENDANT VARIABLE NAME
           #TIMES": PRINT "-----
    ----------------
     ------"
560  FOR J = 1 TO NI(I)
562 PL = PL + 1
570  IF PL = 15 THEN  PRINT "HIT
    ANY KEY TO CONTINUE": GET Q$
    : HOME : PRINT "INDEPENDANT
    VARIABLE NAME          #TIMES
    ": PRINT "----------------
    -------          ------":PL =
    0
580  PRINT  LEFT$ (ID$(I,J),36); TAB(
    38);NA(I,J)
590  NEXT J
595  PRINT : PRINT "HIT ANY KEY T
    O CONTINUE": GET Q$
600  NEXT I
610  INPUT "REPEAT? (Y/N)";Q$: IF
    Q$ <  > "N" GOTO 550
620  END
```

APPENDIX M

MOPADS FINAL REPORT:

THE UNDERLYING PERSON MODEL BEHIND HOMO (HUMAN OPERATOR MODEL)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SECTION I. TECHNICAL DISCUSSION

This report presents a model of the human that will be used in MOPADS. This model is distinct from the quantitative models which will be used to predict human performance times. The quantitative models are not aimed at treating the human holistically; rather, they treat the human as a performer of particular tasks such as identification, tracking, and so on. The model presented herein is a formal model of the underlying human. The reason for creating such a formal human model is first, to provide a holistic framework within which the overall operator models can be described and discussed, and second, to serve as a cross-check on the quantitative models. The method of cross-checking will be presented later in this report.

The development of this "underlying person-model" began with a review of the air defense scenario to be modeled. We determined that the following pieces of air defense equipment would be used: Missile Minder (AN/TSQ-73), HAWK missile, Redeye missile, Chaparral missile, and Vulcan gun. The scenario involves the loading, aiming, firing, and other uses of these weapons and devices, but not setup, teardown, or movement of the weapons. Given this context, we identified six overall tasks to be performed at various positions in the air defense network. These are: tracking of targets on a CRT, assignment of targets using automated or semiautomated equipment, communication via voice or teletype, command decision making and team coordination, ammunition preparation and loading, and gunnery involving optical tracking and physical guidance of the weapon system.

Proceeding from this knowledge of tasks to be performed, we constructed a model of the human operator. Systems used extensively in air defense tasks are given greater emphasis and definition in relation to the less important body parts and activities. For example, our person-model has no digestive system, but has a huge finder dexterity system.

This model will enable the modelers to focus on those aspects of the human which are most important. It will be used in part to decide which skills are involved in any given air defense task. It will also be used as a check on the completeness of the skill moderator functions, to ensure that all moderating functions have been fully specified, that is that all important variables have been included. A moderating function is specified in terms of terms of individual, environmental, and human-machine system variables that will affect performance of any skill from the skills taxonomy, a copy of which is presented in Table I-1.

Table I-1. Skills Taxonomy.

GENERAL BEHAVIORS
VIGILANCE
PROBABILITY ESTIMATION
TIME ESTIMATION
HUMAN RELIABILITY ASSESSMENT
MEANINGFUL MEMORY ABILITY
VERBAL KNOWLEDGE
WORD FLUENCY
NUMERICAL ABILITY
CONCEPT FLUENCY
DISCOVERY OF PRINCIPLES
FLEXIBILITY
SYMBOL MANIPULATION
DECISION MAKING
INTELLEGENCE
FINE MANIPULATIVE ABILITIES
POSITION ESTIMATION
GROSS MANIPULATIVE ABILITIES
CONTROL PRECISION
SPEED OF ARM MOVEMENT
MULTILIMB COORDINATION
POSITION REPRODUCTION
MOVEMENT ANALYSIS
MOVEMENT PREDICTION
ACCELERATION CONTROL
REACTION TIME
TRACKING
DETECTION
DISCRIMINATION ABILITIES
TIME SHARING
CLOSURE ABILITIES(PERCEPTUAL)
AUDITORY ABILITIES
SPATIAL ABILITIES-ORIENTATION
SPATIAL ABILITIES-VISUALIZATION
ASSOCIATE MEMORY-ROTE MEMORY
ASSOCIATE MEMORY-MEANINGFUL MEMORY
MEMORY SPAN-SHORT TERM MEMORY
MEMORY SPAN-INTEGRATION OF INFORMATION
VISUAL MEMORY
EXPLOSIVE STRENGTH-GENERAL

Table I-1 (continued)

EXPLOSIVE STRENGTH-LEG EMPHASIS
EXPLOSIVE STRENGTH-UPPER BODY EMPHASIS
STATIC STRENGTH-ARM/HAND/SHOULDER
STATIC STRENGTH-LEG/TRUNK
DYNAMIC STRENGTH-ARMS
DYNAMIC STRENGTH-LEGS
TRUNK STRENGTH
EXTENT FLEXIBILITY
DYNAMIC FLEXIBILITY
GROSS BODY EQUILIBRIUM
BALANCE-VISUAL CUES
GROUP COOPERATION
GROUP COMMUNICATION
LEADERSHIP
GENERAL COGNITIVE ABILITIES
GENERAL PHYSICAL ABILITIES

The person-model itself is a collection of human systems and activities. One constraint, stated above, is that they be relevant to the air defense scenario. Activities which are not relevant are not part of the model. Another constraint on these activities is that they can be viewed as bodily systems.

The model divides the human into five systems: 1) perceptual, 2) information processing, 3) information storage, 4) motor, and 5) affective. In accordance with the above description of the model's purpose, these five systems represent areas of importance to the air defense modeling goal. It would have been possible to compress or expand this list, but it was felt that these represent equivalent levels of description. Each of the systems is then subdivided into one or more activities.

The model can be understood best by reviewing these systems and activities, along with a definition of each activity. The reader who wishes to understand this model is urged to build a mental picture of a human which has only these activities as its attributes. Table I-2 presents an overview of the activity listing.

Figure I-1 presents a matrix of the interrelationships between the skills in the taxonomy and the subsystems of the person-model. Those cells of the matrix which are marked by an "X" indicate that the subsystem (identified by the column title) plays a significant part in the performance of the skill (identified by the row title). This matrix was prepared based upon an intuitive examination by the authors.

The next step in the use of this person-model behind MOPADS, will be as a cross-check on the completeness of the moderator functions. Once the candidate set of independent variables that will mediate human performance is prepared, (to be summarized in Laughery (1983)) another matrix will be developed. This matrix will be similar to that presented in Figure I-2. The cells in this matrix marked with an "X" represent a subsystem which is affected in some manner by an independent variable. This matrix will also be developed on the basis of intuitive judgements by the authors. Since it is intended to be only a cross-check on the completeness of the moderator functions, a full-scale research effort on the development of this matrix is unnecessary. Finally, simple matrix algebra will be performed on the matrices in Figures I-1 and I-2 to determine which independent variables should be included to moderate each skill in the taxonomy. These results will then be compared to the moderator functions being developed to determine whether they include all of the independent variables suggested by this analysis. At this point, suggestions for further literature review and research where required will be amended to this report.

Table I-2. Person-Model Systems and Subsystems

I. Perceptual System

1. Auditory subsystem - The perception of sound stimuli for communication, target location, or any other purpose.
2. Visual subsystem - The perception of visual stimuli.
3. Tactile subsystem - Touch and pressure stimuli at any part of the body. This includes vibration stimuli.
4. Proprioceptive subsystem - The perception of body part and limb position, either kinesthetic or static body position.
5. Vestibular subsystem - Perception of the upright and all balance activities. This subsystem also includes perception of acceleration.
6. Pain subsystem - The registration of pain or discomfort from any physical source. Input sources may be body damage or fatigue.

Table I-2 (contirued)

II. Information Processing System

1. Attention - The act of focussing processing power on a given object or topic.
2. Problem solving activity - The act of generating potential solutions (alternatives) to problems. This process is involved, for example, in determining which weapons will have an effective envelope which includes the target, at an AN/TSQ-73.
3. Decision making activity - The process of deciding among alternatives, based on knowledge of subjective and objective costs and benefits. The assignment of targets to weapons at the AN/TSQ-73 is an example of decision making.
4. Symbolic output - The activity of communicating via symbols, verbal or visual. This would include speech, typing, or symbol selection using a CRT and light pen.

Table I-2 (continued)

III. Information Storage System

1. Long term symbolic memory - The location of information in symbolic code which must be retained for a time greater than five minutes. Symbols include words, letters, symbolic pictures, or similar devices. An example would be remembering AN/TSQ-73 screen codes.

2. Long term scene memory - The location of information on scenes, pictures, or non-symbolic sound patterns which must be retained for more than five minutes. An example would include remembering what a MIG-15 looks like.

3. Short term memory - Memory for information which needs to be stored for less than five minutes and greater than five seconds, such as recent commands.

Table I-2 (continued)

4. Iconic/echoic memory - Memory for visual or auditory stimuli presented within the past five seconds. In some cases this activity may be considered to be of longer duration, such as when a response is deferred due to an activity already initiated. An example of iconic memory acting alone would be remembering target aircraft aspect and speed as it goes behind a shadowing object.

IV. Motor System[1]

1. Erect locomotor activity - The act of walking, running, or jumping, with or without a load.

2. Non-erect locomotor activity - Moving the body in some way other than walking, running, or jumping, such as crawling, or shifting one's weight in a chair.

[1]Parts of this model are derived from Harrow's A Taxonomy of the Psychomotor Domain, (1972).

Table I-2 (continued)

3. Non-locomotor movement with heavyweight - Movement involving the limbs, or portions of the trunk moving around an axis, performed while carrying a moderate to heavy load. Loading a Chaparral missile into a launcher is an example of this type of activity coupled with locomotion.

4. Non-locomotor movement with light weight - Movement involving the limbs, or portions of the trunk moving around an axis, performed while carrying a light load or no load. An example would be linking ammunition for a Vulcan.

5. Prehension manipulation activity - Reaching for, gripping, and releasing objects. Holding a joystick or any equipment part are examples of prehension.

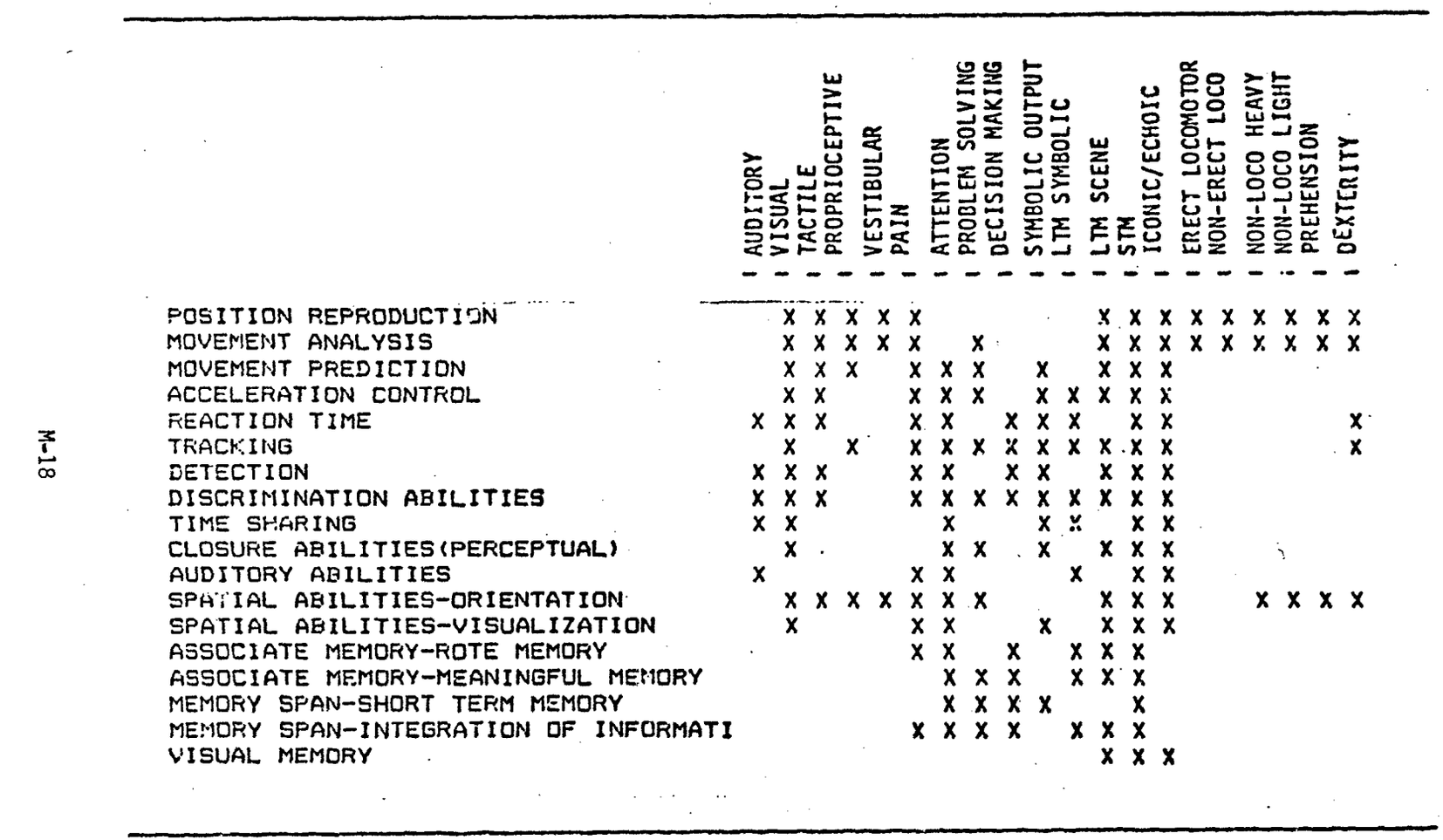
6. Dexterity manipulation activity - Quick, precise movements of the hand and fingers. Typing requires dexterity.

Table I-2 (continued)

V.	Affective System
	1. Motivation to act - The internal emotional state governing whether a person will perform an activity, and how much effort a person is willing to expend in this direction. This motivation may be affected by memories or by recent perceptual inputs, such as pain.

	AUDITORY	VISUAL	TACTILE	PROPRIOCEPTIVE	VESTIBULAR	PAIN	ATTENTION	PROBLEM SOLVING	DECISION MAKING	SYMBOLIC OUTPUT	LTM SYMBOLIC	LTM SCENE	STM	ICONIC/ECHOIC	ERECT LOCOMOTOR	NON-ERECT LOCO	NON-LOCO HEAVY	NON-LOCO LIGHT	PREHENSION	DEXTERITY
GENERAL BEHAVIORS	X	X			X	X		X		X				X	X	X			X	X
VIGILANCE	X	X	X			X														
PROBABILITY ESTIMATION							X	X	X		X		X							
TIME ESTIMATION								X			X		X							
HUMAN RELIABILITY ASSESSMENT							X				X		X							
MEANINGFUL MEMORY ABILITY						X					X	X	X							
VERBAL KNOWLEDGE							X		X	X	X	X	X	X						
WORD FLUENCY							X	X	X	X	X	X	X	X						
NUMERICAL ABILITY							X	X	X	X	X	X	X	X						
CONCEPT FLUENCY	X	X					X	X	X	X	X	X	X	X						
DISCOVERY OF PRINCIPLES		X	X				X		X		X	X	X	X						
FLEXIBILITY									X		X	X	X	X						
SYMBOL MANIPULATION									X	X	X	X	X	X						
DECISION MAKING									X	X	X	X	X	X						
INTELLEGENCE									X	X	X	X	X	X						
FINE MANIPULATIVE ABILITIES							X		X		X		X							
POSITION ESTIMATION	X	X	X	X	X	X	X	X	X		X	X	X							
GROSS MANIPULATIVE ABILITIES		X	X	X	X	X	X	X	X		X	X	X							
CONTROL PRECISION		X	X	X	X	X	X	X	X		X	X	X							
SPEED OF ARM MOVEMENT			X	X	X	X	X	X	X		X	X	X							
MULTILIMB COORDINATION		X	X	X	X	X	X	X	X		X	X	X							

Figure I-1. Skills X Person Parts Matrix



	AUDITORY	VISUAL	TACTILE	PROPRIOCEPTIVE	VESTIBULAR	PAIN	ATTENTION	PROBLEM SOLVING	DECISION MAKING	SYMBOLIC OUTPUT	LTM SYMBOLIC	LTM SCENE	STM	ICONIC/ECHOIC	ERECT LOCOMOTOR	NON-ERECT LOCO	NON-LOCO HEAVY	NON-LOCO LIGHT	PREHENSION	DEXTERITY
POSITION REPRODUCTION		X	X	X	X	X						X	X	X	X	X	X	X	X	X
MOVEMENT ANALYSIS		X	X	X	X	X		X				X	X	X	X	X	X	X	X	X
MOVEMENT PREDICTION		X	X	X		X	X	X		X		X	X	X						
ACCELERATION CONTROL		X	X			X	X	X		X	X	X	X	X						
REACTION TIME	X	X	X			X	X		X	X	X		X	X						X
TRACKING		X		X		X	X	X	X	X	X	X	X	X						X
DETECTION	X	X	X			X	X		X	X		X	X	X						
DISCRIMINATION ABILITIES	X	X	X			X	X	X	X	X	X	X	X	X						
TIME SHARING	X	X					X			X	X		X	X						
CLOSURE ABILITIES(PERCEPTUAL)		X					X	X		X		X	X	X						
AUDITORY ABILITIES	X					X	X				X		X	X						
SPATIAL ABILITIES-ORIENTATION		X	X	X	X	X	X	X				X	X	X			X	X	X	X
SPATIAL ABILITIES-VISUALIZATION		X				X	X			X		X	X	X						
ASSOCIATE MEMORY-ROTE MEMORY						X	X		X		X	X	X							
ASSOCIATE MEMORY-MEANINGFUL MEMORY							X	X	X		X	X	X							
MEMORY SPAN-SHORT TERM MEMORY							X	X	X	X			X							
MEMORY SPAN-INTEGRATION OF INFORMATI						X	X	X	X		X	X	X							
VISUAL MEMORY												X	X	X						

Figure I-1. (continued)

	AUDITORY	VISUAL	TACTILE	PROPRIOCEPTIVE	VESTIBULAR	PAIN	ATTENTION	PROBLEM SOLVING	DECISION MAKING	SYMBOLIC OUTPUT	LTM SYMBOLIC	LTM SCENE	STM	ICONIC/ECHOIC	ERECT LOCOMOTOR	NON-ERECT LOCO	NON-LOCO HEAVY	NON-LOCO LIGHT	PREHENSION	DEXTERITY
EXPLOSIVE STRENGTH-GENERAL						X									X X	X X X	X X X			
EXPLOSIVE STRENGTH-LEG EMPHASIS						X X X									X X	X X X	X X X			
EXPLOSIVE STRENGTH-UPPER BODY EMPHAS						X X X										X X	X X X			X
STATIC STRENGTH-ARM/HAND/SHOULDER				X		X X X									X	X X	X X X	X X X	X X X	X X X
STATIC STRENGTH-LEG/TRUNK						X X X										X X	X X X	X X X	X X X	X X X
DYNAMIC STRENGTH-ARMS						X X X										X X	X X X		X X X	X X X
DYNAMIC STRENGTH-LEGS						X X X										X X	X X X			X X X
TRUNK STRENGTH						X X X									X X	X X	X X X			X X X
EXTENT FLEXIBILITY			X X	X X	X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
DYNAMIC FLEXIBILITY			X X	X X	X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
GROSS BODY EQUILIBRIUM		X X	X X		X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
BALANCE-VISUAL CUES		X X	X X		X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
GROUP COOPERATION	X X	X X				X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
GROUP COMMUNICATION	X X	X X				X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
LEADERSHIP	X X	X X				X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
GENERAL COGNITIVE ABILITIES	X	X X	X X			X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X	X X X			X X X
GENERAL PHYSICAL ABILITIES			X X		X X	X X X									X X X	X X X	X X X	X X X	X X X	X X X

Figure I-1. (continued)

PERSON-MODEL SUBSYSTEM / INDEPENDENT VARIABLE	INDEPENDENT VARIABLE 1	INDEPENDENT VARIABLE 2		INDEPENDENT VARIABLE N
PERCEPTUAL				
AUDITORY	×	×		
VISUAL		×		
TACTILE				×
PROPRIOCEPTIVE	×			×
VESTIBULAR		×		
PAIN				
INFORMATION PROCESSING				
ATTENTION		×		
PROBLEM SOLVING	×	×		
DECISION MAKING	×	×		
SYMBOLIC OUTPUT				×
INFORMATION STORAGE				
LONG-TERM SYMBOLIC				×
LONG-TERM SCENE				
SHORT TERM	×			
ICONIC/ECHOIC		×		
MOTOR	×	×		
ERECT LOCOMOTOR				
NON-ERECT LOCOMOTOR		×		
NON-LOCOMOTOR-HEAVY	×			×
NON-LOCOMOTOR-LIGHT				
PREHENSION				
DEXTERITY		×		

Figure I-2. Example of Matrix of Relationships Between Person-Model Subsystems and Independent Variables

II. REFERENCES

Laughery, K. R. HOMO establishment of performance criteria for non-decision making tasks (MOPADS Vol. 5.6). Pritsker & Associates, Inc, (Calspan Corporation subcontractor)prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX,1983.

III. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACC-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

APPENDIX N

MOPADS FINAL REPORT:

MAKING SAINT MODELS MORE REPRESENTATIVE OF HUMAN BEHAVIOR:

THE THEORY AND APPLICATION OF THE MOPADS SKILL MODERATOR FUNCTION SUBROUTINES

The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as official Department of the Army position, policy, or decision, unless so designated by other official documentation.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

I. PURPOSE

The purpose of this report is to document the thinking and procedures behind the development of the skill moderator functions described in MOPADS Report 5.10. In report 5.10., a series of computer subroutines are defined which will facilitate more realistic modeling of human operator performance. Herein, we will discuss the logic behind this approach and the steps taken to develop the subroutines presented in MOPADS Report 5.10.

The remainder of this section will present some background information for individuals with a limited understanding of human operator modeling via network simulation. Section II will discuss the logic behind the development of these subroutines as well as presenting a brief description of how they would be incorporated into a simulation (a more detailed discussion of specific steps is presented in MOPADS Report 5.6). Section III will discuss the steps which were followed in the development of these skill moderator function subroutines. Finally, Section IV will briefly define some advantages and disadvantages of this approach.

1-1 Background

Simulation modeling is a powerful analytic tool whose potential is just being recognized. Low cost computational capabilities and readily available simulation languages are sure to result in a rise in the use of simulation to answer questions which would otherwise go unanswered.

Additionally, simulation modeling is being applied to new problem areas. One of the more promising applications of simulation is in modeling and evaluating human operator performance. In the past, the primary means for evaluating human performance was empirical data collection. For systems or situations which did not yet exist, the only recourse available to the analyst was to look for existing analogous situations. Then, the analyst could draw inferences about

how the human would perform in the new situation by estimating the differences between the new and the old situation. In many cases, reasonably analogous situations did not exist. Even when they did exist, the process of extrapolating to the new situation was largely based on intuition. The result was a poor, if any, estimate of human performance.

Now, simulation tools exist which are specifically designed for modeling the human operator of a system. The analyst interested in evaluating human performance can use these tools to construct a computer model of the human in much the same way that a bridge designer can build a computer model of a bridge. Then, the computer model can be used to test various concepts of human operator/system design in a reasonable manner long before the system or situation actually exists. Additionally, exercising computer models is far less costly than experimentation with human subjects. Therefore, more alternatives can be considered within the same amount of time and resource expenditures.

Over the past few years, techniques for better simulation of humans within systems have become available (e.g., Wortman, Duket, Hann, Chubb, and Seifert; 1976, Streib and Wherry, 1979). One of the most notable approaches is task network simulation as used by the SAINT (Systems Analysis of Integrated Networks of Tasks) simulation language. With SAINT, human performance is separated into a network of tasks. Each node of the network represents a discrete task performed by the human. The human only performs one task at a time. The network structure of the tasks defines ordering relationships representing how the human performs the tasks. Therefore, the task that the human is performing during the simulation is represented by the node in the network currently being executed. Associated with each node are the following attributes:

1. Time to complete the task.

2. If the task is always followed by the same task, the task to be performed after the current task is complete.

3. If the task can be followed by several tasks, the list of possible tasks to be performed after the current task is complete as well as decision rules for selecting among each of the possible next tasks.

4. Any user code to modify simulation variables when the task is performed, if performance of the task affects any of these variables.

5. Task number (arbitrarily assigned).

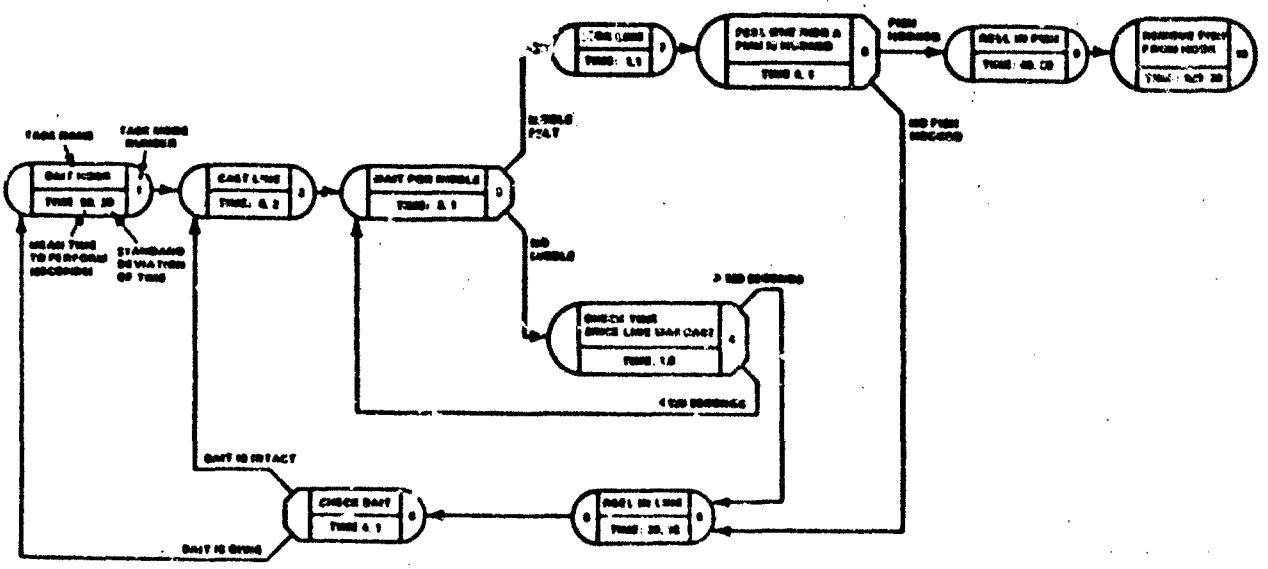
6. Task name.

Once these attributes are determined for each subtask, a computer model of the task network can be readily developed via the SAINT modeling language. Once the model is computerized, simulation experiments can be conducted by varying the subtask attributes. For example, if we want to explore the overall effect of the slower performance of some of the subtasks, we would simply change the "time to perform" attribute of those subtasks and then run the simulation model. Since performance times are probabilistic (i.e., they are randomly sampled from a known distribution), we can obtain estimates of the probability distribution representing time to perform the overall task network.

To provide an example of a SAINT network, let us use an example of a human fishing. Figure I-1 presents a visual depiction of our proposed network of a human fishing. As one follows through the network, you can see the human cast, wait for a nibble, attempt to catch a fish if a nibble occurs, or reel in the line after a period of time if no nibble occurs. If, for example, we wanted to explore the effect on the time required to catch a fish of reeling in the line more frequently to check the bait, we would simply change the attribute of task number 4 from 120 seconds to whatever number is desired. This, of course, would confound with the

Figure I-1

SAINT NETWORK FOR A HUMAN FISHING



probability of a nibble occurring in task 3. A parametric experiment testing the effects of these two variables could easily be conducted using SAINT. This example is intended to provide the reader with an example of SAINT modeling using a familiar task. Far more practical applications have been found for SAINT including visual search (e.g., Kraiss and Knaeuper, 1982) and aircraft piloting (e.g., Muralidharan, Baron, and Feehrer, 1979). It is safe to say that SAINT has become the de facto standard language in task network modeling of human performance.

The remainder of this report discusses an approach to improving the predictive ability of SAINT or other task network models of human operator performance.

II. THE LOGIC AND CONCEPTS BEHIND THE MODERATOR FUNCTION APPROACH TO INCLUDING HUMAN VARIABILITY IN SAINT MODELS

In most previous applications of SAINT, the time required by the human to perform any task in a network was defined by a probability distribution type and the associated distribution parameters (e.g., normal distribution with a mean of 4 seconds and a standard deviation of 1 second). No attention was given to the fact that human performance variability is not truly random but, rather, somewhat predictable. In other words, when a human's performance speeds up or slows down significantly, it is not usually because of random variation but because of causal factors such as heat stress, training level, and/or a host of other ability, environmental, or task variables which could affect the human's performance. We know that these variables influence human performance but how can we quantify these relationships in a manner that can be used by task network models?

What was needed to include the effects of "human factors" into SAINT models were mathematical functions which relate changes in human performance to the values of independent variables of interest (e.g., ambient temperature, drug use, etc.). Moreover, we can assume that these changes in performance will not be constant among all tasks in a task network. For example, the use of some drugs may affect cognitive performance but not motor performance. Consequently, those tasks in the network involving cognitive behavior would be affected and those tasks involving motor behavior would not be affected. In essence, a unique mathematical relationship linking human performance to the values of human factors independent variables needed to be developed for every task in a network. Stated mathematically for every task in the task network we need the following:

Change in time to perform task i in the network $= f_i (x_1, x_2, x_3, \ldots x_n)$

where

f_i = the function relating changes in performance of task i to the independent variables known to affect performance of task i

x_j = value of independent variable j which affects the performance of task i

n = the number of independent variables which affect the performance of task i

Even in a relatively small task network of 100 tasks, this would entail the development of 100 functions to "moderate" task performance. What was needed, and has been developed, is a method of building these moderator functions in a logical and timely manner.

On the other hand, one general moderator function could be developed for all tasks. The problem with using the same moderator function for each task in a network is that human factors variables will affect different skills in different manners. As was mentioned previously, some variables will affect cognitive skills only, some will affect only motor skills, and some will affect both. Even a variable which affects cognitive skills only may not affect all kinds of cognitive skills. For example, fine-motor skills may be affected by cold stress whereas numeric manipulation skills may not be. However, there may be a level of skill breakdown where all tasks involving a certain skill type will be similarly affected by human factors variables. In other

words, we may be able to define a finite set of skill categories so that all tasks requiring a particular skill is affected by the human factors variables in approximately the same way.

For example, if one of the skill categories is "fine motor manipulation", then all tasks which involve the skill of fine motor manipulation will be affected by human factors variables in the same way. Then, we should be able to develop a moderator function for tasks requiring fine motor manipulation which relates task performance changes to independent variables known to affect fine motor manipulation. For example, let us assume that we know the relationship between fine motor manipulation and the variable "hours since last sleep was received" to be the following:

$$\text{Change in time to perform a task involving fine motor manipulation skills} = \text{Mean time to perform the task} \times \frac{(\text{hours since last sleep}) - 10}{20.0}$$

(if hours since last sleep >10.0)

or

= 0

(if hours since last sleep <10.0)

This says that for any task involving fine motor manipulation, we can expect a 5.0% increase in time to perform the task for every hour beyond 10.0 hours since the operator last slept. Note that this function would only apply to tasks involving fine motor manipulation. Other skills might have other functions relating skill performance changes to sleep deprivation. Additionally, fine motor manipulation might be affected by a host of other independent variables such as ambient temperature, hours of sleep received, etc.

We took the aforementioned approach for the MOPADS effort. Obviously, in selecting this approach, two major hurdles had to be overcome. First, what was the set of skills which could be used to describe all operator task types? Second, for each skill, what human factors variables affected skill performance and what was the quantitative link (e.g., moderator function) between skill performance and these independent variables?

Two years were spent answering the above questions. The skill categories were selected by reviewing numerous categorization schemes of human performance. We were looking for the right balance of detail and, yet, not so many categories that it would be difficult to decide which skill correctly describes a task. The categories which were finally selected are presented in Table 1.

Table 1
SKILL CATEGORIES

1. Detection
2. Fine Manipulative Ability
3. General Physical Effort
4. Gross Manipulative Ability
5. Long-term Memory/Sensory
6. Long-term Memory/Symbolic
7. Numeric Manipulation
8. Probability Estimation
9. Reaction Time
10. Recognition
11. Short-term Memory/Sensory
12. Short-term Memory/Symbolic
13. Team Coordination
14. Time Estimation
15. Timesharing
16. Tracking

Then, we had to quantitatively link performance of a task requiring that skill to variables known to affect that skill. This required two steps. First, we had to determine which human factors variables affected each skill. Second, we had to mathematically define how the variables affected the skills. Obviously, we did not want to develop "armchair" models. Rather, we wanted our model to reflect sound theory and/or empirical data regarding human behavior. Consequently, we wanted to base the moderator functions on existing human performance literature.

The development of these skill moderator functions will be described in detail in the following section. The functions and their use are presented in detail in MOPADS Report 5.10. To better facilitate an understanding of this approach, let us briefly describe how they would be used in a task network simulation.

2-1 Using the Moderator Functions

The task to the modeler is reduced to relatively few activities. First, he or she must identify the skills involved in each task. It is anticipated that some tasks will involve more than one skill. In this case, the modeler must decide the relative importance of each skill by assigning a percentage to each of the component skills of the task (the percentage must add up to 100 percent).

Second, the modeler must develop a data structure scheme to maintain the independent variables. In the MOPADS context, the information is maintained in the MOPADS data base. If these moderator functions are used in another setting, the data may be maintained in any data base accessible to the moderator function module in a standard format. To accomplish this, the modeler must build the following four subroutines to transfer the values of the independent variables to the moderator function subroutines:

1. GETEVA
2. GETOVA
3. GETTSA
4. TIMEA

More detail on the parameters which are passed to and from these subroutines are covered in MOPADS Report 5.10.

Finally, the modeler must write a small amount of software to identify the skills involved in the task currently being executed by the model, call the moderator function subroutines for these skills, and then combine the new estimates of the mean (if multiple skills are required by the task) into a single weighted estimate.

A specific task moderator function, which determines the modified value for an individual task performance at any point in the simulation, will then be represented by the following weighted sum.

$$\text{Moderated mean task performance for task } t = M_t + P_{t_1} \Delta_{S_1} + P_{t_2} \Delta_{S_2} + \ldots P_{t_m} \Delta_{S_m}$$

where

M_t = the a priori mean value for task performance

m = the number of skills required in the performance of task t

S_i = the i^{th} skill involved in task performance (from the skill list in Table 1)

P_{t_i} = the proportion of skill S_i required in task t

= the value of skill moderator function i at this point in the simulation

As the simulation proceeds, the endogenous independent variables which affect the value of will change reflecting environmental or task changes. Between simulations, the analysts may wish to change the value of exogenous independent variables, such as aptitude or learning variables, to examine their effect on performance.

As an example, consider an assembly line inspection task which is one of several performed by an operator. For this particular task, the operator must pick up an object, move it around in his or her hands, check to see if there are any flaws in the object and, if so, decide whether to accept or reject it. This task involves three skills (from Table 1):

1. Fine motor manipulation (pick the object up and move it around)
2. Visual Detection (check to see if there are any flaws)
3. Visual Recognition (if there are flaws, decide to reject or accept the item)

This task may be represented by one task node in the task network. Moreover, by watching an operator performing the task, we determine that the task requires 30%, 50%, and 20% of fine motor manipulation, visual detection, and visual recognition, respectively. Mean time to perform the task (the parameter of interest) is normally 20 seconds. In this case:

m = 3 (the number of skills involved in the task)

S_1 = 13 (fine manipulative ability from Table 1)

S_2 = 2 (visual detection)

S_3 = 4 (visual recognition)

P_{t_1} = 0.3

P_{t_2} = 0.5

P_{t_3} = 0.2

M_t = 20

Therefore, the moderator function for this task will be the following:

$$\text{Moderated task performance time} = 20 + 0.3\Delta_{13} + 0.5\Delta_2 + 0.2\Delta_4$$

where

$\Delta_{13} = f_{13}(\bar{X}_{13})$

$\Delta_2 = f_2(\bar{X}_2)$

$\Delta_4 = f_4(\bar{X}_4)$

where

$\bar{X}_i$ = the array of independent variables known to affect performance of skill i.

For argument's sake, let us say that $\Delta_{13} = 3$, $\Delta_{2} = -1$, and $\Delta_{4} = 5$. At some hypothetical point in the simulation. The moderated mean performance time would equal the following:

$$\begin{aligned} \text{time} &= 20 + .3(3) + .5(-1) + .2(5) \\ &= 21.4 \text{ seconds} \end{aligned}$$

It is also worthy to note that these moderator functions should not be viewed as "sacred cows not to be fooled with." In fact, we hope that the moderator functions can continue to develop and grow with the state-of-the-art in mathematical modeling of human performance. Special care has been given to document every segment of code which affects modeled human performance so that the source of that mathematical relationsnip can be determined. During the course of developing these subroutines, many decisions had to be made as to which model was best and, therefore, should be included. Others may justifiably dispute these judgements. We encourage anyone wishing to change these subroutines to do so. We ask only that, if possible, these changes be forwarded to the authors so that we may continue to improve the fidelity of the skill moderator subroutines.

III. THE MODERATOR FUNCTION DEVELOPMENT PROCESS

The human factors literature is particularly weak in most areas of quantitative human performance modeling and prediction. Therefore, our task was basically two-fold. First, we had to review the literature to locate articles and publications which were conducive to quantitative human performance modeling. The second, and more difficult, job was to take this conglomerate of literature and process it into usable computer code.

For the remainder of this section, we will separate our efforts in the development of moderator functions into the following five steps:

1. Literature review
2. Development of a computerized literature data base
3. Selection of articles/publications for use in the moderator functions
4. Development of equations relating human performance to variables which effect performance
5. Development and testing of moderator functions

3-1 Literature Review

A computer literature search was conducted on the DTIC and Pyschological Abstracts files of the DIALOG data base. The underlying search theme was Quantitative Human Performance Models. Also, an iterative search procedure yielded a number of key words which might yield literature relevant to this topic. A total of 350 abstracts were reviewed from which 115 articles were selected as potentially useful for the proposed effort.

During the review, the reference sections of particularly interesting articles were examined. Those references which were not already in the inventory were ordered and also reviewed. Two articles which significantly affected the final model were Siegel (1979) and Pew, et al (1977).

All articles were given a cursory review to determine their potential utility. If this review was positive, a more thorough examination was conducted. This thorough review was summarized on a form, an example of which is shown in Figure II-1.

Each model within an article was also summarized on the form shown in Figure III-1. Because some articles included a number of human performance models, it was frequently the case that an article was summarized on more than one form.

The categories on the forms are defined as follows:

1. Reference. Information regarding author(s), title, and other source data.

2. Appropriateness. A judgement was made by the reviewer regarding the extent to which this article would be useful. These judgements were based upon two criteria. First, did the article present a significant contribution to human performance modeling? If the answer was yes, the article was appropriate. Secondly, did the article provide data which would be useful for modeling or validating air defense systems? In other words, while the article may not be generally useful for modeling, could it provide some task specific information? If so, the article was appropriate.

3. Dependent Variables. The specific dependent variable(s) defined by the model is described.

4. Independent Variables. Independent variables which were included as mediators of the dependent

Figure III-1
FORMS USED TO SUMMARIZE LITERATURE

Reference Phatak, A., Weinert, H., Segall, I., and Day, C.N. Identification of a modified optimal control model for the human operator. Automatica, 1976, 12, 31-41.

Appropri: Yes

Dependent Variables

Tracking

Independant Variables

Operator gain vector
Observation noise covariance

Page 35-38

Behaviors Tracking

Quantitativeness

Functions

Supporting Data

- ☐ No Validation
- ☒ Validation with Questionable Results
- ☐ Validated

Comments

Simplification of the optimal control model
The intent is to increase the identifiability of the model parameters it seems to have worked

variables are defined. No units were recorded, but a short description was included.

5. Page. The page(s) where the model was best summarized was recorded. This was intended to simplify future references to the model.

6. Behaviors. The general behavior categories which were modeled were defined. These categories were selected from the taxonomy in Table II-1. Normally, a model would fit under only one behavior category. If this was not the case, all relevant behavior categories were listed.

7. Quantitativeness. The degree of quantitative model development was recorded. The lowest degree of quantitativeness was nominal, or zero quantitativeness. This implied that only a verbal description of the relationship between the independent and dependent variables was provided. The next level was unscaled graphs, in which the independent and dependent variables were presented as constructs without specific units. The third level was tables, or scaled graphs, in which units were attached to independent and dependent variables. Finally, the highest level was functional, in which case mathematical models were presented.

8. Supporting Data. The extent to which the model was supported by data was determined. If no data or references were presented, "No Validation" was indicated. If the data were questionable, this was also indicated. Admittedly, this judgement was subjective; however, at some point, this type of judgement must be made.

9. Comments. Additional comments were recorded.

3-2 Development of a Computerized Literature Data Base

To facilitate the development and analysis of a data base on models of human performance, a set of computer programs and a computerized data base were developed. Through the use of these programs, human performance models can easily be stored into the data base and later analyzed in a number of different ways. This approach was more practical than hand recording of the data because computer input of the data was less time consuming than hand recording, and each computer analysis required only execution of a program, whereas manual manipulation would require slower hand calculations and data restructuring. Also, additions and modifications to the data base were less cumbersome using a computerized data base.

Three programs were written to create, modify, and analyze the data bases. They are currently on an Apple II Plus microcomputer with 48K RAM. These programs are as follows:

1. ENTER ARTICLES. This program allows the entry of models from articles in the open literature into the computer data base.

2. DATA BASE ANALYSIS. This program permits an examination of the data base to determine independent variables which affect skill categories, number of times the variable is referenced, and articles referencing the skill categories.

3. IV REF COUNT. This program counts the total number of times that all independent variables are referenced in the data base.

For more details on the programs, see Laughery (1982a).

3-3 Selection of Articles/Publications For Use In The Moderator Functions

In many cases in the literature, there were several articles which related skill performance to a particular human factors independent variable or set of independent variables. This provided two alternatives regarding how to model that skill/independent variable relationship in our moderator functions:

1. Take the data and/or models from all of the articles and use them simultaneously (e.g., average the predicted results from all of the models).

2. Select one model and/or data set and use that to link skill performance to independent variables.

The first alternative was rejected because of two factors. First, it would necessarily increase the length and execution time of many moderator functions. While execution time was not a driving factor in moderator function development, it was an issue to which we were sensitive. However, if the moderator functions were overly complex, as they would be if all models were included, we believed that this would decrease the face validity of the moderator function approach. Consequently, we elected the second alternative, select one model and/or data for inclusion, as our approach.

The selection of which model to include was made by considering the following factors:

1. Was the model based on theory, data, or both?

2. Had a predictive validation study been conducted on the model?

We would accept models which were based upon data over those based strictly upon theory. Additionally, if a model had

been validated, we would accept it over another model which had not.

There are two other points which should be mentioned here. First, if interactions were involved we would accept a variable more than once. For example, if variable A were found in two separate models to interact with variables B and C, respectively, then variable A would be included in both models in the moderator function. To do otherwise would have required that we "hypothesize away" interactions.

The second point relates to the modular nature of these models. In each moderator function every model is clearly documented with respect to its source including references and page numbers. As stated in the previous section, if any user of the moderator functions has a better model, then that model can be substituted by changing a few (e.g., less than five) lines of computer code. This modularity was seen as necessary for expansion and improvement of the moderator functions.

3-4 Development of Equations Relating Human Performance to Variables That Affect Human Performance

To say that the human factors literature is not generally aimed at quantitative human performance prediction is to understate the point. The human factors community has largely ignored the development of quantitative human performance prediction models in favor of normative models and inferential statistical analysis. There are, of course, some notable exceptions. However, we have generally contented ourselves with statements regarding the direction of effects of variables affecting human performance without attempting to determine the extent of the effect as a function of the variable's value.

Perhaps the best examples of this are the frequently encountered studies which use regression analysis to analyze their data. In many cases, a variety of regression statistics were reported, including percent of variance accounted for and type I error probability; however, the

coefficients of the regression equation were not reported. From the modeler's perspective, the coefficients of the regression equation are the essence of the data.

Because of this inattention of the human factors literature to predictive model development, the greatest portion of the task of developing equations involved the running of statistical analyses on either raw data or descriptive statistics in order to develop the equations.

Calspan developed a set of computer programs for an Apple II microcomputer which allowed us to develop the predictive models interactively. The programs were written so that when the data from an article or report were entered into the computer they were automatically stored in a data file where they could then be analyzed via one of the following seven methods:

1. Simple Regression ($y = a + bx$)
2. Exponential Regression ($y = a + e^{bx}$)
3. Inverse Exponential Regression ($y = a + e^{-bx}$)
4. Squared Exponential Regression ($y = a + e^{bx^2}$)
5. Inverse Squared Exponential Regression ($y = a + e^{-bx^2}$)
6. Polynomial Regression ($y = a + b_1x + b_2x^2$)
7. Multiple Regression ($y = a + b_1x_1 + b_2x_2$)

where a and b are parameters estimated from the data, y is the dependent variable, and x is the independent variable.

Once the data were stored in a data file, they could be reanalyzed by any of the seven methods simply by specifying the file name at the beginning of program execution. The

analysis of each data file consisted of the following two types of information:

1. The summary statistics of the regression analysis.

2. A plot of the line which would be predicted by the regression model vs. the data points in the data file (which were used to create the model). The graph axes were automatically scaled to the data.

Examples of the summary statistics and plots for each of the seven analysis types are included in Figures III-2 through III-8. Printouts of each of the seven computer programs (one for each model type) are included in Appendices A through G.

To select the "best" equation (i.e., regression model type), both the percent of variance accounted for and a visual examination of the predicted vs. actual points were considered. The visual examination helped to ensure that a model was a good predictor over the range of values of the independent variable. Additionally, the law of parsimony was considered in that if two models predicted equally well with respect to both variance and a review of the range of values, the least complex model was selected using the following order of complexity:

Least complex	Simple Regression
.	Exponential Regression
.	Inverse Exponential Regression
.	Squared Exponential Regression
.	Inverse Squared Exponential Regression
.	Polynomial Regression
Most complex	Multiple Regression

Finally, for those studies where a predictive model was presented, that model was incorporated into the moderator functions without changes.

Figure III-2

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR SIMPLE REGRESSION

```
HIT RETURN FOR PLOTINTERCEPT=.456426631
SLOPE=.0313858695

 R SQUARED=.924762238

MEAN OF X=3.3
MEAN OF Y=.56

EQUATION
.456426631 + .0313858695 * X
```

```
HIT RETURN FOR PLOT
```

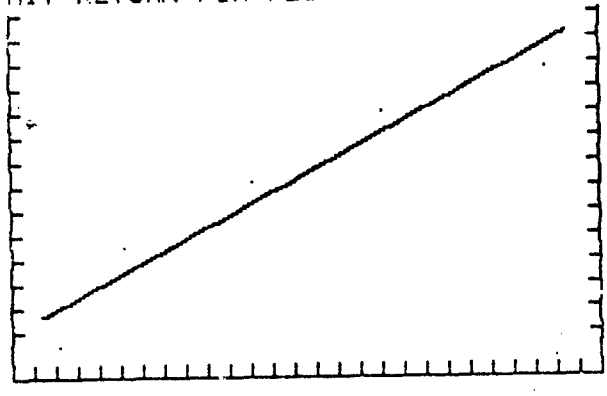


Figure III-3

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR EXPONENTIAL REGRESSION

```
ASSYMPTOTE=.17390625
SLOPE = .283110194
EXPONENT = .0851536923

 R SQUARED = .863426997

EQUATION
 .173 + .283 * EXP(.085 * X)
```

HIT RETURN FOR PLOT OF DATA AND FUNC

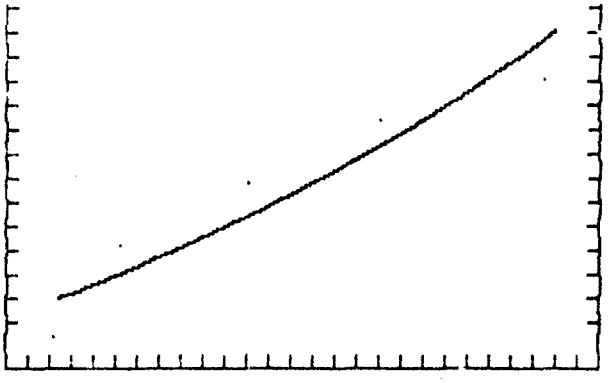


Figure III-4

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR INVERSE EXPONENTIAL REGRESSION

```
HIT RETURN FOR PLOT OF DATA AND FUNC
ASSYMPTOTE=13.75
SLOPE = -2.59312865
EXPONENT = .325059264


 R SQUARED = .996386943

EQUATION
 13.75 - 2.593 * EXP(.325 * X)
```

```
HIT RETURN FOR PLOT OF DATA AND FUNC
```

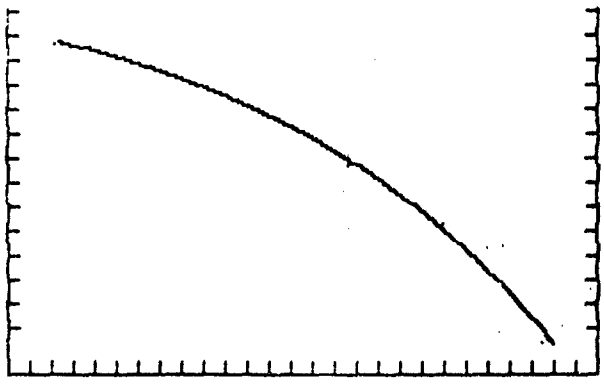


Figure III-5

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR SQUARED EXPONENTIAL REGRESSION

```
ASSYMPTOTE=-7.659375
SLOPE = 9.35758531
EXPONENT = .0197928057

 R SQUARED = .943536258

EQUATION
 -7.86 + 9.357 * EXP(.019 * X^2 )

HIT RETURN FOR PLOT OF DATA AND FUNC
```

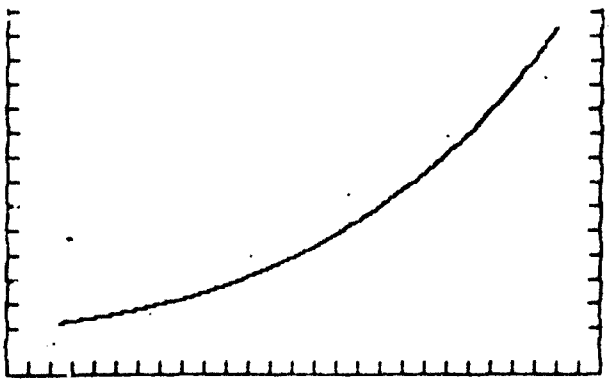


Figure III-6

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR
INVERSE SQUARED EXPONENTIAL REGRESSION

```
ASSYMPTOTE=1.9590625
SLOPE = -1.10631385
EXPONENT = 1.82153652E-03

 R SQUARED = .723429619

EQUATION
 1.959 - 1.106 * EXP(1E-03 * X^2)
```

HIT RETURN FOR PLOT OF DATA AND FUNC

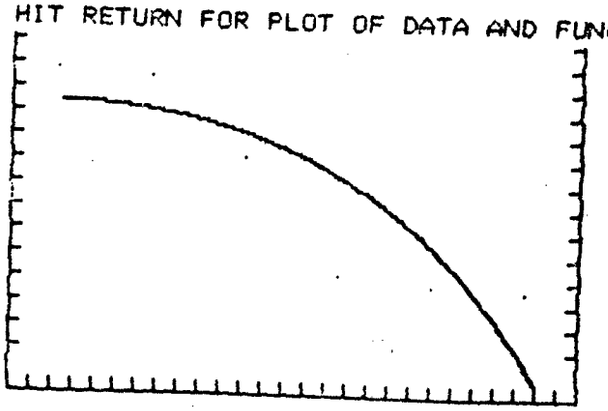


Figure III-7

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR POLYNOMIAL REGRESSION

```
        REGRESSION ANALYSIS

  B(0)=.433579061
  B(1)=.0583711891
  B(2)=-3.62761549E-03

 R SQUARED = .981618056

    ANOVA
 SS DUE TO Y=.0384794273
 MS DUE TO Y=.0192397137
 SS DUE TO ERROR=7.20572192E-04
 MS DUE TO ERROR=3.60286096E-04

    F(2,2)=299.04

 EQUATION
 .433 + .058 * X + -4E-03 * X^2

 HIT RETURN FOR PLOT
```

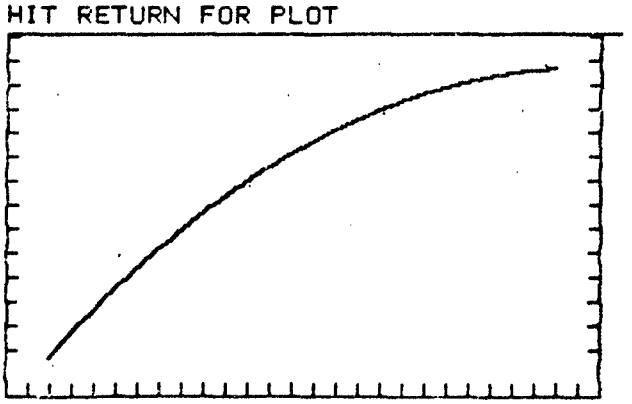


Figure III-8

EXAMPLE OF SUMMARY STATISTICS AND PLOTS FOR MULTIPLE REGRESSION

```
    REGRESSION ANALYSIS

 B(0)=-14.8842778
 B(1)=-2.59297686
 B(2)=4.31843572

R SQUARED = .975755838

  ANOVA
SS DUE TO Y=774.750136
MS DUE TO Y=387.375068
SS DUE TO ERROR=19.2498646
MS DUE TO ERROR=4.81246615

  F(2,4)=450.76

EQUATION
-14.885 + -2.593*X(1)) + 4.318*X(2))
```

3-5 Development and Testing of Moderator Functions

Once all of the equations had been prepared for each skill category, they had to be combined into FORTRAN subroutines which represented the moderator function for each skill category. Aside from simply including all the equations, two other issues had to be considered as well.

First, there was the issue of whether some equations should be weighted more than others. We immediately decided that there were no readily justifiable means for subjectively or objectively weighting the quality of each model (i.e., the equation). Therefore, we decided that the only weighting we would do would be based upon the number of independent variables in the model. For example, if one equation included three independent variables, then it would have three times more influence in moderating skill performance than an equation which included only one independent variable. This provided a roughly equal weighting for each independent variable.

Second, we had to ensure that the final moderated skill times or probabilities did not exceed legitimate bounds. Times had to be greater than zero and probabilities had to range between zero and one.

Once the FORTRAN subroutine was completed and loaded into the computer, debugging and sensitivity testing commenced. Sensitivity testing simply involved manipulating each of the independent variables used in a routine to determine if that moderated skill performance behaved reasonably. For example, if increased sleep deprivation led to improved performance, the odds were good that there was either a model development or coding error. As errors were identified, the software was reviewed and corrected.

Currently, all software has been debugged and sensitivity tests conducted. The next step would logically be a study of the predictive validity of one or several of the functions in an operational environment. Unfortunately,

neither the MOPADS operator computer models nor real-world data were available at the completion of this study for this analysis.

IV. ADVANTAGES AND DISADVANTAGES

The advantage of the approach described in the previous chapters is expected to be higher fidelity simulation. If assessment of human performance is one of the simulation objectives, this approach will allow more accurate simulation of performance under a variety of conditions. The model will behave more like humans behave than it would if static task performance parameters were used.

An ancillary advantage is that it does not require a human performance modeling expert. Once the task network is created, all that is required of the analyst is to specify the skills required for each task (from the predetermined list presented in Table I-1) and the proportion of each skill for task performance. Then, the skill moderator functions will be used to automatically manipulate task performance parameters. The skill moderator functions have been developed modularly so that they may be improved as the human factors profession gains more knowledge about quantitative human performance modeling. That is, as better data become available linking the independent variables to consequent skill performance, these data can be translated into models and these models can be incorporated into the moderator function software. The software is well documented in this regard.

The primary disadvantage is that the subroutines require the additional human factors independent variables to be included in the computer model. Endogenous variables will have to be updated during the simulation and exogenous variables will need to be specified. This can be a formidable task. The current moderator functions have been designed to accommodate default values so the analyst may "skip" some human factors variables, if desired. However, if the modeler wants improved fidelity, he or she must pay for it with the maintenance of additional variables. However, in any simulation, the modeler must be prepared to maintain the variables that he or she wishes to study.

The other current disadvantage is the absence of a predictive validation study. The skill moderator functions have internal validity in that they were developed from validated human performance models or data in the open literature. However, the "bottom line" of predictive validity is, as yet, undetermined.

V. REFERENCES

Baron, S., Kleinman, D. L., and Levison, W. H., An Optimal Control Model of Human Response. Automatica, Volume 6, 1970.

Kraiss, K. and Knaeuper, A., Using Visual Lobe Area Measurements to Predict Visual Lobe Performance. Human Factors, Volume 24 (6), 1982.

Laughery, K.R. and Gawron, V.J., MOPADS Report 5.10: MOPADS Moderator Function Subroutines. Draft Report prepared for the Army Research Institute, Ft. Bliss Field Unit, Ft. Bliss Texas. 1983

Laughery, K.R. and Polito, J., MOPADS Year 1 Progress Report. Report prepared for the Army Research Institute, Ft. Bliss Field Unit, Ft. Bliss, TX, 1982.

Laughery, K.R., A Data Base for Quantitative Human Performance Models. Report Prepared for the Army Research Institute, Ft. Bliss Field Unit, Ft. Bliss, TX, 1982.

Muralidharan, R., Baron, S., and Feehrer, C., A Decision, Monitoring, and Control Model of the Human Operator Applied to an RPV Control Problem. Air Force Office of Scientific Research Report AFOSR-TR-79-0675, Bolling AFB, D.C., March 1979.

Rouse, W., Systems Engineering Models of Human-Machine Interaction. North Holland Publishing; New York, New York, 1980.

Siegel, A.I., Pfeiffer, M.G., Kopstein, F.S., Wilson, L.G., and Ozkaptan, H., Human Performance in Continuous Systems. (ARI Research Product No. 808-4A), Applied Psychological Services, Inc., Wayne, PA, 1979. (NTIS No. AD-808-6131)

Wortman, D.B., Duket, S.D., Seifert, D.J., Hann, R.L., and Chubb, A.P., Using SAINT: A User Oriented Manual. Aerospace Medical Research Laboratory Report AMRL-TR-77-61, July 1978.

Appendix A

COMPUTER PROGRAM USED FOR SIMPLE REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
15  DIM D(2,200)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
     DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
     ND / 2:FL = 1: GOTO 500
50  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
 30 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
300  REM  BEGIN REGRESSION
510  IF FL = 0 THEN  INPUT "NUMBER OF DATA PAIRS ";N
520  FOR I = 1 TO N
530  IF FL = 0 THEN  PRINT " DATA PAIR #";I;" (IND VAR, DEP VAR)"
540  INPUT A,B:D(1,I) = A:D(2,I) = B
550 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A *
     B
560  NEXT I
565  PRINT D$;"CLOSE"
570 SX = A2 - (A1 * A1) / N
580 SY = B2 - (B1 * B1) / N
590 SP = AB - ((A1 * B1) / N)
600 A1 = A1 / N:B1 = B1 / N
610  HOME : PRINT "INTERCEPT=";B1 - (SP / SX) * A1
620 IC = B1 - (SP / SX) * A1
630  PRINT "SLOPE=";SP / SX
640 SL = SP / SX
650  PRINT : PRINT : PRINT " R SQUARED=";(SP * SP) / (SX * SY)
```

```
660  PRINT : PRINT "MEAN OF X=";A1: PRINT "MEAN OF Y=";B1
665  PRINT : PRINT "EQUATION"
666  PRINT IC;" + ";SL;" * X": PRINT
670  INPUT "HIT RETURN FOR PLOT";Q$
680  DEF  FN RG(WW) = IC + SL * WW
690 T7 = 270:05 = 150:01 = 1
700 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
710  FOR I = 1 TO N
720  PRINT D(1,I),D(2,I), FN RG(D(1,I))
730  IF D(1,I) > HX THEN HX = D(1,I)
740  IF D(1,I) < LX THEN LX = D(1,I)
750  IF D(2,I) < LY THEN LY = D(2,I)
760  IF D(2,I) > HY THEN HY = D(2,I)
770  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
780  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
790  NEXT I
800 LX = LX - 0.1 * (HX - LX)
810 HX = HX + 0.1 * (HX - LX)
820 LY = LY - 0.1 * (HY - LY)
830 HY = HY + 0.1 * (HY - LY)
840  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
850  FOR I = 1 TO 140 STEP 10
860  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
870  NEXT I
880  FOR I = 10 TO 270 STEP 10
890  HPLOT I,145 TO I,150
900  NEXT I
910 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
920  FOR I = 1 TO N
930  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
940  NEXT I
950  FOR I = 15 TO 255
960  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 - (
      FN RG(I / MX + LX) - LY) * MY
970  NEXT I
980 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
    100) / 100:HY =  INT (HY * 100) / 100
985  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
    X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
990  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ = "E" THEN  TEXT
    : END
1000  TEXT
1010  GOTO 610
```

Appendix B

COMPUTER PROGRAM USED FOR EXPONENTIAL REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
     DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
     ND / 2:FL = 1: GOTO 500
60  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
300 H =  - 9999:L = 9999
310  DIM D(2,500)
320  IF FL = 0 THEN  INPUT "NUMBER OF DATA PAIRS";N
330  FOR I = 1 TO N
340  IF FL = 0 THEN  PRINT " DATA PAIR #";I;" (IND VAR, DEP VAR)"
350  INPUT D(1,I),D(2,I)
360  IF D(2,I) > H THEN H = D(2,I)
370  IF D(2,I) < L THEN L = D(2,I)
380  NEXT I
385  PRINT D$;"CLOSE": HOME
400  DIM EX(5):CV = .001:SN = 25
410 IR = 1
420 HI = L:LI = 2 * L - H
430 EX(1) = 0
440 EX(5) = 0
445 SI = HI - LI
450 C = LI + .5 * SI
460  GOSUB 10000:EX(3) = R2
```

```
700 C = LI + .25 * SI: GOSUB 10000:EX(2) = R2
710 C = LI + .75 * SI: GOSUB 10000:EX(4) = R2
720 BV =  - 999
730  FOR I = 2 TO 4
740  IF EX(I) > BV THEN BS = I:BV = EX(I)
750  NEXT I
760 AX = EX(BS - 1):AY = EX(BS):AZ = EX(BS + 1)
770 EX(1) = AX:EX(3) = AY:EX(5) = AZ
790 IR = IR + 1
800 LI = LI + (BS - 2) * .25 * SI:SI = SI / 2.0
810  IF EX(3) - EX(1) < CV OR EX(3) - EX(5) < CV GOTO 1000
815  IF IR = SN GOTO 1000
820  GOTO 700
1000 C = LI + SI / 2
1100  FOR I = 1 TO N
1110 A = D(1,I):B =  LOG (D(2,I) - C)
1130 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A *
     B
1140  NEXT I
1150 SX = A2 - (A1 * A1) / N
1160 SY = B2 - (B1 * B1) / N
1170 SP = AB - ((A1 * B1) / N)
1180 A1 = A1 / N:B1 = B1 / N
1190 SL = B1 - (SP / SX) * A1
1200 SL =  EXP (SL)
1210 EX = SP / SX
1215  PRINT "NUMBER OF ITERATIONS=";IR - 1: PRINT
1220  PRINT : PRINT "ASSYMPTOTE=";C
1230  PRINT "SLOPE = ";SL: PRINT "EXPONENT = ";EX
1235 R2 = (SP * SP) / (SX * SY)
1240  PRINT : PRINT : PRINT " R SQUARED = ";R2
1245  PRINT : PRINT "EQUATION": PRINT " "; INT (C * 1000) / 1000;" + "; INT
     (SL * 1000) / 1000;" * EXP("; INT (EX * 1000) / 1000;" * X)": PRINT

1250  PRINT : INPUT "HIT RETURN FOR PLOT OF DATA AND FUNC";Q$
1260  DEF  FN RG(WW) = C + SL *  EXP (EX * WW)
1270 T7 = 270:O5 = 150:O1 = 1
1280 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
1290  FOR I = 1 TO N
1310  IF D(1,I) > HX THEN HX = D(1,I)
1320  IF D(1,I) < LX THEN LX = D(1,I)
1330  IF D(2,I) < LY THEN LY = D(2,I)
1340  IF D(2,I) > HY THEN HY = D(2,I)
1350  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1360  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1370  NEXT I
1380 LX = LX - 0.1 * (HX - LX)
1390 HX = HX + 0.1 * (HX - LX)
1400 LY = LY - 0.1 * (HY - LY)
1410 HY = HY + 0.1 * (HY - LY)
1420  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1430  FOR I = 1 TO 140 STEP 10
1440  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1450  NEXT I
```

```
1460  FOR I = 16 TO 270 STEP 10
1470  HPLOT I,145 TO I,150
1480  NEXT I
1490 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1500  FOR I = 1 TO N
1510  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1520  NEXT I
1530  FOR I = 25 TO 250
1540  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
     ( FN RG(I / MX + LX) - LY) * MY
1550  NEXT I
1560 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
     100) / 100:HY =  INT (HY * 100) / 100
1570  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
      X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
1580  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ = "E" THEN  TEXT
     : END
1590  TEXT : GOTO 1220
10000  FOR I = 1 TO N
10860 A = D(1,I):B =  LOG (D(2,I) - C)
10870 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A
      * B
10880  NEXT I
10890 SX = A2 - (A1 * A1) / N
10900 SY = B2 - (B1 * B1) / N
11000 SP = AB - ((A1 * B1) / N)
11500 R2 = (SP * SP) / (SX * SY)
12000 A1 = 0:A2 = 0:B1 = 0:B2 = 0:AB = 0
12020  RETURN
```

Appendix C

COMPUTER PROGRAM USED FOR INVERSE EXPONENTIAL REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
    DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
    ND / 2:FL = 1: GOTO 500
60  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
     DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO . - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
500 H =  - 9999:L = 9999
510  DIM D(2,500)
520  IF FL = 0 THEN  INPUT "NUMBER OF DATA PAIRS";N
530  FOR I = 1 TO N
540  IF FL = 0 THEN  PRINT " DATA PAIR #";I;" (IND VAR, DEP VAR)"
550  INPUT D(1,I),D(2,I)
560  IF D(2,I) > H THEN H = D(2,I)
570  IF D(2,I) < L THEN L = D(2,I)
580  NEXT I
585  PRINT D$;"CLOSE": HOME
600  DIM EX(5):CV = .001:SN = 25
610 IR = 1
620 LI = H:HI = 2 * H - L
630 EX(1) = 0
640 EX(5) = 0
645 SI = HI - LI
650 C = LI + .5 * SI
660  GOSUB 10000:EX(3) = R2
700 C = LI + .25 * SI: GOSUB 10000:EX(2) = R2
710 C = LI + .75 * SI: GOSUB 10000:EX(4) = R2
720 BV =  - 999
```

```
730  FOR I = 2 TO 4
740  IF EX(I) > BV THEN BS = I:BV = EX(I)
750  NEXT I
760 AX = EX(BS - 1):AY = EX(BS):AZ = EX(BS + 1)
770 EX(1) = AX:EX(3) = AY:EX(5) = AZ
790 IR = IR + 1
800 LI = LI + (BS - 2) * .25 * SI:SI = SI / 2.0
810  IF EX(3) - EX(1) < CV OR EX(3) - EX(5) < CV GOTO 1000
815  IF IR = SN GOTO 1000
820  GOTO 700
1000 C = LI + SI / 2
1100  FOR I = 1 TO N
1110 A = D(1,I):B =  LOG (C - D(2,I))
1130 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A *
     B
1140  NEXT I
1150 SX = A2 - (A1 * A1) / N
1160 SY = B2 - (B1 * B1) / N
1170 SP = AB - ((A1 * B1) / N)
1180 A1 = A1 / N:B1 = B1 / N
1190 SL = B1 - (SP / SX) * A1
1200 SL =  EXP (SL)
1210 EX = SP / SX
1215  PRINT "NUMBER OF ITERATIONS=";IR - 1: PRINT
1220  PRINT : PRINT "ASSYMPTOTE=";C
1230  PRINT "SLOPE = "; - SL: PRINT "EXPONENT = ";EX
1235 R2 = (SP * SP) / (SX * SY)
1240  PRINT : PRINT : PRINT " R SQUARED = ";R2
1245  PRINT : PRINT "EQUATION": PRINT " "; INT (C * 1000) / 1000;" - "; INT
     (SL * 1000) / 1000;" * EXP("; INT (EX * 1000) / 1000;" * X)": PRINT

1250  PRINT : INPUT "HIT RETURN FOR PLOT OF DATA AND FUNC";Q$
1260  DEF  FN RG(WW) = C - SL *  EXP (EX * WW)
1270 T7 = 270:05 = 150:01 = 1
1280 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
1290  FOR I = 1 TO N
1310  IF D(1,I) > HX THEN HX = D(1,I)
1320  IF D(1,I) < LX THEN LX = D(1,I)
1330  IF D(2,I) < LY THEN LY = D(2,I)
1340  IF D(2,I) > HY THEN HY = D(2,I)
1350  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1360  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1370  NEXT I
1380 LX = LX - 0.1 * (HX - LX)
1390 HX = HX + 0.1 * (HX - LX)
1400 LY = LY - 0.1 * (HY - LY)
1410 HY = HY + 0.1 * (HY - LY)
1420  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1430  FOR I = 1 TO 140 STEP 10
1440  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1450  NEXT I
1460  FOR I = 10 TO 270 STEP 10
1470  HPLOT I,145 TO I,150
1480  NEXT I
```

```
1490 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1500  FOR I = 1 TO N
1510  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1520  NEXT I
1530  FOR I = 25 TO 250
1540  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
     ( FN RG(I / MX + LX) - LY) * MY
1550  NEXT I
1560 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
     100) / 100:HY =  INT (HY * 100) / 100
1570  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
      X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
1580  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ = "E" THEN  TEXT
     : END
1590  TEXT : GOTO 1220
10000  FOR I = 1 TO N
 9860 A = D(1,I):B =  LOG (C - D(2,I))
 9870 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A
      * B
 9880  NEXT I
 9890 SX = A2 - (A1 * A1) / N
 9900 SY = B2 - (B1 * B1) / N
10000 SP = AB - ((A1 * B1) / N)
11500 R2 = (SP * SP) / (SX * SY)
12000 A1 = 0:A2 = 0:B1 = 0:B2 = 0:AB = 0
12020  RETURN
```

Appendix D

COMPUTER PROGRAM USED FOR SQUARED EXPONENTIAL REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
     DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
     ND / 2:FL = 1: GOTO 500
60  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
500 H =  - 9999:L = 9999
510  DIM D(2,500)
520  IF FL = 0 THEN  INPUT "NUMBER OF DATA PAIRS";N
530  FOR I = 1 TO N
540  IF FL = 0 THEN  PRINT " DATA PAIR #";I;" (IND VAR, DEP VAR)"
550  INPUT D(1,I),D(2,I)
560  IF D(2,I) > H THEN H = D(2,I)
570  IF D(2,I) < L THEN L = D(2,I)
580  NEXT I
585  PRINT D$;"CLOSE": HOME
600  DIM EX(5):CV = .001:SN = 25
610 IR = 1
620 HI = L:LI = 2 * L - H
630 EX(1) = 0
640 EX(5) = 0
645 SI = HI - LI
650 C = LI + .5 * SI
660  GOSUB 10000:EX(3) = R2
690 C = LI + .25 * SI: GOSUB 10000:EX(2) = R2
710 C = LI + .75 * SI. GOSUB 10000:EX(4) = R2
```

```
720 BV =  - 999
730  FOR I = 2 TO 4
740  IF EX(I) > BV THEN BS = I:BV = EX(I)
750  NEXT I
760 AX = EX(BS - 1):AY = EX(BS):AZ = EX(BS + 1)
770 EX(1) = AX:EX(3) = AY:EX(5) = AZ
790 IR = IR + 1
800 LI = LI + (BS - 2) * .25 * SI:SI = SI / 2.0
810  IF EX(3) - EX(1) < CV OR EX(3) - EX(5) < CV GOTO 1000
815  IF IR = SN GOTO 1000
820  GOTO 700
1000 C = LI + SI / 2
1100  FOR I = 1 TO N
1110 A = D(1,I) * D(1,I):B =  LOG (D(2,I) - C)
1130 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A *
     B
1140  NEXT I
1150 SX = A2 - (A1 * A1) / N
1160 SY = B2 - (B1 * B1) / N
1170 SP = AB - ((A1 * B1) / N)
1180 A1 = A1 / N:B1 = B1 / N
1190 SL = B1 - (SP / SX) * A1
1200 SL =  EXP (SL)
1210 EX = SP / SX
1215  PRINT "NUMBER OF ITERATIONS=";IR - 1: PRINT
1220  PRINT : PRINT "ASSYMPTOTE=";C
1230  PRINT "SLOPE = ";SL: PRINT "EXPONENT = ";EX
1235 R2 = (SP * SP) / (SX * SY)
1240  PRINT : PRINT : PRINT " R SQUARED = ";R2
1245  PRINT : PRINT "EQUATION": PRINT " "; INT (C * 1000) / 1000;" + "; INT
     (SL * 1000) / 1000;" * EXP("; INT (EX * 1000) / 1000;" * X^2 )": PRINT

1250  PRINT : INPUT "HIT RETURN FOR PLOT OF DATA AND FUNC";Q$
1260  DEF  FN RG(WW) = C + SL *  EXP (EX * WW * WW)
1270 T7 = 270:05 = 150:01 = 1
1280 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
1290  FOR I = 1 TO N
1310  IF D(1,I) > HX THEN HX = D(1,I)
1320  IF D(1,I) < LX THEN LX = D(1,I)
1330  IF D(2,I) < LY THEN LY = D(2,I)
1340  IF D(2,I) > HY THEN HY = D(2,I)
1350  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1360  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1370  NEXT I
1380 LX = LX - 0.1 * (HX - LX)
1390 HX = HX + 0.1 * (HX - LX)
1400 LY = LY - 0.1 * (HY - LY)
1410 HY = HY + 0.1 * (HY - LY)
1420  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1430  FOR I = 1 TO 140 STEP 10
1440  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1450  NEXT I
1460  FOR I = 10 TO 270 STEP 10
1470  HPLOT I,145 TO I,150
```

```
1480  NEXT I
1490 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1500  FOR I = 1 TO N
1510  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1520  NEXT I
1530  FOR I = 25 TO 250
1540  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
     ( FN RG(I / MX + LX) - LY) * MY
1550  NEXT I
1560 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
     100) / 100:HY =  INT (HY * 100) / 100
1570  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
      X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
1580  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ = "E" THEN  TEXT
     : END
1590  TEXT : GOTO 1220
10000  FOR I = 1 TO N
10860 A = D(1,I) * D(1,I):B =  LOG (D(2,I) - C)
10870 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A
      * B
10880  NEXT I
10890 SX = A2 - (A1 * A1) / N
10900 SY = B2 - (B1 * B1) / N
11000 SP = AB - ((A1 * B1) / N)
11500 R2 = (SP * SP) / (SX * SY)
12000 A1 = 0:A2 = 0:B1 = 0:B2 = 0:AB = 0
12020  RETURN
```

Appendix E

COMPUTER PROGRAM USED FOR INVERSE SQUARED EXPONENTIAL REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
     DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
     ND / 2:FL = 1: GOTO 500
60  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = C: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
500 H =  - 9999:L = 9999
510  DIM D(2,500)
520  IF FL = 0 THEN  INPUT "NUMBER OF DATA PAIRS";N
530  FOR I = 1 TO N
540  IF FL = 0 THEN  PRINT " DATA PAIR #";I;" (IND VAR, DEP VAR)"
550  INPUT D(1,I),D(2,I)
560  IF D(2,I) > H THEN H = D(2,I)
570  IF D(2,I) < L THEN L = D(2,I)
580  NEXT
585  PRINT D$;"CLOSE": HOME
600  DIM EX(5):CV = .001:SN = 25
610 IR = 1
620 LI = H:HI = 2 * H - L
630 EX(1) = 0
640 EX(5) = 0
645 SI = HI - LI
650 C = LI + .5 * SI
660  GOSUB 10000:EX(3) = R2
700 C = LI + .25 * SI: GOSUB 10000:EX(2) = R2
710 C = LI + .75 * SI: GOSUB 10000:EX(4) = R2
720 BV =  - 999
```

```
730  FOR I = 2 TO 4
740  IF EX(I) > BV THEN BS = I:BV = EX(I)
750  NEXT I
760 AX = EX(BS - 1):AY = EX(BS):AZ = EX(BS + 1)
770 EX(1) = AX:EX(3) = AY:EX(5) = AZ
780 IR = IR + 1
900 LI = LI + (BS - 2) * .25 * SI:SI = SI / 2.0
810  IF EX(3) - EX(1) < CV OR EX(3) - EX(5) < CV GOTO 1000
315  IF IR = SN GOTO 1000
820  GOTO 700
1000 C = LI + SI / 2
1100  FOR I = 1 TO N
1110 A = D(1,I) * D(1,I):B =  LOG (C - D(2,I))
1130 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A *
     B
1140  NEXT I
1150 SX = A2 - (A1 * A1) / N
 160 SY = B2 - (B1 * B1) / N
 170 SP = AB - ((A1 * B1) / N)
1180 A1 = A1 / N:B1 = B1 / N
1190 SL = B1 - (SP / SX) * A1
1200 SL =  EXP (SL)
 210 EX = SP / SX
 215  PRINT "NUMBER OF ITERATIONS=";IR - 1: PRINT
1220  PRINT : PRINT "ASSYMPTOTE=";C
1230  PRINT "SLOPE = "; - SL: PRINT "EXPONENT = ";EX
1235 R2 = (SP * SP) / (SX * SY)
1240  PRINT : PRINT : PRINT " R SQUARED = ";R2
1245  PRINT : PRINT "EQUATION": PRINT " "; INT (C * 1000) / 1000;" - "; INT
     (SL * 1000) / 1000;" * EXP("; INT (EX * 1000) / 1000;" * X^2)": PRINT

1250  PRINT : INPUT "HIT RETURN FOR PLOT OF DATA AND FUNC";Q$
1260  DEF  FN RG(WW) = C - SL *  EXP (EX * WW * WW)
1270 T7 = 270:05 = 150:01 = 1
1280 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
1290  FOR I = 1 TO N
1310  IF D(1,I) > HX THEN HX = D(1,I)
1320  IF D(1,I) < LX THEN LX = D(1,I)
1330  IF D(2,I) < LY THEN LY = D(2,I)
1340  IF D(2,I) > HY THEN HY = D(2,I)
1350  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1360  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1370  NEXT I
1380 LX = LX - 0.1 * (HX - LX)
1390 HX = HX + 0.1 * (HX - LX)
1400 LY = LY - 0.1 * (HY - LY)
1410 HY = HY + 0.1 * (HY - LY)
1420  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1430  FOR I = 1 TO 140 STEP 10
1440  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1450  NEXT I
1460  FOR I = 10 TO 270 STEP 10
1470  HPLOT I,145 TO I,150
```

```
1480  NEXT I
1490 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1500  FOR I = 1 TO N
1510  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1520  NEXT I
1530  FOR I = 25 TO 250
1540  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
      ( FN RG(I / MX + LX) - LY) * MY
1550  NEXT I
1560 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
      100) / 100:HY =  INT (HY * 100) / 100
1570  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
      X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
1580  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ = "E" THEN  TEXT
      : END
1590  TEXT : GOTO 1220
2000  FOR I = 1 TO N
2060 A = D(1,I) * D(1,I):B =  LOG (C - D(2,I))
2070 A1 = A1 + A:A2 = A2 + A * A:B1 = B1 + B:B2 = B2 + B * B:AB = AB + A
      * B
2080  NEXT I
2090 SX = A2 - (A1 * A1) / N
2100 SY = B2 - (B1 * B1) / N
2000 SP = AB - ((A1 * B1) / N)
1500 R2 = (SP * SP) / (SX * SY)
2000 A1 = 0:A2 = 0:B1 = 0:B2 = 0:AB = 0
2020  RETURN
```

Appendix F

COMPUTER PROGRAM USED FOR POLYNOMIAL REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
     DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
     ND / 2:FL = 1: GOTO 500
50  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "IND VARS ARE ODD ENTRIES": PRINT "DEP VARS ARE EVEN ENTRIES"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
500  DIM D(2,500)
510  PRINT :D$ =  CHR$ (4)
520  IF FL = 0 THEN  INPUT "# DATA GROUPS";N
530  FOR I = 1 TO N
540  IF FL = 0 THEN  PRINT "INPUT DATA GROUP ";I;" (IND VAR, DEP VAR)"
550  INPUT A,C:B = A * A:D(1,I) = A:D(2,I) = C
560 X = X + A:Y = Y + B:Z = Z + C
570 X2 = X2 + A * A:Y2 = Y2 + B * B:Z2 = Z2 + C * C
580 XY = XY + A * B:XZ = XZ + A * C:YZ = YZ + B * C
590  NEXT I
600  PRINT D$;"CLOSE": HOME
610 DT = N * X2 * Y2 + X * XY * Y * 2 - Y * X2 * Y - N * XY * XY - X * X *
     Y2
620  DIM XI(3,3),YM(3),BM(3)
630 YM(1) = Z:YM(2) = XZ:YM(3) = YZ
640 XI(1,1) = X2 * Y2 - XY * XY
650 XI(1,2) =  - (X * Y2 - XY * Y)
660 XI(1,3) = X * XY - X2 * Y
670 XI(2,2) = N * Y2 - Y * Y
680 XI(2,3) =  - (N * XY - X * Y)
```

```
690 XI(3,3) = N * X2 - X * X
700 XI(2,1) = XI(1,2)
710 XI(3,1) = XI(1,3)
720 XI(3,2) = XI(2,3)
730  FOR I = 1 TO 3: FOR J = 1 TO 3:XI(I,J) = XI(I,J) / DT: NEXT J: NEXT
     I
740  FOR I = 1 TO 3
750  FOR J = 1 TO 3:BM(I) = BM(I) + XI(I,J) * YM(J): NEXT J
760  NEXT I
770  FOR I = 1 TO 3:ZH = ZH + BM(I) * YM(I): NEXT I
780 R = 1 - (Z2 - ZH) / (Z2 - (Z * Z) / N)
790  HOME
800  VTAB 5: PRINT "    REGRESSION ANALYSIS"
810  PRINT
820  FOR I = 1 TO 3
830  PRINT " B(";I - 1;")=";BM(I)
840  NEXT I
850  PRINT : PRINT "R SQUARED = ";R
860 SY = R * (Z2 - (Z * Z) / N)
870 SE = Z2 - ZH
880  PRINT : PRINT "  ANOVA "
890  PRINT "SS DUE TO Y=";SY
900  PRINT "MS DUE TO Y=";SY / 2
910  PRINT "SS DUE TO ERROR=";SE
920  PRINT "MS DUE TO ERROR=";SE / (N - 3)
930  PRINT : PRINT "  F(2,";N - 3;")="; INT (((SY / 2) / (SE / (N - 3))) *
     560) / 100
932  PRINT : PRINT "EQUATION"
934  PRINT  INT (BM(1) * 1000) / 1000;" + "; INT (BM(2) * 1000) / 1000;"
     * X + "; INT (BM(3) * 1000) / 1000;" * X^2"
940  INPUT "HIT RETURN FOR PLOT";Q$
950  DEF  FN RG(WW) = BM(1) + BM(2) * WW + BM(3) * WW * WW
960 T7 = 270:05 = 150:01 = 1
970 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
979  FOR I = 1 TO N
990  PRINT D(1,I),D(2,I), FN RG(D(1,I))
1000  IF D(1,I) > HX THEN HX = D(1,I)
1010  IF D(1,I) < LX THEN LX = D(1,I)
1020  IF D(2,I) < LY THEN LY = D(2,I)
1030  IF D(2,I) > HY THEN HY = D(2,I)
1040  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1050  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1060  NEXT I
1070 LX = LX - 0.1 * (HX - LX)
1080 HX = HX + 0.1 * (HX - LX)
1090 LY = LY - 0.1 * (HY - LY)
1100 HY = HY + 0.1 * (HY - LY)
1110  HGR : HCOLOR= 3: HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1120  FOR I = 1 TO 140 STEP 10
1130  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1140  NEXT I
1150  FOR I = 10 TO 270 STEP 10
1160  HPLOT I,145 TO I,150
1170  NEXT I
```

```
1180 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1190  FOR I = 1 TO N
1200  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1210  NEXT I
1220  FOR I = 20 TO 250
1230  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
     ( FN RG(I / MX + LX) - LY) * MY
1240  NEXT I
1250 LX =  INT (LX * 100) / 100:LY =  INT (LY * 100) / 100:HX =  INT (HX *
     100) / 100:HY =  INT (HY * 100) / 100
1255  VTAB 21: PRINT "FILE=";DS$: PRINT "LOW X VALUE=";LX; TAB( 21);"HIGH
      X VALUE=";HX: PRINT "LOW Y VALUE=";LY; TAB( 21);"HIGH Y VALUE=";HY
1260  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$:
1270  TEXT
21260  INPUT "HIT RETURN TO CONTINUE, 'E' TO END";QW$: IF QW$ <  > "E" THEN
      TEXT : GOTO 800
```

Appendix G

COMPUTER PROGRAM USED FOR MULTIPLE REGRESSION

```
5  LOMEM: 16385
10 D$ =  CHR$ (4)
20  PRINT "ENTER NAME OF DATA BASE FOR ANALYSIS": PRINT " OR 'RETURN' IF
    DATA ARE TO BE ": PRINT " ENTERED MANUALLY"
30  INPUT DS$: IF  LEN (DS$) < 1 GOTO 500
35  ONERR  GOTO 60
40  PRINT D$;"OPEN ";DS$: PRINT D$;"READ ";DS$: INPUT ND: POKE 216,0:N =
    ND / 3:FL = 1: GOTO 500
60  IF  PEEK (222) <  > 5 THEN  STOP
70  POKE 216,0
80  PRINT : PRINT "THIS DATA SET DOES NOT CURRENTLY EXIST": PRINT " ENTER
      DATA TO CREATE IT OR HIT": PRINT " 'RETURN' TO EXIT THE PROGRAM"
90 I = 0: DIM A$(500)
100  PRINT
110  PRINT " TYPE '$$' TO QUIT"
120  PRINT " TYPE '$B' TO BACKUP"
125  PRINT "ORDER-IND VAR 1, IND VAR 2, DEP VAR"
130 I = I + 1
140  PRINT "INPUT #";I
150  INPUT A$(I)
155  IF I = 1 AND A$(I) = "" THEN  PRINT D$;"DELETE ";DS$: STOP
160  IF A$(I) = "$B" THEN I = I - 1: GOTO 140
170  IF A$(I) = "$$" GOTO 200
180  GOTO 100
190  PRINT
200  PRINT D$;"OPEN ";DS$
210  PRINT D$;"WRITE ";DS$
220  PRINT I - 1
230  FOR J = 1 TO I - 1
240  PRINT A$(J)
250  NEXT J
260  PRINT D$;"CLOSE ";DS$
270  GOTO 40
500  DIM D(2,500)
510  PRINT :D$ =  CHR$ (4)
520  IF FL = 0 THEN  PRINT "INPUT DATA GROUP ";I
530  FOR I = 1 TO N
540  PRINT "INPUT DATA GROUP ";I:
550  INPUT A,B,C
560 X = X + A:Y = Y + B:Z = Z + C
570 X2 = X2 + A * A:Y2 = Y2 + B * B:Z2 = Z2 + C * C
580 XY = XY + A * B:XZ = XZ + A * C:YZ = YZ + B * C
590  NEXT I
600  PRINT D$;"CLOSE"
610 DT = N * X2 * Y2 + X * XY * Y * 2 - Y * X2 * Y - N * XY * XY - X * X *
    Y2
620  DIM XI(3,3),YM(3),BM(3)
630 YM(1) = Z:YM(2) = XZ:YM(3) = YZ
640 XI(1,1) = X2 * Y2 - XY * XY
```

```
650 XI(1,2) =  - (X * Y2 - XY * Y)
660 XI(1,3) = X * XY - X2 * Y
670 XI(2,2) = N * Y2 - Y * Y
680 XI(2,3) =  - (N * XY - X * Y)
690 XI(3,3) = N * X2 - X * X
700 XI(2,1) = XI(1,2)
710 XI(3,1) = XI(1,3)
720 XI(3,2) = XI(2,3)
730  FOR I = 1 TO 3: FOR J = 1 TO 3:XI(I,J) = XI(I,J) / DT: NEXT J: NEXT
     I
740  FOR I = 1 TO 3
750  FOR J = 1 TO 3:BM(I) = BM(I) + XI(I,J) * YM(J): NEXT J
760  NEXT I
770  FOR I = 1 TO 3:ZH = ZH + BM(I) * YM(I): NEXT I
780 R = 1 - (Z2 - ZH) / (Z2 - (Z * Z) / N)
790  HOME
800  VTAB 5: PRINT "    REGRESSION ANALYSIS"
810  PRINT
820  FOR I = 1 TO 3
830  PRINT " B(";I - 1;")=";BM(I)
840  NEXT I
850  PRINT : PRINT "R SQUARED = ";R
860 SY = R * (Z2 - (Z * Z) / N)
870 SE = Z2 - ZH
880  PRINT : PRINT "  ANOVA "
890  PRINT "SS DUE TO Y=";SY
900  PRINT "MS DUE TO Y=";SY / 2
910  PRINT "SS DUE TO ERROR=";SE
920  PRINT "MS DUE TO ERROR=";SE / (N - 3)
930  PRINT : PRINT "  F(2,";N - 3;")="; INT (((SY / 2) / (SE / (N - 3))) *
     560) / 100
932  PRINT : PRINT "EQUATION"
934  PRINT  INT (BM(1) * 1000) / 1000;" + "; INT (BM(2) * 1000) / 1000;"*
     X(1)) + "; INT (BM(3) * 1000) / 1000;"*X(2))"
940  END
950  DEF  FN RG(WW) = BM(1) + BM(2) * WW + BM(3) * WW * WW
960 T7 = 270:D5 = 150:D1 = 1
970 LX = 99999:LY = 99999:HX =  - 99999:HY =  - 99999
980  FOR I = 1 TO N
990  PRINT D(1,I),D(2,I), FN RG(D(1,I))
1000  IF D(1,I) > HX THEN HX = D(1,I)
1010  IF D(1,I) < LX THEN LX = D(1,I)
1020  IF D(2,I) < LY THEN LY = D(2,I)
1030  IF D(2,I) > HY THEN HY = D(2,I)
1040  IF  FN RG(D(1,I)) > HY THEN HY =  FN RG(D(1,I))
1050  IF  FN RG(D(1,I)) < LY THEN LY =  FN RG(D(1,I))
1060  NEXT I
1070 LX = LX - 0.1 * (HX - LX)
1080 HX = HX + 0.1 * (HX - LX)
1090 LY = LY - 0.1 * (HY - LY)
1100 HY = HY + 0.1 * (HY - LY)
1110  HGR : HCOLOR= 3. HPLOT 0,0 TO 0,150 TO 270,150 TO 270,0
1120  FOR I = 1 TO 140 STEP 10
```

```
1130  HPLOT 0,I TO 5,I: HPLOT 265,I TO 270,I
1140  NEXT I
1150  FOR I = 10 TO 270 STEP 10
1160  HPLOT I,145 TO I,150
1170  NEXT I
1180 MX = 270 / (HX - LX):MY = 150 / (HY - LY)
1190  FOR I = 1 TO N
1200  HPLOT (D(1,I) - LX) * MX,150 - (D(2,I) - LY) * MY
1210  NEXT I
1220  FOR I = 20 TO 250
1230  HPLOT I - 01,05 - ( FN RG((I - 01) / MX + LX) - LY) * MY TO I,05 -
     ( FN RG(I / MX + LX) - LY) * MY
1240  NEXT I
1250  VTAB 21: PRINT "LX=";LX;" LY=";LY: PRINT "MX=";MX;" MY=";MY
1260  INPUT "HIT RETURN TO CONTINUE";QW$
1270  TEXT
```

APPENDIX O

MOPADS FINAL REPORT:

MOPADS TASK SEQUENCING STRUCTURE

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

I. INTRODUCTION

Operators modeled by MOPADS perform rule-based activities called "operator tasks." These are checklist actions consisting of a) elemental (at least as far as MOPADS is concerned) actions such as entering numbers on a keyboard, and b) simple decisions. The operators memorize the procedures for these tasks.

A different sort of activity is required by the operators when they decide which task to perform. Such decisions by the operators represent tactical decisions required to accomplish their missions. These decisions are influenced by the current tactical situation, standard operating procedures, the operator's experience, and the operators personal motivations. It is necessary to model this "knowledge based" activity, because the MOPADS simulated operators must sequence these tasks in such a way that they respond realistically to the air battle. This report describes the model used by MOPADS for operator "task sequencing."

The intended audience is the MOPADS modeler who will implement or modify the task sequencing procedures for an air defense system module. MOPADS users need to be familiar with the material, but a less detailed discussion is contained in Polito (1983a). The system module specific details which instantiate this procedure for a particular system module are contained in the user and reference documents for that module, e.g., Goodin & Polito (1983a); Goodin & Walker (1983a).

Section II discusses the issues in selecting a task sequencing methodology for MOPADS. The selected algorithm is presented in Section III, and the implementation details are discussed in Section IV. A brief example of specifying the data and preparing subprograms is given in Section V.

II. FACTORS IN SELECTING A TASK SEQUENCING METHOD FOR MOPADS

1-0 DESIRABLE FEATURES OF A TASK SEQUENCING METHOD

In selecting a method to sequence MOPADS operator tasks, several objectives were considered. Each of them is discussed below.

1. Causal. The methodology must be causal. In other words, the operator should sequence his/her tasks in response to the air battle and other pertinent tactical scenario issues. This consideration immediately ruled out simple sequencing rules such as random or cyclic sequencing.

2. Consistent with Established Theory. The selected method should have been considered as a reasonable decision making model by researchers. This consideration lead to examination of utility theory and goal seeking approaches.

3. Computationally Attractive. The method should not require frequent accomplishment of extensive numerical procedures such as curve fitting or solution of simultaneous equations.

4. Intuitively Meaningful. The data required from the user should be meaningful to an air defense specialist. In other words, abstract or derived parameters should not be required.

5. Economical in Terms of Data Requirements. The method should not require unusually large amounts of data.

The above considerations represent an ideal profile against which candidate methods were compared.

2-0 CANDIDATE TASK SEQUENCING METHODOLOGIES

As stated above, utility theory and goal seeking methods were considered in order to satisfy objective (2) above. Both of these methods satisfy the need for causality, since the evaluation of a particular utility function or set of goals can be based upon parameters derived from the current state of the air battle and environment. Both techniques have a body of literature suggesting that humans make decisions in a way to optimize a utility function or to satisfy goals.

There can be a substantial difference in the computational effort required to implement the two methods. To use the utility

theory approach we would define a function U(S) which maps the multidimensional vector S onto the non-negative real numbers. The vector S represents the state or attributes of the air defense system for a particular operator. Once U(S) is known, the steps in selecting the next task are simple:

i) Evaluate $U_o = U(S_o)$ for the current system state, S_o

ii) If alternatives a, b, c, etc. are available to the operator, estimate the state vectors, S_a, S_b, S_c, etc., that will result from selecting each alternative

iii) Evaluate $U_a = U(S_a)$, $U_b = U(S_b)$, etc. and select the alternative that most improves the operator's utility.

The difficulty with this method results from determination of the utility function, U. To do this in its most general form would require obtaining values of utility at a large number of points and developing the function U(S) from some curve fitting technique such as linear regression. Presumably, such data would be gathered from more than one subject; also this procedure would have to be repeated for each operator type since we would not expect a single utility function to apply to the AN/TSQ-73 Tactical Director and to the IHAWK Fire Control Operator. Since the domain of S is a continuum, the utility values obtained from a subject will, of necessity, be cardinal values. These will have to be normalized in some way to make the data from several subjects comparable.

In addition to these considerations, it is desirable to provide at least the potential of dynamically altering an operator's decision making structure during the course of the simulation. This would represent an operator who changes the relative importance of his/her objectives based upon the state of his current situation. It can be argued that this type of behavior can be embodied in the utility function by expanding the set of independent variables. While this is theoretically true, from a practical standpoint, a set of parameters whose values change discretely with changing conditions is easier to implement. This is true because the data collection and reduction problem discussed above would be greatly expanded if the effects on goals of stress, fatigue, etc. had also to be estimated. A more practical approach is to attempt to parameterize the utility functions with these variables and to re-compute the coefficients of the utility function as conditions change.

Any implementation of this dynamic capability with traditional utility functions will involve extensive data collection and/or computation. It could conceivably require re-computation of utility function coefficients during the course of a simulation which would correspondingly increase run time. In short, it would not be

economical in terms of data requirements or computational needs to implement this capability with utility functions.

Finally, it is desirable to have an individual editing capability for operators that is analogous to the editing features for other human factors parameters. Each simulated operator in a MOPADS simulation can be individually edited so that his/her responses need not be identical to those for a nominal operator. A similar capability for task sequencing should be available, so the MOPADS user can test the effects of variations in the operator's objectives.

In order to do this, parameters must be available for the MOPADS user to edit. These parameters should have intuitive meaning to the user. Therefore, the ability to alter coefficients in the utility function (which may have been determined by, say, linear regression) is really not sufficient. On the other hand, altering the data from which the utility function is determined requires that the function coefficients be re-computed and also may lead to unintended effects on the operators utility function. In other words, it may be difficult for the user to predict the impact of certain changes on the resulting utility function.

The other major alternative, a goal seeking procedure, can be implemented in such a way so as to satisfy most of the goals in 1-0 above. Suppose an operator has a set of (not necessarily independent) goals, G_1, G_2, G_3, etc., each of which has an associated goal state. For example, if the goal is to maximize the distance of enemy aircraft to any protected site, then the goal state is the minimum distance of any hostile aircraft to any protected site. Each goal state is a function of certain variables whose values are known. There is a subset of the range of each goal state in which the goal is satisfied. When the goal state is not in this subset, it is to some degree unsatisfied.

The operator must be able to assign a degree-of-dissatisfaction to unsatisfied goals which is a function of its goal state. He then identifies his most dissatisfied goal(s) and selects a task which will improve his/her level of satisfaction. The steps in this process are:

i) Evaluate each goal state $G_1(S_o)$, $G_2(S_o)$, $G_3(S_o)$, at the current set of system variables, S_o.

ii) If alternatives a, b, c, etc., are available to the operator, estimate the system variables which will result from each alternative, S_a, S_b, S_c, etc.

iii) Evaluate the goal states expected to result from each alternative, $G_1(S_a)$, $G_2(S_b)$, etc.

iv) Select the alternative that most improves the operators level of dissatisfaction with the expected goal states.

Now, there are many ways to implement a procedure like that above. Not all of them would necessarily be less complex than the general utility function method. There are several aspects of the goal seeking approach, however, that give it the potential of satisfying the objectives of 1-0 above.

First, the operators' objectives are stated as a discrete set of goals. It is not necessary that the goals be independent. In other words, the sets of variables used to evaluate the goal states need not be mutually exclusive. This makes the 'state space," over which data must be collected from subject matter experts, discrete rather than continuous. Thus, the experts' estimation of the relative dissatisfaction (called "goal priorities" in the next section) of two or more goals can be obtained by pairwise comparisions. These comparisons can be made by considering only a limited set of variables (i.e., "other things equal"). This contrasts with the utility function approach in which a utility value must be assigned (or derived) which is, at least in theory, a function of all of the state variables.

Furthermore, when comparing two goals, it is necessary only to rank the goals for various values of their goal states rather than to assign cardinal utility values. Naturally, eventually some (arbitrary) scale of goal dissatisfaction ("goal priority") must be established, but since the rankings are the most significant aspects, it is easier to develop a scale that resolves the rankings of more than one expert with the goal approach than with the utility method.

Thus, the data requirements are reduced when using a goal seeking approach. Also, the computational burden of the method need not be excessive. The computation of the goal priorities can be simplified by using a restricted, but flexible class of functional forms (this will be discussed in Section III below) which yield a goal priority from a goal state. Since it is these goal priority functions, not the goal states, which are parameterized, intuitive editing of operator task sequencing parameters is possible.

Another way to say this is that while the goal states may be complex functions of the state of the system, the priority (or dissatisfaction level) of the goal is a function of only the computed goal state. Therefore, the goal priority is a function only of a single variable, and, since an operator's goal seeking behavior is dependent on the priorities of the goals, this behavior can be altered by editing the priority functions only.

To illustrate, consider Figure II-1. Suppose an operator has two goals, A and B. The goal priority functions for these goals are labelled A_1 and B in Figure II-1. The goal states for these goals run along the horizontal axis and the goal priorities are represented on the vertical axis. At a goal state of y, the line B is greater than the line A_1 so the operator assigns a higher

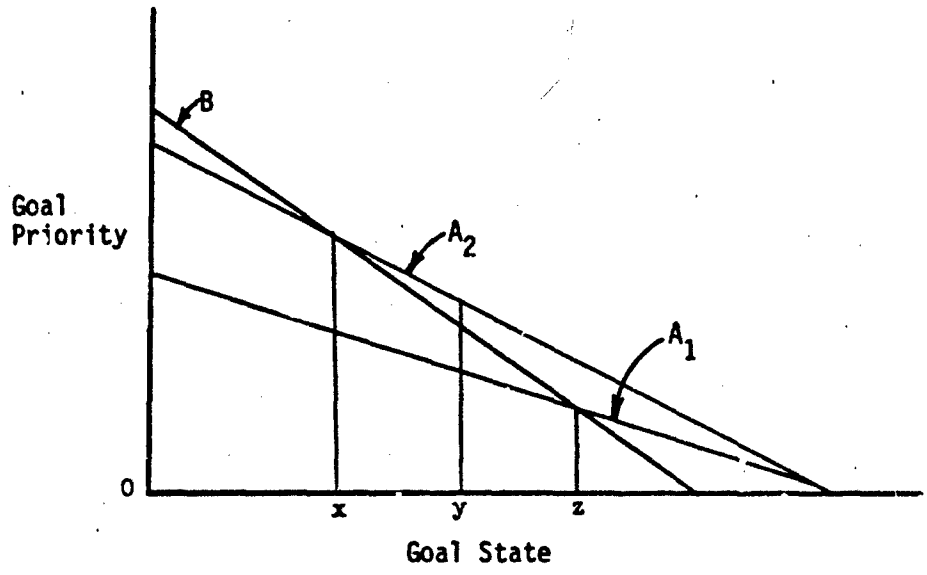


Figure II-1. Illustration of Goal Priority Function Editing.

priority to goal B than to goal A at this value of the goal state. The situation is reversed for goal states greater than z. For goal states less than z, the operator will select actions that improve goal B. This behavior can be modified by altering the goal priority functions only. Suppose now that the goal priority function for goal A is line A_2 in Figure II-1. Now goal B is most important only for goal states less than x. For values greater than x, the operator selects actions to improve goal A. Note that this change in behavior was accomplished without modification to the definitions for the goals or the method of computation of the goal states. Also, modifying the goal priority function from A_1 to A_2 is relatively intuitive since the impact of the change is readily apparent.

A goal seeking method based on this discussion was selected and is presented in greater detail in the next section. In fact, the goal seeking approach is a special case of the general utility theory ideas discussed earlier. This correspondence between the utility and goal methods will also be shown at the appropriate time in the discussion that follows.

III. THE TASK SEQUENCING ALGORITHM

1-0 GOAL PRIORITY FUNCTIONS

The goal seeking behavior of the operators is characterized by the following:

1. The operators have a set of goals which they desire to satisfy simultaneously. They are capable of determining the value or state of each goal. For example, the operator may desire to maximize the distance from a critical asset to any hostile aircraft. The goal state is the minimum distance to any hostile track.

2. The operators can rank the importance of their goal states. In other words, they can assign priorities to their goals based upon their current states. For example, an AN/TSQ-73 operator can determine whether an uncleared alert message or a hostile aircraft within 30 miles of a critical asset should be attended to next.

3. The operators are capable of estimating the changes that will occur in their goal states if a particular task is performed.

4. Operators are limited in their ability to satisfy their goals. They may not be able to consider all of their goals at once.

These concepts are implemented in the following ways. For each operator goal, the goal state (denoted GS) must be explicitly specified in a way that allows GS to be assigned a unique value (e.g., GS = the number of unassigned hostile tracks). Then a goal priority function, GP, is specified for the goal that assigns a non-negative value to each value of GS. Figure III-1 shows an hypothetical example. The goal is satisfied when the goal state, GS, is between m and M. This is signified by GP = 0 when $m \leq GS \leq M$. If the goal state is less than m or greater than M, then the priority of the goal (i.e., its degree of dissatisfaction) increases linearly.

The meaning of the particular goal priority function in Figure III-1 is that the operator is satisfied and indifferent to any value of the goal state between m and M. Downside deviations (i.e., values of GS less than m) are apparently more important than upside deviations (i.e., values of GS greater than M) since the slope for downside deviations is greater than the slope for upside deviations.

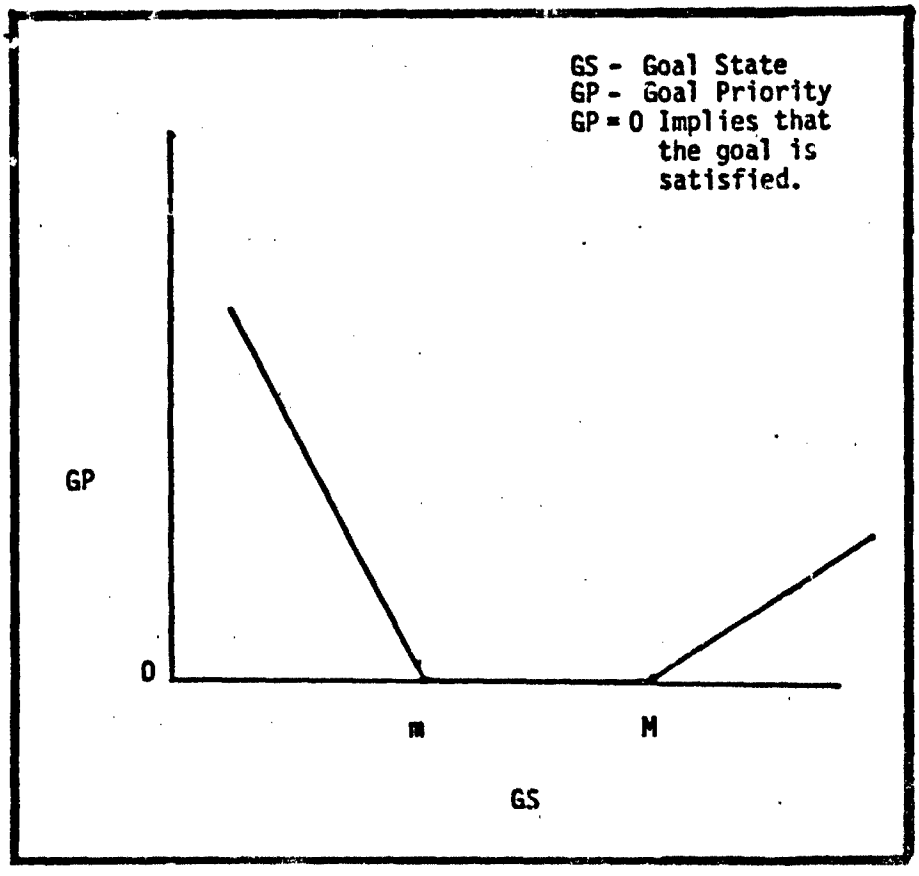


Figure III-1. Example Goal Priority Function.

Each goal that an operator has may have a different priority function. See Figure III-2 for examples. The values of the goal priorities for each goal are compared in an ordinal fashion during task sequencing to determine the most dissatisfied goal or goals. This scheme allows the parameters for a goal priority function to be specified or changed without affecting the priority functions for other goals. Of course, complete independence is not obtained because the modeler must always be aware of the priority functions of the rest of the goals.

In order to simplify the implementation, a standard functional form for goal priority functions is used in MOPADS. Six parameters are associated with each goal for each operator. In the discussion that follows, subscripts on the notation are suppressed for simplicity. The reader should be aware, however, that parameters such as

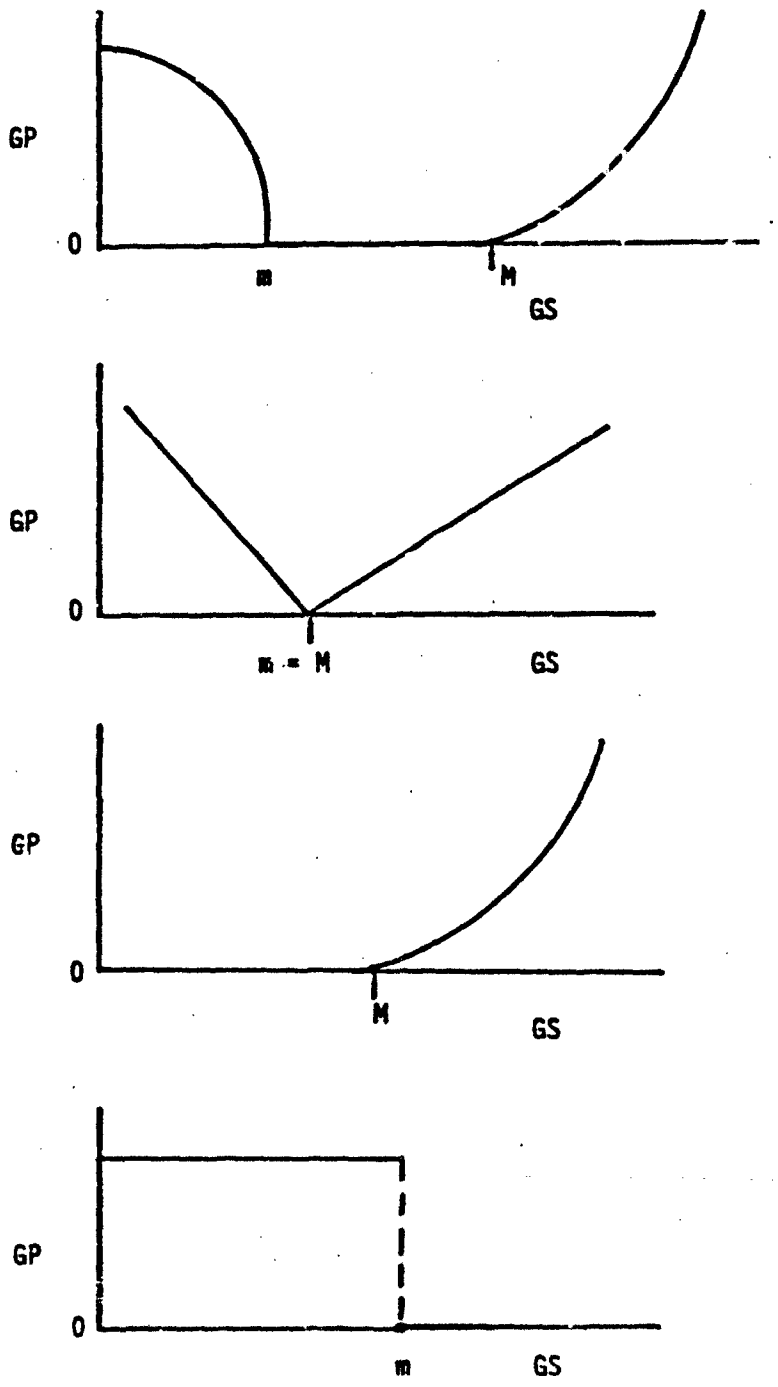


Figure III-2. Example Goal Priority Function Forms.

"little'm" should be written m_{ik} to indicate the parameter value for operator i and goal k.

For each goal a range of satisfaction (m,M) is specified. If the goal state, GS, falls in this interval, then the goal is satisfied and the goal priority, GP, is zero. For values outside the range of satisfaction, the goal priority is evaluated using an exponential function with four parameters denoted a, b, A, and B. The complete goal priority functional form is given below.

If $GS < m$ then $GP = a(m-GS)^b$; $b \geq 0$

If $m \leq GS \leq M$ then $GP = 0$

If $GS > M$ then $GP = A(GS-M)^B$; $B \geq 0$

The purpose of specifying two sets of exponential parameters (i.e., (a,b) and (A,B)) is to provide different priorities for "upside" deviation above the range of satisfaction and "downside" deviations below the range of satisfaction.

This functional form provides considerable flexibility in specifying goal priority function. All of the forms in Figure III-2 can be obtained with appropriate values of the six parameters. For example, in III-2(a), $b < 1$ and $B > 1$; in III-2(b), $m = M$ and $b = B = 1$. In III-2(c), $m = -\infty$ and $B > 1$; the values of a and b need not be specified. In III-2(d), $M = \infty$ and $b = 0$. Since the upside and downside priority functions may be specified independently, a wide variety of functional forms can be obtained. Of course, certain types of functions cannot be represented; for example, asymptotic priority functions are not possible.

Certain types of functions are prohibited by the restriction that $b \geq 0$ and $B \geq 0$. Negative exponents result in functions such as that shown in Figure III-3. With $B < 0$, the priority first increases then decreases for ever increasing deviations from the range of satisfaction. Since this type of goal priority function represents unusual behavior, the MOPADS software does not accept priority functions with b or B negative (the MOPADS user can circumvent this representation by using the basic user interface commands EXAMINE and DEPOSIT to directly insert negative values into the operator state vectors. Descriptions of EXAMINE and DEPOSIT are contained in Polito (1983), and the goal priority functions for the IHAWK and AN/TSQ-73 are contained in Goodin & Walker (1983a,b).

Specification of the parameters of the goal priority functions can be made simple and intuitive by recognizing that a and b, for example, can be determined uniquely by knowing two points on the downside deviation function. Suppose we know two such points as follows:

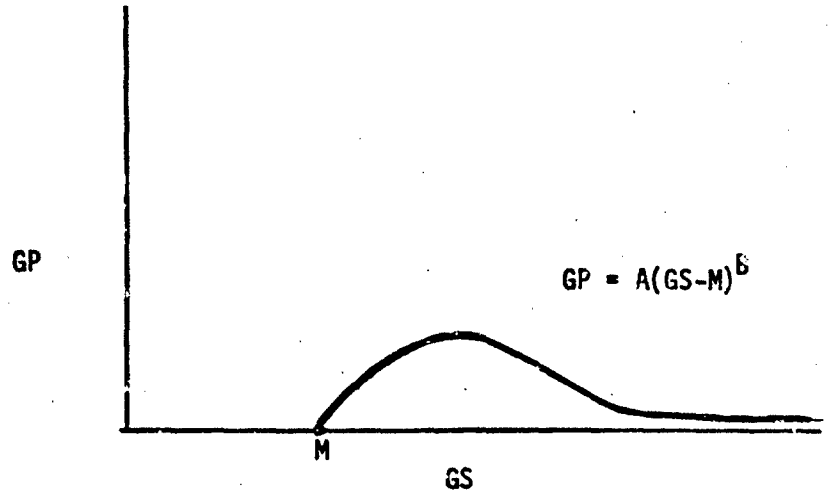


Figure III-3. Example of Prohibited Goal Priority Function with $B < 0$.

$$\text{point 1} = (GS_1, GP_1)$$
$$\text{point 2} = (GS_2, GP_2)$$

where m is finite, $GS_1 < GS_2 < m$ and $GP_1 \geq GP_2 > 0$.

The logorithm (to any base) of the downside priority function is

$$\log(GP) = \log(a) + b\log(m - GS).$$

Substituting the two known points into this equation, eliminating log(a), and solving for b yields

$$b = \frac{\log(GP_1/GP_2)}{\log([m-GS_1]/[m-GS_2])}$$

Then, solve for a from

$$a = \frac{GP_1}{(m-GS_1)^b}$$

For the upside deviation priority function, the corresponding equations are (for two points numbered 3 and 4):

M finite, $M < GS_3 < GS_4$, and $0 < GP_3 \leq GP_4$

$$B = \frac{\log(GP_3/GP_4)}{\log([GS_3-M]/[GS_4-M])}$$

$$A = \frac{GP_3}{(GS_3-M)^B}$$

Using the above four equations, the goal priority functions can be determined automatically from intuitively meaningful data specified by the user.

2-0 THE OPERATOR OBJECTIVE FUNCTION

Suppose an operator evaluates his/her goals and obtains values for each goal priority, GP_i, i = 1, 2...N (where N is the number of goals). At this point he must decide which goal or goals he/she will try to improve. There are several objectives he/she might use. For example, he might try to improve that goal state which has the highest goal priority. This is "putting out the biggest fire first." Or he might seek a course of action that would improve the states of several goals simultaneously. An operator who does this would be taking a more global approach to problem solving.

The operator might also attempt to obtain rapid improvement in his/her goal states. This would involve estimating the time required to perform the various options available to him and selecting the one which yields the most rapid improvement in one or several goals.

MOPADS provides operator objective functions which can account for all of the above considerations. Suppose the operator considers NG goals ($1 \leq NG \leq N$) when selecting the next task. Compute weights for the NG largest goal priorities as follows:

$$w_i = \frac{GP_1}{\sum_{i}^{NG} GP_i}$$

Note that the goal priorities have been ranked from highest to lowest and the w_i are computed only for the NG largest priorities.

Now compute the special weighted average goal priority, AVNG, as follows:

$$AVNG = \sum_{i}^{NG} (w_i \, GP_i)$$

AVNG is the composite goal priority for the operator. Note that the highest weight is assigned to the most dissatisfied goal. If NG is one, then AVNG is the priority of the most dissatisfied goal (the "biggest fire"). As NG increases, the operator is able to consider more and more goals, but he still assigns the greatest weight to the most dissatisfied. In fact, the w_i assign weights to the goals that are proportional to each goal's contribution to the "total dissatisfaction" (i.e., the sum of the NG most largest goal priorities).

When selecting the next task, the operator evaluates AVNG for the current goal states, then the outcomes of each option are evaluated and an expected value of AVNG is calculated for each option. The option which yields the greatest expected reduction in AVNG is selected. In computing the expected values of AVNG, the values of w_i are not re-computed and the same set of NG goals are used as in computing the current value of AVNG. If the operator seeks rapid improvement in his/her goals, then an estimate of the time to perform each option is made, also, and the improvement in AVNG is divided by this time to obtain the improvement per minute. The option that yields the greatest improvement per minute is selected.

As stated earlier, this formulation can be stated in a utility theory framework. The utility function is AVNG which is a function of NG and the goal priorities, i.e., AVNG (NG, GP_1, GP_2, ...GP_N). Each of the goal priorities is a function of its related goal state which is in turn a function of the state of the system, S:

$$U(S) = AVNG(NG, GP_i(GS_i(S)), \; i = 1, 2, \ldots N)$$

The operator's seek to minimize the above utility function.

Thus the goal seeking methodology is a special case of utility theory in which the utility function has been restricted to a special class of parametric expressions which are fairly easy to deal with. The intermediate functions, GP_i, are adjusted by the user to affect the decision making behavior of the simulated operators.

3-0 A PRECISE STATEMENT OF THE TASK SEQUENCING ALGORITHM

Before a formal specification of the task sequencing algorithm can be given, two more concepts must be introduced. First, MOPADS maintains a last-in-first-out stack of operator tasks which the operator will perform. This is because in task sequencing the operator may select to sequentially perform two related tasks. After each task the operator returns to the task sequencing procedure, but if the stack is non-empty, the next task on the stack is selected automatically without a complete goal evaluation. An example of a situation where this capability is needed concerns the Azimuth Speed Operator (ASO) in the IHAWK. This operator has two tasks (Detect New Target and Establish Target Priority) which are always performed in sequence. The second task, Establish Target Priority, is loaded on his/her task stack when the first task is selected.

Secondly, a method is also provided to interrupt this stack processing. Interruptions are always caused by high priority messages. The artifact used to implement this in MOPADS is to make the priority of interrupting messages negative. When an interrupting message is present, a normal goal evalation takes place even if the task stack is non-empty.

Finally, some sequences of tasks on the task stack are so important in terms of mission accomplishment and in terms of correct operation of the MOPADS models that a method is provided to override the interrupting message mechanism. This is a programming convenience that allows the MOPADS modeler to ensure that certain illogical contingencies do not occur in the model.

Also, while it is not strictly a part of task sequencing, an operator can be interrupted during the performance of a task element using normal MSAINT features. Thus it is possible with MOPADS to interrupt the actions of an operator during the performance of an operator task as well as between the performance of operator tasks.

Figure III-4 shows a flow chart of the task sequencing algorithm. If the stack is empty at decision 1, a complete goal evaluation is performed starting at box 5. Otherwise, decision 2 determines if the stack may be interrupted; if not the next task is selected from the stack at box 3. If the stack can be interrupted, decision 4 determines if interrupting messages are present. If not, the stack is not interrupted and the next task is selected at box 3. If interrupting messages exist, a goal evaluation is performed beginning at box 5.

The goal states are computed at box 5. Example goal states are:

1. The maximum priority of any message for this operator.

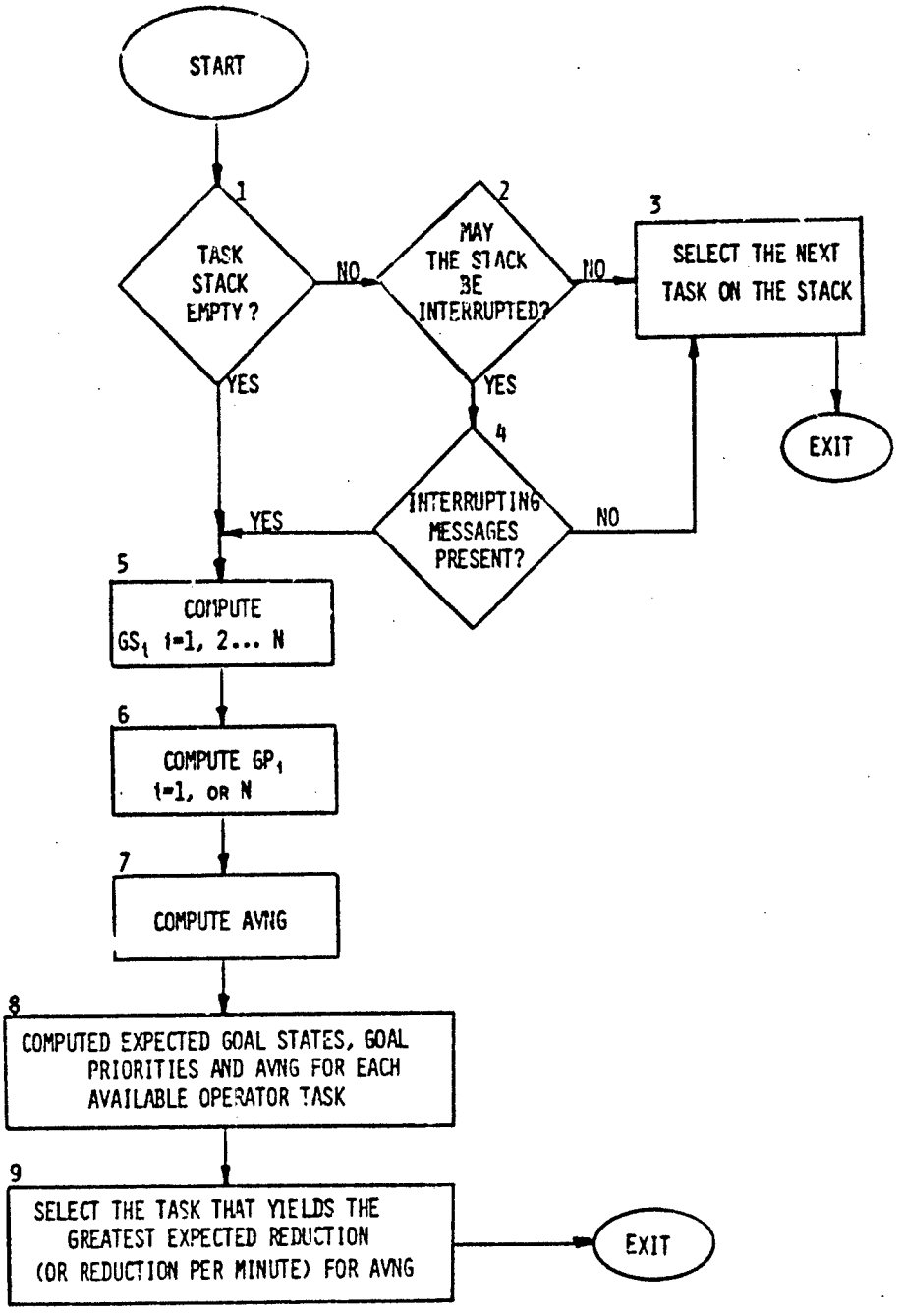


Figure III-4. The Task Sequencing Algorithm.

2. The minimum time for any hostile aircraft to arrive at any protected site.

3. The number of unidentified tracks.

The goal priorities are calculated at box 6 using the goal states and the exponential functions discussed in Section 1-0 above. AVNG is computed at box 7 as discussed in Section 2-0 above.

In box 8, the list of available operator tasks is examined. For each available task, an estimate is made of the expected changes that will occur to the goal states. These expected goal states are used to compute expected values of the goal priorities and, finally, of AVNG. If the operator's objective is to rapidly reduce AVNG, then an estimate of the time to perform each available task is also obtained.

Suppose AVNGJ is the expected value of AVNG if operator task j is selected and TJ is the estimated time to perform task j. Then for each task j compute:

$$DELJ = AVNG - AVNGJ$$

or

$$DELJ = (AVNG - AVNGJ)/TJ$$

and select task j which gives the largest value of DELJ. Each operator has an idle or scan-screen task which is selected if all DELJ are negative or if no other task can be selected. This activity is performed at box 9 of Figure III-4.

IV. IMPLEMENTATION OF THE TASK SEQUENCING PROCEDURE

1-0 TASK SEQUENCING SUBPROGRAMS

The task sequencing procedure has been organized in a modular way to facilitate modification of additions of additional system modules. Figure IV-1 shows the structure of the software. The four programs ending in "Y", TSEQY, GEVALY, GFRIY, and OEVALY, are contained in the common system module programs, Goodin & Polito (1983b), and need only minor modification as new system modules are added.

SUBROUTINE GEVALY is responsible for evaluating the goal states, GS_i, at box 5 in Figure III-4. It calls a separate subroutine for each system module type (e.g., IHAWK, AN/TSQ-73). The three programs GEVALG, GEVALQ, and GEVALW evaluate the goals for the operators of that particular system module type. For example, GEVALQ calls programs GEV1Q, GEV2Q, etc. to evaluate the goal states for goal 1, goal 2, etc. of the battalion AN/TSQ-73 operators.

SUBROUTINE OEVALY is responsible for calculating the expected goal states that will result from selection of available operator tasks (box 8 in Figure III-4). OEVALY calls a program for each system module type (OEVALG, OEVALW, or OEVALQ). The organization in Figure IV-1 is used for the IHAWK. OEVALW calls a program for each operator type. For example, OEASOW is called to evaluate tasks performed by the ASO.

Programs which are specific to a particular system module type are documented in the reference report for that system module. For example, all subprograms ending in the letter Q in Figure IV-1 are contained in Goodin & Walker (1983a).

Modification of the task sequencing programs for the IHAWK ASO would require changes to only SUBROUTINE OEASOW. Addition of a new system module would require trivial modification to GEVALY and OEVALY, and then programs analogous to GEVALQ and OEVALQ and their descendents would need to be written.

2-0 TASK STACK PARAMETERS

The task stack is more than a list of tasks to be performed. Associated with each task is a list of four parameters which can be retrieved at any time. The parameters are used to pass information to MOPADS about a task being performed. For example, if an AN/TSQ-73 operator selects a task to assign a track to a fire unit, the track number and the fire unit identifier will be stored as parameters in the stack for later retrieval when the task is performed.

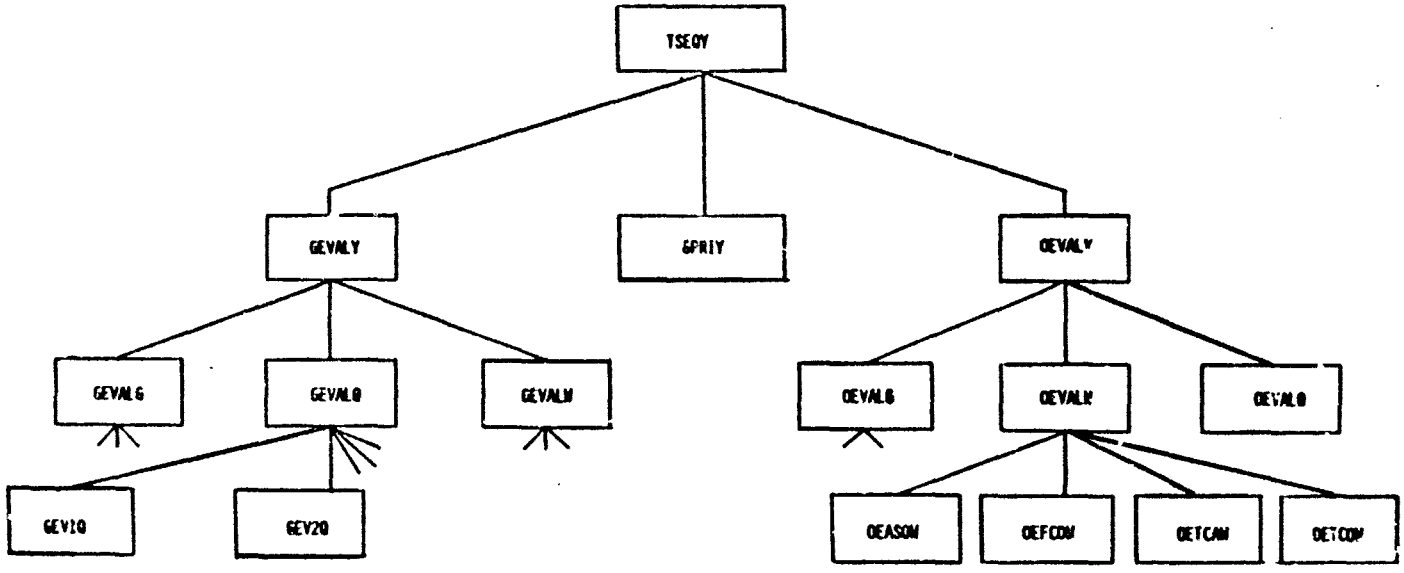


Figure IV-1. Structure of the Task Sequencing Software.

A task sequencing procedure for a new system module can be developed by using the existing programs as guides, and making use of the stack parameter feature.

V. AN EXAMPLE OF GOAL PRIORITIES

There are three important steps in developing a task sequencing procedure for an operator:

1. determining the operator's goals,

2. specifying the goal priority functions, and

3. writing the subprograms GEVAL? and OEVAL? That evaluate the goal states and estimate changes to the goal states that will result from performing various operator tasks.

Steps one and two require discussion with subject matter experts, and they require modeling judgement. An operator's goals must be sufficient to motivate the simulated operator to perform all of his/her tasks if the correct conditions arise. Some of the operator's goals will be straightforward. Subject matter experts will readily identify them. For example, a goal of the IHAWK Tactical Control Officer (TCO) is: maximize the minimum time to arrive at any protected site of any threatening aircraft. This goal motivates the TCO to take actions that protect his critical assets.

Other goals, however, may need to be added as modeling conveniences in order to accommodate the way the MOPADS software represents information. For example, the TCO also has the goal "minimize the maximum priority of outstanding messages." This "goal" is a surrogate for the TCO's obvious attention to messages from a Battalion or Group AN/TSQ-73 and to voice communications from other members of the IHAWK crew. MOPADS represents all of these communications as messages, and it would be impractical to specify a separate goal for responding to each of these communication channels. This goal is somewhat contrived, but does reflect real behavior by the operator. It also allows subject matter experts to give their intuitive priorities to the various messages that the operator receives.

The goals for the IHAWK TCO are shown in Table V-1. Goal six may need some explanation. The IHAWK will engage unknown pop-up targets that threaten it or its assets. Pop-up targets may be engaged without being identified if events occur so quickly that identification is precluded. However, if such an aircraft is identified as friendly during the course of an engagement, the TCO will immediately take action to prevent the aircraft from being destroyed. In the current MOPADS simulations, this situation is the only one in which a friendly aircraft will be engaged.

Table V-1. Goals for the IHAWK Tactical Control Officer.

1. ENGAGE ASSIGNED TRACKS	
Goal State:	The minimum time for any in range, assigned, but not engaged, track to arrive at the IHAWK or its protected site if it immediately turned inbound
2. SELF DEFENSE	
Goal State:	The minimum time for any hostile or unknown, not receding, unassigned track to arrive at the IHAWK it it immediately turned inbound
3. PROTECT CRITICAL ASSETS	
Goal State:	The minimum time for any hostile or unknown, not receding, unassigned track to arrive at a protected site if it immediately turned inbound to the site
4. ATTEND TO MESSAGES	
Goal State:	The maximum priority of any outstanding messages
5. CONSERVE AMMUNITION	
Goal State:	The total number of hot and cold missile available less those expected to be fired at currently engaged tracks
6. PROTECT FRIENDS	
Goal State:	Number of friendly tracks currently being engaged

Figure V-1 shows a set of candidate goal priority functions for the goals given in Table V-1. The goal priority functions are identified by the goal number enclosed in a circle. By convention, a goal priority value of 10 is considered an emergency. Thus, for goal two, an aircraft less than about 1.3 minutes from the IHAWK constitutes an emergency, and the TCO would select actions to immediately attack such an aircraft. Note that in the region less than 1.3 minutes, self defense (goal 2) as the most important goal followed by protection of critical assets (goal 3). Both of these goals take priority over similar tracks assigned by a Battalion AN/TSQ-73. At a time greater than 1.3 minutes, the situation is reversed, however, and the TCO would engage tracks assigned from the battalion before self-initiating engagements on other tracks.

Note that self defense and protection of critical assets takes precedence over protection of friendly aircraft in the critical range below 1.3 minutes but not above this time. The TCO is concerned about conserving ammunition only when three or less remain, since this is the number of missiles on one launcher.

The priority functions for messages is the same as the message priority. In other words, a message whose priority is 5 has a goal priority of 5, also. Be sure to note that the goal state axis at the bottom of Figure V-1 has different units depending on the particular goal. It is possible that goal one has a goal state of three minutes which implies a goal priority of about three while the highest priority outstanding message has priority six. In this case the goal priority function for goal four would have a value of six, and the TCO would respond to the message before attending to an assigned track.

The goal priority functions in Figure V-1 were developed in consultation with subject matter experts by asking them pairwise type questions. For example, "Which is of greater concern to the TCO, an unengaged hostile aircraft within 2 minutes or a new assignment message from the Battalion?" The results of these discussions can be synthesized to goal priority functions whose ordinal rankings reflect the priorities of the experts. This task is made easier by the fact the ordinal rather than cardinal rankings are most significant. For example, consider goal 2 (self defense) and goal 6 (protect friendly). The significant characteristics of their goal priority functions is that for a goal 2 goal state under about 1.3 minutes, the self defense goal is more important than protecting friendly. The amount by which the goal 2 priority exceeds the goal 6 priority is not significant. This fact greatly simplifies the task of developing goal priority functions.

The task sequencing algorithm presented in this report separates the functions of goal evaluation, goal priority evaluation, and task selection. By doing so, it simplifies the third step in developing a task sequencing procedure, i.e., writing the subprograms that evaluate goal states and estimate the impact on goal

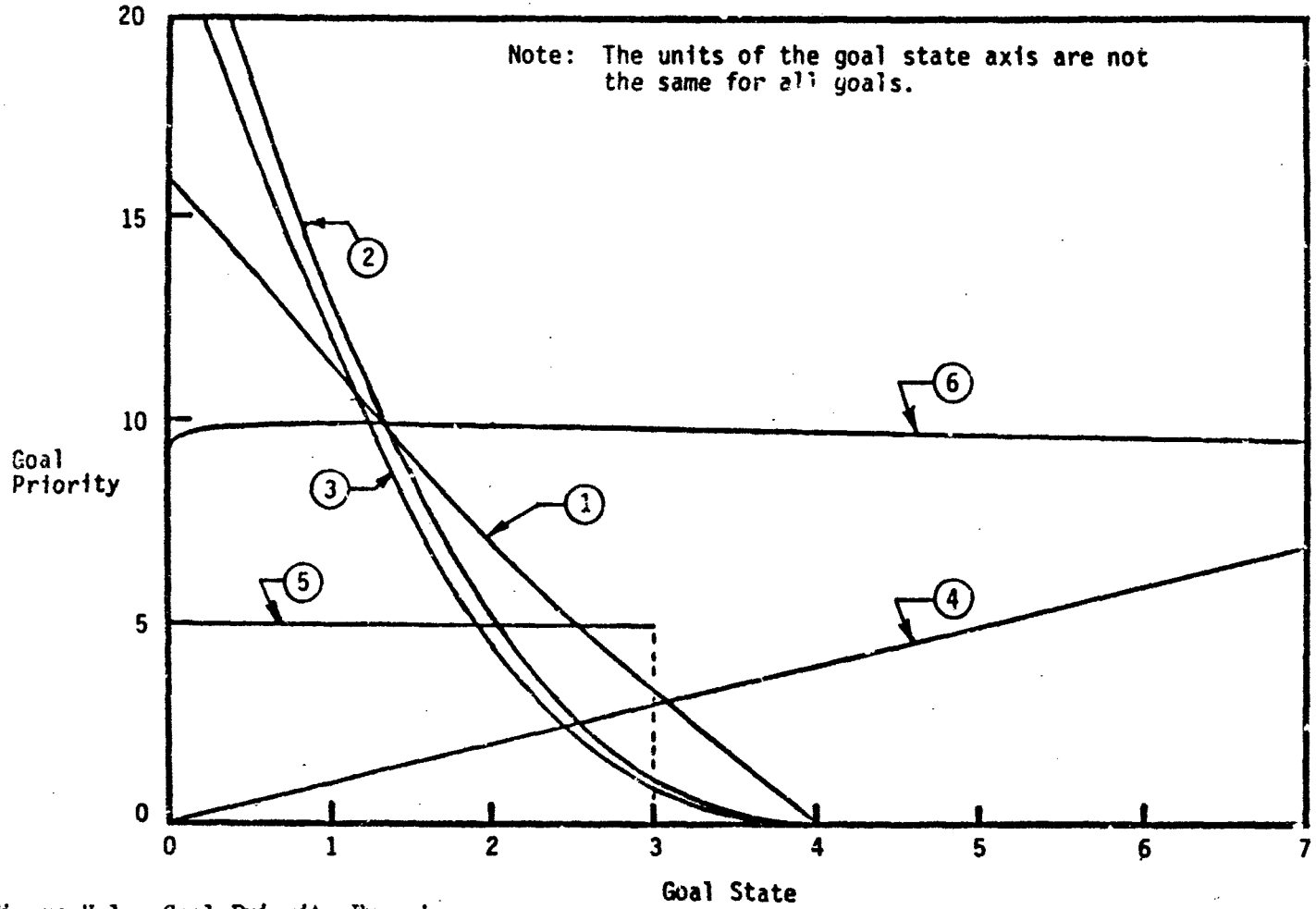


Figure V-1. Goal Priority Functions.

states of various operator actions. These subprograms can be developed without regard to how the operator will weight the importance of the goals.

The modularity achieved by this goal seeking methodology:

1. permits goal evaluation programs to be written independently of other goals.

2. allows the MOPADS user to alter an operator's goal seeking behavior by changing only parameters of the goal priority functions,

3. allows automatic generation of goal priority functions from intuitive inputs from the MOPADS user, and

4. causes the simulated operators to respond to the air battle in a rational manner.

Thus, the methodology satisfies all of the desirable features of a task sequencing method discussed in Section II, 1-0.

VI. REFERENCES

Goodin, J. R. & Polito, J. User guide for the AN/TSQ-73 system module (MOPADS Vol. 3.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Goodin, J. R. & Polito, J. Documentation manual for the MOPADS control module (MOPADS/CNTRL) and the MOPADS common system module programs (MOPADS/CSMP) (MOPADS Vol. 5.19). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Goodin, J. R. & Walker, J. L. Documentation manual for the AN/TSQ-73 system module (MOPADS Vol. 5.15). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Goodin, J. R. & Walker, J. L. Documentation manual for the IHAWK system module (MOPADS Vol. 5.16). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

VII. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VIII. CHANGE NOTICES

APPENDIX P

MOPADS FINAL REPORT:

MOPADS DOCUMENTATION STYLE MANUAL

TABLE OF CONTENTS

TABLE OF CONTENTS (continued)

LIST OF FIGURES

I. INTRODUCTION

This manual is intended as a guide for both authors and typists of MOPADS reports. Its purpose is to ensure uniform organization for MOPADS reports, to expedite the distribution of reports, and to satisfy contractual requirements for form and format.

Authors will be primarily concerned with Section II which describes report organization. Topics discussed in this section are section and subsection numbering, figure and table numbering, and preparation of references and footnotes.

Typists must also be familiar with Section II, but they will be most concerned with Sections III and IV. These last two sections describe typing and delivery of reports.

The requirements presented in the subsequent sections are minimal and should not hinder preparation of reports. Any special cases which arise should be discussed with the Pritsker & Associates Project Leader.

II. REPORT ORGANIZATION

1-0 SECTION AND SUBSECTION ORGANIZATION.

Figure II-1 is an example Table of Contents for a MOPADS report. It shows how major sections and subsections are to be designated. Important features of this organization follow.

1-1. Major sections are designated by Roman Numerals. Titles are in upper case and no text is inserted between the major section title and the first subsection. Major sections begin new, right-hand pages.

1-2. First subsections are numbered 1-0, 2-0, 3-0, etc. A first subsection has a title and is capitalized. No text appears on the same line as a first subsection heading.

1-3. Second subsections are numbered 1-1, 1-2, 1-3, etc. The first few words of these sections may be underlined. Also, text may begin on the same line as the subsection title.

1-4. Third subsections are lettered with lower case letters, e.g., 1-2.a, 1-2.b, etc. The first few words of the title may be underlined, and text may begin on the same line as the subsection number/letter.

1-5. The beginning of each subsection is indented six spaces. Subsequent text extends fully from the left to the right margins.

2-0 FIGURES AND TABLES.

2-1. Figures.

Figures will be placed on a separate page and will be numbered consecutively within major sections (e.g., I-1, I-2, II-1, II-2, etc.). Figures must fit within the typing guide (see Section III), but figures may be continued to additional pages. Figure II-2 is a sample of a List of Figures page and Figure II-3 is a sample figure. Note that the figure title appears at the bottom of the page on the margin line. The format shown in these figures should be followed. Computer output may be used as figures.

Generally, foldout pages are to be avoided, but very large figures which will be difficult to breakup or reduce may be presented on foldouts. Authors must retain original drawings in case reformatting is later required.

TABLE OF CONTENTS

Figure II-1. Example Table of Contents.

LIST OF FIGURES

Figure II-2. Sample List of Figures.

Figure II-3. Sample Figure.

MOPADS PROGRESS

PROJECT TASKS

Task Description	Progress	Notes	Task Description	Progress	Notes
Task 1.1	0 50 100		Task 3.2	0 50 100	
Task 1.2	0 50 100		Task 3.3	0 50 100	
Task 1.3	0 50 100		Task 3.4	0 50 100	
Task 1.4	0 50 100			0 50 100	
Task 1.5	0 50 100			0 50 100	
Task 1.6	0 50 100			0 50 100	
Task 2.1	0 50 100			0 50 100	
Task 2.2	0 50 100			0 50 100	
Task 3.1	0 50 100			0 50 100	

Figure B-2. MOPADS Project Tasks.

2-2. Tables.

Tables will be numbered consecutively in each major section (e.g., I-1, I-2, II-1, II-2, II-3, etc.). Table titles will be centered above the table, and more than one table may appear on a single page if they will fit. Tables may also be continued to more than one page. Foldout pages are also permitted.

Figure II-4 is a sample List of Tables, and Figure II-5 is a sample table. The format shown in these figures should be followed.

Computer output may be used in tables, but must conform to margin requirements, see Figure II-6. Program listings and other long output should be put in appendices. As a rule, long program lists will not appear in reports since the programs themselves are deliverable items.

3-0 REFERENCES AND FOOTNOTES.

3-1. References.

References cited in the text and listed in the REFERENCES section will conform to the recommended format of the American Psychological Association. The reference for this standard is:

> Publication Manual of the American Psychological Association, 2nd ed.

Copies may be ordered from Publication Sales, APA, 1200 Seventeenth St., N.W., Washington, D.C., 20036.

Pages 59 through 63 of this manual contain the pertinent material on references. References to other MOPADS reports should include the term "MOPADS" (e.g., MOPADS Volume 5.3). Figure II-7 is a sample REFERENCE section.

3-2. Footnotes.

Footnotes will be numbered consecutively throughout the report. Footnotes will appear on the page they are noted. Long footnotes should be incorporated in the text or left out. Footnotes are designated as follows: 2/ 7/. This notation is used both in the text and at the bottom of the page where the footnote text appears.

Footnotes in figures and tables will use strings of asterisks (e.g., "*", "***").

4-0 APPENDICES.

Appendices are major sections that are designated by capital letters instead of Roman numerals (e.g., A, B, etc.). Tables and Figures will be numbered within the appendix as in other major sections (e.g., A-1, A-2, etc.).

LIST OF TABLES

Figure II-4. Sample List of Tables.

Table II-3

AN/TSQ-73 FLOWCHART CODES

DC	-	Display Console Controls
E	-	AN Keyboard Entry
RIE	-	RIE Panel 1
KPU	-	Requires use of keyboard printing unit
R	-	System Response

Numeric codes:

DC - Display console controls

XX.YY.ZZ

XX:

Alerts	1
Background data display	2
Track data display	3
Fire unit data display	4
System mode	5
Task selections	6
Task functions	7
Faults	8
Symbol brightness	9
Video brightness	10
Video selections	11
ARO data selections	12
Voice comm station	13
AN keyboard	14
VAR SCALE	15
POSN TAB	16

YY: Row number

Figure II-5. Sample Table.

Table II-4

INDEPENDENT VARIABLES-APTITUDE

RISK ACCEPTANCE
INTELLEGENCE
AGE
OPERATOR TYPE
SENSE OF DIRECTION
WILLINGNESS TO TAKE RISKS
OBSERVATION NOISE
PERCEPTUAL TIME DELAY
MOTOR NOISE
OPERATOR GAIN (8&9 COMBINED)
GENERAL PROFICIENCY FACTOR
THRESHOLD CONTRAST
ASPIRATION LEVEL
OPERATOR PRECISION
TEAM SKILL LEVEL
TRACKING ABILITIES
MOTIVATION LEVEL
VISUAL ACUITY
GENERAL APTITUDE SCORE
MORALE LEVEL
LEADERSHIP ABILITY

Table II-5

INDEPENDENT VARIABLES-LEARNING/PRACTICE

PRACTICE TRIALS
TIME SINCE LAST PRACTICED
AMOUNT OF OVERLEARNING
TIME SINCE INITIALLY LEARNED
TRAINING EMPHASIS(SPEED VS COORDINATION)
AMOUNT OF TRAINING
PRESENCE OF FEEDBACK
NUMBER OF CORRECT CHOICES
MASSED VS DISTRIBUTED PRACTICE
AMOUNT OF INITIAL TRAINING
TYPE OF FEEDBACK
SUCCESSFUL TRIALS

Figure II-6. Sample Tables with Computer Output.

V. REFERENCES

Laughery, K. R., HOMO Establishment of Performance Criteria for Non-Decision Making Tasks, U. S. Army Research Institute, MOPADS Volume 5.6DF, January 1982.

Operator's Manual: Initialization and Operating Procedures, Guided Missile Air Defense System AN/TSQ-73, Headquarters, Department of the Army, TM-9-1430-652-10-3, Change 6, August 1978.

Politc, Joseph and Laughery, K. R., MOPADS Technical Report Year-One, Under U. S. Army Research Institute Contract MDA903-81-C-AA06.

Pritsker, A. Alan B. and Pegden, Claude Dennis, Introduction to Simulation and SLAM, A Halsted Press Book, John Wiley & Sons, Inc., New York, 1979.

U. S. Army Air Defense Artillery Operations, Headquarters, Department of the Army, FM 44-1-1, Change 0, October 1969.

Wortman, David B., Duket, Steven D., Seifert, Deborah J., Hann, Reuben L., and Chubb, Gerald F., Simulation Using SAINT: A User-Oriented Instruction Manual, Aerospace Medical Research Laboratory, AMRL-TR-77-61, July 1978 (a).

Wortman, David B., Duket, Steven D., Seifert, Deborah J., Hann, Reuben L., and Chubb, Gerald F., The SAINT User's Manual, Aerospace Medical Research Laboratory, AMRL-TR-77-62, July 1978 (b).

Figure II-7. Sample References.

5-0 COMPUTER OUTPUT

All computer output in figures and tables will be trimmed, and pasted before duplication for a professional appearance. No lines from lined paper or tractor holes should be visable.

Long lists of raw computer generated information are to be avoided. Where such data can be considered a deliverable item, magnetic tapes will be delivered and only pertinent sections will be reproduced in reports. This restriction is not intended to preclude the use of computers to generate graphical or tabular information for use as tables and figures.

III. REPORT PREPARATION

1-0 ARRANGEMENT OF THE REPORT.

The report will be arranged as shown in Figure II-1. Each section is described briefly below.

1-1. Cover. Company covers may be used until a standard MOPADS cover is available. The MOPADS volume number and date appear in the upper right corner of the cover. See Figure III-1 for an example.

1-2. Title Page. A sample title page is shown in Figure III-2. The format of the title page must be followed exactly.

1-3. Disclaimer Statement. This statement will appear in all MOPADS documents. Figure III-3 contains the disclaimer. It may be copied and used directly in future documents.

1-4. DD1473. This form will be prepared by Pritsker & Associates when the document is distributed. Preparers must leave space for it. A sample is shown in Figure III-4.

1-5. Table of Contents, List of Figures, List of Tables, Abbreviations and Terminology. See Figures II-1, 2, 4, and III-5 for samples.

1-6. Main Body of Report. Text.

1-7. References. Figure II-7 is a sample.

1-8. Appendices. When appendices are present, they should be placed directly after the references. For bulky apendices, appendices may be bound separately. The appendix section must contain a note explaining the separate binding. The separately bound appendices will contain the table of contents for the entire volume but only the text for the appendices.

1-9. Distribution List. Distribution will be coordinated with the Pritsker & Associates Project Leader prior to duplication. As a general rule, the distribution list shown in Figure III-6 will be used.

1-10. Change Notices. This section will contain the cover pages of change notices. The section will be present but empty when the document is first prepared.

MOPADS VOLUME 4.3DF
APRIL 15, 1982

DOCUMENTATION REQUIREMENTS AND DEVELOPMENT GUIDELINES

FOR

MOPADS AIR DEFENSE SYSTEM MODULES

Author(s)	Jack L. Walker	Pritsker & Associates, Inc.
	Joseph Polito	Pritsker & Associates, Inc.

U. S. Army Contract MDA903-81-C-AA06
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

P&A

Pritsker & Associates, Inc.

Figure III-1. Sample Cover Page.

MOPADS VOLUME 4.2DF
FEBRUARY 28, 1982

FORTRAN STYLE AND DOCUMENTATION REQUIREMENTS FOR MOPADS OPERATOR AND NON-OPERATOR SOFTWARE

Author(s): Jack L. Walker — Pritsker & Associates, Inc.
Joseph Polito — Pritsker & Associates, Inc.

U. S. Army Contract MDA903-81-C-AA06
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916
Contract Expiration Date: July 31, 1983

Prime Contractor:
Pritsker & Associates, Inc.
P. O. Box 8345
Albuquerque, New Mexico 87198
(505) 255-5597

Subcontractor:
Arvin/Calspan
Advanced Technology Center
Applied Technology Group
P.O.Box 400
Buffalo, New York 14225
(716) 632-7500

Figure III-2. Sample Title Page.

The views, opinions, and/or findings contained in this report are those of the authors and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other official documentation.

Figure III-3. Disclaimer Statement.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle)		5. TYPE OF REPORT & PERIOD COVERED
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s)		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS		12. REPORT DATE
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report)
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)		

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Figure III-4. Sample DD1473.

ABBREVIATIONS AND TERMINOLOGY

1-0 ABBREVIATIONS

(This section starts on a new page.)

BCC	Battery Control Central
CWAR	Continuous Wave Acquisition Radar
CWTDC	Continuous Wave Target Detection Console
FC	Firing Console
HIFIR	High-Powered Illuminator Radar
PCP	Platoon Command Post
TAS	Tracking Adjunct System
TCC	Tactical Control Console
TDECC	Tactical Display and Engagement Control Console

2-0 STANDARD MOPADS TERMINOLOGY

(See Appendix A for contents of this section. This section starts on a new page.)

3-0 OTHER TERMINOLOGY

(As required for the report. Use the format in Appendix A.)

Figure III-5. Sample Abbreviations and Terminology.

V. DISTRIBUTION LIST

Dr. Charles Jorgensen (5)
PERI-IB
U. S. Army Research Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 8345
Albuquerque, New Mexico 87198

ACO- Loretta McIntire (2)
DCASMA (S1501A)
Building 1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service - Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. Robert Sugerman (1)
Calspan Corporation
Advanced Technology Center
P. O. Box 400
Buffalo, New York 14225

Mr. Ron Laughery (1)
Calspan Corporation
Advanced Technology Center
9132 Thunderhead Drive
Boulder, Colorado 80302

Figure III-6. Distribution List.

2-0 TYPE FACE, MARGINS, AND SPACING.

2-1. Type Face.

An Elite 12 (12 characters/inch) typing element will be used for the text typing of all reports (this requirement does not apply to computer output used in figures and tables). It is recommended that major section headings be typed in upper case with an Orator element. This may not be possible if the report is produced on a word processor, hence upper case Elite 12 is acceptable.

2-2. Margins.

The margins shown below are to be used without variation.

Top Edge:	3/4" from top of paper
Left Side:	1-3/4" from side of paper
Right Side:	1-1/8" from side of paper
Bottom Edge:	1-10/16 from bottom of page
Page No:	1-1/8" from bottom of paper, centered
Text Body:	5-1/2" wide by 8-1/2" high

The above assumes 8-1/2" x 11" paper. The margins are designed to provide correct margins when duplicated on 8" x 10-1/2" paper. A sample typing guide is shown in Figure III-7. Originals for local reproduction may be obtained from Pritsker & Associates or may be copied from the sample in Appendix B.

2-3. Spacing.

The following rules apply for spacing.

1. Text will be single spaced.
2. Double space between paragraphs.
3. Double space between subsections and between subsection headings and text in that subsection.
4. Double space after table number and table title.
5. Double space twice after a major section, table of contents, list of figures, list of tables, abbreviations, references, appendices, distribution list, and change notices.
6. Indent six spaces for each subsection heading and for new paragraphs.
7. Triple space before the beginning of first subsections (2-0, 3-0, etc.).

3-0 PRELIMINARY PAGES AND PAGE NUMBERING.

3-1. Preliminary Pages.

The preliminary pages are made up of the following:

1. Cover and title page
2. Disclaimer statement
3. DD1473
4. Table of Contents
5. List of Figures
6. List of Tables
7. Abbreviations

3-2. Page Numbering.

Pages are numbered as follows:

Page	Number
cover	not numbered
title page	i (not numbered)
disclaimer statement	ii
DD1473	iii and iv (not numbered)
Table of Contents	v
List of Figures	vi
List of Tables	vii
Abbreviations	viii
Main body	numbered within major sections (e.g., I-1, I-2)
Appendices	numbered within appendix (e.g., A-1, A-2)

Page numbers will be centered at the bottom of the page 1/2" below the bottom margin (see Figure III-7).

4-0 A LIST OF KEY WORDS AND ABSTRACT.

A list of "key words" is to be provided to Pritsker & Associates when the reports are delivered. These words will be incorporated into the DD1473.

An abstract is to be written for each report and also sent on a separate sheet of paper to Pritsker & Associates when the reports are delivered.

IV. DELIVERY OF THE REPORT

1-0 DISTRIBUTION AND BINDING.

1-1. Binding.

Reports one inch (1") thick or less will be bound with General Binding Corporation combs. Reports over one inch will be bound in three-hole loose leaf notebooks. Long reports should be copied on both sides of the pages to fit within the one inch size if possible.

1-2. Distribution.

Distribution of copies will be handled as follows:

1-2.a. The author will keep the "original" in a camera-ready form (i.e., copied on one side of the paper, unbound, no holes, staples, etc.).

1-2.b. A "reference copy" will be sent to Pritsker & Associates which will be a copy of the original in the same condition (unpunched, unbound, clean, no holes, staples, etc.). For reports authored by Pritsker & Associates, the reference and original copies may be the same.

1-2.c. Sufficient copies for distribution will be sent to Pritsker & Associates with prepared covers. They will be drilled or punched (excluding the reference copy) for the appropriate binding but will be unbound and collated. This includes the copies to be distributed to the author(s). Pritsker & Associates will supply the bindings and/or notebooks. Do not three-hole punch the prepared covers for reports over one inch thick.

1-2.d. Pritsker & Associates will bind the copies and ship them to the recipients.

1-2.e. When final versions of MOPADS documents are to be submitted to the Army or whenever the Army requests, camera-ready copies will be prepared from the originals.

2-0 SUBMISSION OF CHANGES.

When changes are made to drafts, those pages which are changed will be distributed to all recipients of the document. Distribution

and binding requirements will be the same as in Section 1-0 above. If the report was distributed copied on both sides of the paper, changed pages must be similarly copied so one-for-one replacement is possible. Changed pages will contain no annotation to indicate that changes have been made.

Inserted pages will be numbered fractionally (e.g., VI-2.1, V-2.2, etc.). Deleted pages will be denoted by multiple page references on the preceding page (e.g., "III-2 to III-4" to indicate that pages III-3 and III-4 have been deleted).

Packets of changed pages will be accompanied by a Change Notice form shown in Figure IV-1. Copies for local reproduction are contained in Appendix B.

Recipients of changes will insert/delete pages from their reports and file the Change Notice in the Change Notice section.

MOPADS Volume 4.2

CHANGE
No. 01

Date: 3/15/1982

FORTRAN Style and Documentation Requirements
for MOPADS
Operator and Non-Operator Software

Contract ARI Field Unit, MDA903-81-C-AA06

MOPADS Volume 4.2, dated February 28, 1982, is changed as follows:

1. Remove old pages and insert new pages as indicated below.

Remove Pages	Insert Pages
v,vi	v,vi
II-3	II-3
	IV-3 (this form)

2. File this change sheet in the back of the publication under Change Notices.

Figure IV-1. Sample Change Notice Form.

V. DISTRIBUTION LIST

Dr. Charles Jorgensen (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 8345
Albuquerque, New Mexico 87198

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service - Washington
Room 1D-245
The Pentagon
Washington, D.C. 20310

Dr. Robert Sugerman (1)
Calspan Corporation
Advanced Technology Center
P. O. Box 400
Buffalo, New York 14225

Mr. Ron Laughery (1)
Calspan Corporation
Advanced Technology Center
9132 Thunderhead Drive
Boulder, Colorado 80302

VI. CHANGE NOTICES

MOPADS Volume 5.8 DF

CHANGE
No. 1

Date: August 16, 1982

Contract ARI Field Unit, MDA903-81-C-AA06

1. Remove old pages and insert new pages as indicated below.

Remove Pages	Insert Pages
II-5,6	II-5,6
II-9,10	II-9,10
III-3,4	III-3,4

2. File this change sheet in the back of the publication under Change Notices.

APPENDIX A

The following pages contain the Standard MOPADS Terminology which will be used in the MOPADS documents. These pages are to be placed following the Abbreviations section in the reports.

(Blank Page)

STANDARD MOPADS TERMINOLOGY

AIR DEFENSE SYSTEM

A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.

AIR DEFENSE SYSTEM MODULE

Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.

AIR SCENARIO

A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.

BRANCHING

A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.

DATA BASE CONTROL SYSTEM

That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.

DATA SOURCE SPECIALIST

A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE

An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR

An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.

MODERATOR FUNCTION

A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.

MOPADS DATA BASE

A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulations.

MOPADS MODELER

An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.

MOPADS USER

An analyst who will design and conduct simulation experiments with the MOPADS software.

MSAINT (MOPADS/SAINT)

The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE

One element of an Operator State Vector.

OPERATOR STATE VECTOR

An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK

An operator activity identified on Directorate of Training Development (DTD) "critical task lists" or analogous documentation for Air Defense Systems.

SAINT

The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.

SIMULATION DATA SET

The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.

SIMULATION STATE

At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.

SYSTEM MODULES

See Air Defense System Modules.

TACTICAL SCENARIO

The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT

An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.

TASK

See Operator Task

TASK ELEMENTS

Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE

A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.

TASK SEQUENCING MODERATOR FUNCTION

A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified on Directorate of Training Development (DTD) "critical task lists" or analogous documentation for Air Defense Systems.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

APPENDIX B

This section contains blank forms which may be locally reproduced.

APPENDIX Q

MOPADS FINAL REPORT:

MOPADS UTILITY PROGRAMS

TABLE OF CONTENTS

TERMINOLOGY

1-0 STANDARD MOPADS TERMINOLOGY.

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

I. PURPOSE

This document describes a collection of utility programs used by the MOPADS software. Each of the major modules has similar needs for performing routine operations such as array copying, initialization, simple input and output, and file opening and closing. Also, there are several functions relatively specific to MOPADS such as certain geometric calculations and encoding/decoding of operator information that are also performed by more than one module. In keeping with the modular design approach of MOPADS, these functions have been collected into a separate module and documented separately here.

II. DATA STRUCTURE DESCRIPTION

Since UTIL is a collection of more or less unrelated programs it has no major internal data structure. The single exception to this is a set of programs that perform encoding and decoding of characters to numbers and vice versa (see LPOSU and CPOSU in Section V). A single array containing these correspondences is contained in a COMMON block described in Section VIII.

For all other subprograms in UTIL, the required data is passed as formal parameters.

III. OVERVIEW OF THE FLOW OF CONTROL

As a general rule, the programs of UTIL are at the lowest level of processing. In other words, they call only other routines in UTIL or implicit FORTRAN functions. Some I/O programs, however, call programs in the FFIN2 module (Polito).

IV. EXTERNAL FILE USAGE

In every case where a UTIL program must perform operations with external files, the unit number (and/or the file name) is passed to the program as a formal parameter. With the exception of Subroutines OPENSU and OPENDU (whose functions are to open files), the UTIL programs assume that an appropriate file has been correctly associated with the unit number by the calling module. The UTIL programs do no error checking on the files.

V. SUBPROGRAM DESCRIPTIONS

The MOPADS assigned suffix for UTIL is the letter "U". All subprograms, COMMON block labels, and variables in COMMON end with this letter.

```
      BLOCK DATA BLOCKU
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
```

```
      LOGICAL FUNCTION CEQVU (CHR,CREF)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION IS USED TO DETERMINE IF A CHARACTER IS
C            EQUAL TO A REFERENCE CHARACTER.
C
C**INPUT PARAMETERS:   CHR=CHARACTER TO BE COMPARED AGAINST SOME
C                          REFERENCE CHARACTER TO SEE IF THEY ARE
C                          EQUIVALENT
C                      CREF=REFERENCE CHARACTER
C
C**OUTPUT PARAMETERS:  CEQVU=.FALSE. IF CHR DOES NOT EQUAL CREF
C                           =.TRUE. IF CHR EQUALS CREF
C
```

```
      SUBROUTINE COPYCU (CARAY1,N,CARAY2)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE COPIES THE CONTENTS OF THE CHARACTER ARRAY
C            CARAY1 TO THE CHARACTER ARRAY CARAY2.
C
C**INPUT PARAMETERS:   CARAY1=THE CHARACTER ARRAY TO BE COPIED INTO
C                             CARAY2
C                      N=THE DIMENSION OR LENGTH OF ARRAYS CARAY1 AND
C                        CARAY2
C
C**OUTPUT PARAMETERS:  CARAY2=THE CHARACTER ARRAY COPIED FROM CARAY1
C
```

```
      SUBROUTINE COPYIU (IARAY1,N,IARAY2)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE COPIES THE CONTENTS OF THE INTEGER ARRAY
C            IARAY1 TO THE INTEGER ARRAY IARAY2.
C
C**INPUT PARAMETERS:   IARAY1=THE INTEGER ARRAY TO BE COPIED INTO
C                             IARAY2
C                      N=THE DIMENSION OR LENGTH OF ARRAYS IARAY1 AND
C                        IARAY2
C
C**OUTPUT PARAMETERS:  IARAY2=THE INTEGER ARRAY COPIED FROM IARAY1
C
```

```
      SUBROUTINE COPYRU (RARAY1,N,RARAY2)
C
C**MODULE MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE COPIES THE CONTENTS OF THE REAL ARRAY
C            RARAY1 TO THE REAL ARRAY RARAY2.
C
C**INPUT PARAMETERS:    RARAY1=THE REAL ARRAY TO BE COPIED INTO RARAY2
C                       N=THE DIMENSION OR LENGTH OF THE ARRAYS RARAY1
C                         AND RARAY2
C
C**OUTPUT PARAMETERS:  RARAY2=THE REAL ARRAY COPIED FROM RARAY1
C
```

```
      CHARACTER*1 FUNCTION CPOSU (LPOS)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION IS USED TO CONVERT A TWO DIGIT INTEGER
C            CODE VALUE,LPOS,INTO ITS ONE CHARACTER ALPHANUMERIC
C            EQUIVALENT.
C
C**INPUT PARAMETERS:   LPOS=INTEGER VALUE TO BE DECODED
C
```

```
C**OUTPUT PARAMETERS:  CPOSU=' ' IF LPOS.LE.0 OR LPOS.GT.36,ELSE
C                        =ALPHANUMERIC CHARACTER EQUIVALENT OF LPOS
C                         WHERE 1-26 REPRESENT A-Z AND 27-36
C                         0-9
C
```

```
      FUNCTION EXSMU (A1,X1,A2,X2)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION RETURNS THE SMOOTHED VALUE
C            FOR TWO DATA POINTS.
C
C**INPUT PARAMETERS:   A1=SMOOTHING CONSTANT FOR X1
C                      X1=DATA POINT
C                      A2=SMOOTHING CONSTANT FOR X2
C                      X2=DATA POINT
C
C**OUTPUT PARAMETERS:  EXSMU=SMOOTHED VALUE RETURNED
C
```

```
      SUBROUTINE FSPU(LINE,NCOL,ICODE)
C
C**PURPOSE: FSPU WILL EXAMINE LINE TO DETERMINE IF IT
C           IS THE FIRST OR LAST STATEMENT OF A FORTRAN PROGRAM
C           UNIT OR A NEW LAMP DECK OR COMDECK.
C           FSPU CHANGES THE SEPARATORS AND THE LINE TERMINATOR
C           CHARACTER FOR THE FFIN2 PROGRAMS.  THESE CHANGES ARE
C           NOT REVERSED BY FSPU.
C
C**INPUT PARAMETERS: LINE-CHARACTER VARIABLE CONTAINING A FORTRAN
C                         STATEMENT
C                    NCOL-THE NUMBER OF CHARACTER POSITIONS IN LINE
C**OUTPUT PARAMETERS: ICODE-OUTPUT CODE
C                          -1-END STATEMENT
C                           0-NOT A NEW PROGRAM UNIT, LAMP DECK,
C                             OR END STATEMENT
C                           1-"PROGRAM"
C                           2-"SUBROUTINE"
C                           3-"FUNCTION" (ALL TYPES)
C                           4-"BLOCK DATA"
C                           5-"*DECK"
C                           6-"*COMDECK"
C
```

```
C--STATEMENTS THAT START A NEW PROGRAM UNIT MUST HAVE BLANKS OR
C  "(" AS SEPARATORS.
C         E.G.
C                 "BLOCK DATA" NOT "BLOCKDATA"
C                 "FUNCTION ONE" NOT "FUNCTIONONE"
C
```

```
      SUBROUTINE HEADHU (POS1,POS2,HDN,D1,D2)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO CALCULATE THE COMPASS AZIMUTH
C            FROM POSITION 1 TO POSITION 2,THE GROUND DISTANCE BETWEEN
C            THE TWO POINTS (DISTANCE IGNORING ALTITUDE),AND THE
C            LINEAR DISTANCE (SHORTEST PATH) BETWEEN THE TWO POINTS.
C
C**INPUT PARAMETERS:   POS1(3)=COORDINATES POINT 1
C                           1 =X COORDINATE POINT 1 (UNITS=MILES)
C                           2 =Y COORDINATE POINT 1 (UNITS=MILES)
C                           3 =Z COORDINATE POINT 1 (ALTITUDE IN FEET)
C                      POS2(3)=COORDINATES POINT 2
C                           1 =X COORDINATE POINT 2 (UNITS=MILES)
C                           2 =Y COORDINATE POINT 2 (UNITS=MILES)
C                           3 =Z COORDINATE POINT 2 (ALTITUDE IN FEET)
C
C**OUTPUT PARAMETERS:  HDN=COMPASS AZIMUTH FROM POS1 TO POS2--HEADING
C                         (RADIANS FROM NORTH)(+Y AXIS IS NORTH)
C                      D1=GROUND DISTANCE FROM POS1 TO POS2 (Z1=Z2=0)
C                         (UNITS=MILES)
C                      D2=SHORTEST DISTANCE BETWEEN POS1 AND POS2
C                         (UNITS=MILES)
C
```

```
      FUNCTION IBFCU (LOCN,KEY,NKEY,NDEX)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION ATTEMPTS TO LOCATE LOCN AMONG THE KEYWORDS IN
C            KEY.ONLY THE NON-BLANK PART OF LOCN IS COMPARED WITH THE LE
C            CHARACTERS OF KEY.THUS THE USER MAY ABBREVIATE THE KEYWORDS
C
```

```
C**INPUT PARAMETERS:   LOCN=CHARACTER VARIABLE REPRESENTING THE KEYWORD
C                           TO BE LOCATED
C                      KEY=CHARACTER ARRAY OF KEYWORDS TO BE SEARCHED
C                      NKEY=TOTAL NUMBER OF KEYWORDS IN KEY AND THE
C                           LENGTH OF THE NDEX ARRAY
C
C**INPUT/OUTPUT PAR:   NDEX=INTEGER CROSS REFERENCE ARRAY.IBFCU USES A
C                           BINARY SEARCH TO LOCATE LOCN,BUT THE USER
C                           NEED NOT ARRANGE KEY IN THE CORRECT
C                           COLLATING ORDER.ON THE FIRST CALL,SET
C                           NDEX(1)=0.IBFCU WILL COLLATE KEY BY STORING
C                           THE CORRECT INDECES IN NDEX.DO NOT CHANGE
C                           NDEX BETWEEN CALLS.
C
C**OUTPUT PARAMETERS:  IBFCU=-1 -LOCN AMBIGUOUS
C                            0 -LOCN NOT FOUND IN KEY
C                            I -LOCN EQUALS KEY(I)
C
```

```
      FUNCTION IFCU (LOCN,KEY,NKEY)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION ATTEMPTS TO LOCATE LOCN AMONG THE KEYWORDS
C            IN KEY.ONLY THE NON-BLANK PART OF LOCN IS MATCHED WITH THE
C            LEFMOST CHARACTERS IN KEY.THUS,THE USER MAY ABREVIATE
C            THE KEYWORDS.IFCU USE A LINEAR SEARCH,SEE ALSO IBFCU.
C
C**INPUT PARAMETERS:   LOCN=THE KEYWORD TO BE LOCATED (CHARACTER)
C                      KEY(NKEY)=AN ARRAY OF KEYWORDS TO SEARCH FOR
C                                LOCN (CHARACTER)
C
C**OUTPUT PARAMETERS:  IFCU= -1 - LOCN AMBIGUOUS
C                             0 - LOCN NOT FOUND IN KEY
C                             I - LOCN=KEY(I)
C
```

```
      SUBROUTINE INQIRU (LU,OPEND,FILEN,ACCS,FORMA,IERR)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE DETERMINES THE CURRENT STATUS OF A
C            SPECIFIED UNIT.
```

```
C
C**INPUT PARAMETERS:   LU=UNIT NUMBER FOR WHICH THE STATUS IS BEING
C                         CHECKED
C
C**OUTPUT PARAMETERS:  OPEND=LOGICAL VARIABLE DESIGNATING WHETHER OR
C                           NOT A UNIT NUMBER IS CONNECTED TO A FILE
C                           (=.TRUE. IS CONNECTED TO A FILE,=.FALSE.
C                           IS NOT CONNECTED TO A FILE)
C                      FILEN=CHARACTER VARIABLE REPRESENTING THE NAME
C                           OF THE FILE CONNECTED TO THE UNIT NUMBER
C                      ACCS=A CHARACTER VARIABLE INDICATING THE ACCESS
C                           METHOD OF THE FILE
C                      FORMA=A CHARACTER VARIABLE INDICATING WHETHER
C                           THE FILE IS OPENED FOR FORMATTED I/O
C                           OPERATIONS OR UNFORMATTED I/O OPERATIONS
C                      IERR=SYSTEM ERROR CODE.EQUALS ZERO IF NO ERROR
C                           OCCURS.
C
```

```
      SUBROUTINE INSCLU (M,N,LCOL,MFIRST,IARRAY,JCFILL,IERR)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO INSERT A COLUMN OF VALUES INTO
C            AN ARRAY.THIS SUBROUTINE IS USED WITH ARRAYS THAT ARE
C            UTILIZING A POINTER (MFIRST) THAT POINTS TO THE NEXT
C            AVAILABLE COLUMN IN THE ARRAY WHERE VALUES MAY BE LOADED.
C            BEFORE THIS SUBROUTINE IS CALLED THE USER SHOULD CALL
C            SUBROUTINE SETVU TO SET UP THE POINTER STRUCTURE.
C
C**INPUT PARAMETERS:   M=NUMBER OF ROWS IN IARRAY
C                      N=NUMBER OF COLUMNS IN IARRAY
C                      LCOL(M-1)=VECTOR OF VALUES TO BE LOADED INTO A
C                                COLUMN OF IARRAY (NO VALUE FOR ROW M
C                                TO AVOID DESTROYING POINTER SYSTEM)
C
C--INPUT/OUTPUT PARAMETERS:
C                      MFIRST=FIRST AVAILABLE COLUMN FOR STORING VALUES
C                      IARRAY(M,N)=ARRAY THAT WILL HAVE A COLUMN OF
C                                  VALUES LOADED INTO IT.A -1 WILL BE
C                                  LOADED INTO ROW M OF THE COLUMN
C                                  BEING FILLED,TO SHOW THAT THE COLUMN
C                                  IS FILLED.
C
C**OUTPUT PARAMETERS:  JCFILL=THE NUMBER OF THE COLUMN THAT WAS FILLED
C                      IERR=ERROR FLAG
C                          =0 NO ERROR
C                          =1 IARRAY IS FULL (MFIRST=0)
C
```

```
C**OUTPUT PARAMETERS:  JCFILL=THE NUMBER OF THE COLUMN THAT WAS FILLED
C                      IERR=ERROR FLAG
C                          =0 NO ERROR
C                          =1 IARRAY IS FULL (MFIRST=0)
C
```

```
      FUNCTION ISUMU (IARY,NA)
C
C**MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C**PURPOSE:  TO SUM THE ELEMENTS OF AN ARRAY.
C
C**INPUT PARAMETERS:   IARY(NA)=ARRAY WHOSE ELEMENTS ARE TO BE SUMMED
C
```

```
      SUBROUTINE LOADCU (CVAL,M,N,MROW,CARRAY)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LOAD ONE ROW OF A CHARACTER
C            ARRAY CARRAY WITH THE VALUE STORED IN CVAL.
C
C**INPUT PARAMETERS:   CVAL=THE CHARACTER VALUE TO BE LOADED INTO CARRAY
C                      M=NUMBER OF ROWS IN CARRAY
C                      N=NUMBER OF COLUMNS IN CARRAY
C                      MROW=THE ROW OF CARRAY TO BE LOADED
C
C**INPUT/OUTPUT PAR:   CARRAY=THE CHARACTER ARRAY TO BE LOADED
C                             PARTIALLY OR COMPLETELY.RETURNED AS
C                             THE LOADED ARRAY.
C
```

```
      SUBROUTINE LOADIU (IVAL,M,N,MROW,IARRAY)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LOAD ONE ROW OF AN INTEGER
C            ARRAY IARRAY WITH THE VALUE STORED IN IVAL.
C
```

```
C**INPUT PARAMETERS:   IVAL=THE INTEGER VALUE TO BE LOADED IN IARRAY
C                      M=NUMBER OF ROWS IN IARRAY
C                      N=NUMBER OF COLUMNS IN IARRAY
C                      NROW=THE ROW OF IARRAY TO BE LOADED
C
C**INPUT/OUTPUT PAR:   IARRAY=THE INTEGER ARRAY TO BE LOADED
C                             PARTIALLY OR COMPLETELY.RETURNED AS
C                             THE LOADED ARRAY.
C
```

```
      SUBROUTINE LOADRW (RVAL,M,N,NROW,RARRAY)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LOAD ONE ROW OF AN REAL ARRAY
C            RARRAY WITH THE VALUE STORED IN RVAL.
C
C**INPUT PARAMETERS:   RVAL=THE REAL VALUE TO BE LOADED INTO RARRAY
C                      M=NUMBER OF ROWS IN RARRAY
C                      N=NUMBER OF COLUMNS IN RARRAY
C                      NROW=THE ROW OF RARRAY TO BE LOADED
C
C**INPUT/OUTPUT PAR:   RARRAY=THE REAL ARRAY TO BE LOADED
C                             PARTIALLY OR COMPLETELY.RETURNED
C                             AS THE LOADED ARRAY.
C
```

```
      FUNCTION LPOSU (CHR)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION IS USED TO CODE ALPHANUMERIC CHARACTER
C            INTO AN INTEGER REPRSSENTATION OF THE CHARACTER.THE NUMBERS
C            1-26 REPRESENT THE CHARACTERS A-Z AND THE NUMBERS 27-36
C            REPRESENT THE CHARACTERS 0-9.
C
C**INPUT PARAMETERS:   CHR=CHARACTER STRING TO BE CODED AS AN INTEGER
C
C**OUTPUT PARAMETERS:  LPOSU=N :INTEGER CODE VALUE FOR THE CHARACTER
C                             (CHR)
C
```

```
      SUBROUTINE NBLCU (STRING,LO1,LO2)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE RETURNS THE FIRST AND LAST NON-BLANK
C            CHARACTER POSITIONS OF A STRING.
C
C**INPUT PARAMETERS:   STRING=THE STRING TO DELINEATE (CHARACTER)
C
C**OUTPUT PARAMETERS:  LO1,LO2=THE POSITION OF THE FIRST AND LAST NON-
C                              BLANK CHARACTER IN STRING.IF STRING IS
C                              ALL BLANK,LO1=LO2=0
C
```

```
      FUNCTION NCLIPU (INCLIP,LOW,IHIGH)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS FUNCTION EVALUATES THE VARIABLE INCLIP TO DETERMINE
C            IF IT LIES WITHIN SOME SPECIFIED RANGE OF VALUES.IF IT
C            IS LARGER THAN THE UPPER BOUND,THE FUNCTION
C            RETURNS THE VALUE OF THE UPPER BOUND.IF IT IS SMALLER
C            THAN THE LOWER BOUND THE FUNCTION RETURNS
C            THE VALUE OF THE LOWER BOUND.IF IT LIES WITHIN THE RANGE
C            THE VALUE OF INCLIP IS RETURNED.
C
C**INPUT PARAMETERS:   INCLIP=VARIABLE TO BE EVALUATED ON THE RANGE
C                             FROM LOW TO IHIGH
C                      LOW=LOWER BOUND ON THE RANGE OF ACCEPTABLE
C                          VALUES FOR INCLIP
C                      IHIGH=UPPER BOUND ON THE RANGE OF ACCEPTABLE
C                            VALUES FOR INCLIP
C
C**OUTPUT PARAMETERS:  NCLIPU=VALUE OF THE FUNCTION RETURNED.IF INCLIP
C                             > IHIGH THEN NCLIPU=IHIGH.IF INCLIP <
C                             LOW THEN NCLIPU=LOW.OTHERWISE NCLIPU=
C                             INCLIP.
C
```

```
      SUBROUTINE OPENDU (LU,FILEN,STAT,FM,NREC,IERR,*)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE OPENS A DIRECT ACCESS FILE
C
C**INPUT PARAMETERS:   LU=LOGICAL UNIT NUMBER OF THE FILE TO BE OPENED
C                      FILEN=A CHARACTER EXPRESSION REPRESENTING THE
C                            FILE NAME TO BE OPENED
C                      STAT=A CHARACTER EXPRESSION REPRESENTING THE
C                           FILE STATUS (OLD,NEW,SCRATCH,UNKNOWN)
C                      FM=A CHARACTER EXPRESSION DESIGNATING WHETHER
C                         THE FILE IS FORMATTED OR UNFORMATTED
C                      NREC=RECORD LENGTH
C
C**OUTPUT PARAMETERS:  IERR=THE ERROR NUMBER (=0 IF NONE FOUND)
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN OCCURS WHEN IERR
C                      IS NOT EQUAL ZERO.
C
```

```
      SUBROUTINE OPENSU(LU,FILEN,STAT,FM,IERR,*)
C
C--MODULE MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C--PURPOSE:  TO OPEN A SEQUENTIAL FILE
C
C--INPUT PARAMETERS:  LU-UNIT NUMBER
C                     FILEN-FILE NAME (CHARACTER VARIABLE)
C                     STAT-STATUS (CHARACTER)
C                     FM-FILE FORMAT (CHARACTER)
C
C--OUTPUT PARAMETERS:  IERR-CONDITION CODE
C                         0-NO ERROR
C                         .NE.0-SYSTEM ERROR CODE
C
C--ALTERNATE RETURNS:  0-NO ERROR
C                      1-ERROR, IERR.NE.0
C
```

```
      SUBROUTINE PAGEU (JFL,NL,MAXL,LINES,NPAGE)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO COUNT THE NUMBER OF LINES
C            PRINTED ON A PAGE AND ISSUE A PAGE EJECT IF NEEDED.CALL
C            PAGEU BEFORE WRITING NL MORE LINES ON THE FILE.
C
C**INPUT PARAMETERS:   JFL=THE FILE NUMBER (IF .LE. 0,RETURN WITH NO
C                          ACTION)
C                      NL=NUMBER OF NEW LINES TO BE ADDED
C                      MAXL=MAXIMUM NUMBER OF LINES/PAGE
C
C**INPUT/OUTPUT PAR:   LINE=THE NUMBER OF LINES ON A PAGE BEFORE NL
C                           ARE PRINTED (INPUT)
C                          =THE NUMBER OF LINES ON A PAGE AFTER NL LINES
C                           ARE PRINTED (IF A PAGE EJECT HAS BEEN ISSUED
C                           LINES WILL EQUAL NL+2) (OUTPUT)
C                      NPAGE=THE PAGE NUMBER.PAGEU WILL INCREMENT THE
C                            PAGE NUMBER AND PRINT A PAGE INDICATOR IF A
C                            PAGE EJECT IS ISSUED.IF NPAGE IS SENT
C                            NEGATIVE THIS FEATURE IS SUPPRESSED
C
```

```
      SUBROUTINE PRMU(LFD,MSG,NPRNT)
C
C--MODULE MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C--PURPOSE:  TO PRINT A MESSAGE ON A FILE.
C
C--INPUT PARAMETERS:  LFD-CONTROL PART OF FORMAT(CHARACTER)
C                        (E.G. '//4X')  UP TO 35 CHARACTERS.
C                     MSG-MESSAGE STRING TO PRINT(CHARACTER)
C                     NPRNT-FILE NUMBER TO WRITE TO. IF NPRNT.LE.0
C                           RETURN WITH NO PRINTING.
C
C
C--EXAMPLES:  CALL PRMU('1H1//5X','THIS IS A MESSAGE',6)
C             CALL PRMU('18HMESSAGE=','ANY STRING',6)
C             CALL PRMU('1X,116(1H-)',' ',6)
C
```

```
      SUBROUTINE PRMSU(MSG,NPRNT,NBC)
C
C--MODULE MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C--PURPOSE: TO PRINT A MESSAGE ON A FILE WITH NO TRAILING
C           BLANKS.
C
C--INPUT PARAMETERS: MSG-MESSAGE STRING (CHARACTER)
C                    NPRNT-FILE NUMBER. IF NPRNT.LE.0,
C                          RETURN WITH NO PRINTING
C
C--OUTPUT PARAMETERS: NBC-NUMBER OF LAST NON-BLANK CHARACTER
C                          IN MSG.
C
```

```
      SUBROUTINE PUTCU (CFROM,NF,I1,NT,CTO)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE TRANSFERS THE CONTENTS OF THE CHARACTER
C            VECTOR CFROM TO THE CHARACTER VECTOR CTO STARTING WITH
C            THE I1 ELEMENT OF CTO.IF THE TRANSFER LEADS TO AN OVERFLOW
C            ON CTO (TRANSFER EXCEEDS DIMENSION OF CTO) THE OVERFLOW
C            VALUES WILL NOT BE TRANSFERRED TO CTO.
C
C**INPUT PARAMETERS:   CFROM=CHARACTER VECTOR CONTAINING VALUES TO BE
C                           TRANSFERED TO CTO
C                      NF=NUMBER OF ELEMENTS OF CFROM
C                      I1=ELEMENT NUMBER OF CTO WHERE THE TRANSFER OF
C                         THE CONTENTS OF CFROM TO CTO WILL BEGIN
C                      NT=NUMBER OF ELEMENTS IN CTO
C
C**INPUT/OUTPUT PAR:   CTO=CHARACTER VECTOR TO WHICH THE CONTENTS OF
C                          CFROM WILL BE TRANSFERRED.RETURNED AS THE
C                          ORIGINAL VECTOR WITH THE TRANSFER MODIFI-
C                          CATIONS COMPLETED
C
```

```
      SUBROUTINE PUTIU (IFROM,NF,I1,NT,ITO)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE TRANSFERS THE CONTENTS OF THE INTEGER
C            VECTOR IFROM TO THE INTEGER VECTOR ITO STARTING WITH THE
C            I1 WORD OF ITO.IF THE TRANSFER LEADS TO AN OVERFLOW ON ITO
C            (TRANSFER EXCEEDS DIMENSION OF ITO) THE OVERFLOW VALUES
C            WILL NOT BE TRANSFERED TO ITO.
C
C**INPUT PARAMETERS:   IFROM=INTEGER VECTOR CONTAINING VALUES TO BE
C                            TRANSFERED TO ITO
C                      NF=NUMBER OF WORDS IN IFROM
C                      I1=WORD NUMBER OF ITO WHERE THE TRANSFER OF THE
C                         CONTENTS OF IFROM TO ITO WILL BEGIN
C                      NT=NUMBER OF WORDS IN ITO
C
C**INPUT/OUTPUT PAR:   ITO=INTEGER VECTOR TO WHICH THE CONTENTS OF
C                          IFROM WILL BE TRANSFERED TO.RETURNED AS THE
C                          ORIGINAL VECTOR WITH THE TRANSFER MODIFI-
C                          CATION COMPLETED
C
```

```
      SUBROUTINE PUTRU (RFROM,NF,I1,NT,RTO)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE TRANSFERS THE CONTENTS OF THE REAL VECTOR
C            RFROM TO THE REAL VECTOR RTO STARTING WITH THE I1 WORD OF
C            RTO.IF THE TRANSFER LEADS TO AN OVERFLOW ON RTO (TRANSFER
C            EXCEEDS DIMENSION OF RTO) THE OVERFLOW VALUES WILL NOT BE
C            TRANSFERRED TO RTO.
C
C**INPUT PARAMETERS:   RFROM=REAL VECTOR CONTAINING VALUES TRANSFERRED
C                            TO RTO
C                      NF=NUMBER OF WORDS IN RFROM
C                      I1=WORD NUMBER OF RTO WHERE THE TRANSFER OF THE
C                         CONTENTS OF RFROM TO RTO WILL BEGIN
C                      NT=NUMBER OF WORDS IN RTO
C
```

```
C**INPUT/OUTPUT PAR:   RTO=REAL VECTOR TO WHICH THE CONTENTS OF RFROM
C                          WILL BE TRANSFERRED TO.RETURNED AS THE
C                          ORIGINAL VECTOR WITH TRANSFER MODIFICATIONS
C                          COMPLETED
C
```

```
      SUBROUTINE RETRYU(JIN,JOUT,*,*)
C--MODULE: MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C--PURPOSE:
C      RETRYU WILL ASK IF THE USER WANTS TO RE-TRY WHATEVER
C      HE IS DOING. IF YES, TAKE ALTERNATE RETURN 1. IF NOT
C      USE RETURN 2.
C
C--INPUT PARAMETERS:
C      JIN-INPUT FILE NUMBER
C     JOUT-OUTPUT FILE NUMBER
C
C--ALTERNATE RETURNS
C      1-YES,RETRY
C      2-NO, NO NOT RETRY
```

```
     SUBROUTINE RMVCLU (JCOLN,M,N,MFIRST,IARRAY,IERR)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO MAKE A COLUMN OF AN ARRAY
C            AVAILABLE FOR LOADING VALUES INTO IT.THIS SUBROUTINE IS
C            USED WITH ARRAYS THAT ARE UTILIZING A POINTER (MFIRST)
C            TO THE FIRST AVAILABLE COLUMN IN THE ARRAY WHERE VALUES
C            MAY BE LOADED.
C
C**INPUT PARAMETERS:   JCOLN=COLUMN THAT IS TO BE MADE AVAILABLE
C                      M=NUMBER OF ROWS IN IARRAY
C                      N=NUMBER OF COLUMNS IN IARRAY
C
C**INPUT/OUTPUT PAR:   MFIRST=POINTER TO THE FIRST AVAILABLE COLUMN IN
C                            THE ARRAY THAT A COLUMN IS BEING MADE
C                            AVAILABLE IN.THIS SUBROUTINE UPDATES
C                            THIS POINTER WITH THE NUMBER OF THE
C                            COLUMN BEING REMOVED (JCOLN).
C                      IARRAY(M,N)=ARRAY IN WHICH A COLUMN IS BEING MADE
C                                  AVAILABLE FOR STORING OTHER VALUES.
C                                  COLUMN JCOLN OF ROW M WILL BE UPDATED
```

```
C
C**OUTPUT PARAMETERS:  IERR=ERROR FLAG
C                          =1 IF TRIED TO MAKE A COLUMN AVAILABLE THAT
C                           DOES NOT EXIST (COL # .GT. N .OR. COL # .LE.
C                           0)
C                          =2 IF COLUMN TO BE MADE AVAILABLE IS NOT
C                           CURRENTLY OCCUPIED
C
```

```
      FUNCTION RSUMU (RARY,NA)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  TO SUM THE ELEMENTS OF AN ARRAY.
C
C**INPUT PARAMETERS:   RARY(NA)=ARRAY WHOSE ELEMENTS ARE TO BE SUMMED
C
```

```
      CHARACTER*1 FUNCTION RYNU(JIN,JOUT)
C--MODULE: MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C--PURPOSE:
C      RYNU WILL QUERY THE USER FOR A YES OR NO RESPONSE.
C
C--INPUT PARAMETERS:
C      JIN-INPUT FILE NUMBER
C     JOUT-OUTPUT FILE NUMBER
C
C--OUTPUT PARAMETERS:
C      RYNU='Y' FOR YES
C           'N' FOR NO
```

```
      SUBROUTINE SETVU (INITV,M,N,IARRAY,MFIRST)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO INITIALIZE ALL THE ELEMENTS
C            EXCEPT ROW M OF IARRAY TO SOME VALUE (INITV).THE NEXT
C            AVAILABLE COLUMN POINTER IS SET UP IN ROW M AND MFIRST
C            (FIRST AVAILABLE COLUMN) IS INITIALIZED TO 1.SEE SUBROUTINE
C            INSCLU AND SUBROUTINE RMVCLU.
```

```
C
C**INPUT PARAMETERS:   INITV=THE VALUE TO INITIALIZE ALL THE ELEMENTS
C                            OF IARRAY TO
C                      M=NUMBER OF ROWS IN IARRAY
C                      N=NUMBER OF COLUMNS IN IARRAY
C
C**INPUT/OUTPUT PAR:   IARRAY(M,N)=ARRAY TO BE INITIALIZED.ROW M IS
C                                  WORKING STORAGE USED TO STORE POINTER
C                                  TO THE NEXT AVAILABLE COLUMN FOR
C                                  STORING VALUES.
C                      MFIRST=POINTER TO FIRST AVAILABLE COLUMN IN
C                             IARRAY
C
```

```
      SUBROUTINE VECU (ZPOS1,ZPOS2,HDN,D2,VELMAG,VEC)
C
C**MODULE:  MOPADS/UTIL
C**REFERENCE:  MOPADS VOLUME 5.9
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO CALCULATE THE COMPONENTS OF A
C            THREE DIMENSIONAL (X,Y,Z) VELOCITY VECTOR.THIS SUBROUTINE
C            USES OUTPUT FROM SUBROUTINE HEADNU TO CALCULATE THE
C            COMPONENTS OF THE VELOCITY VECTOR.
C
C**INPUT PARAMETERS:   ZPOS1=THE Z COORDINATE OF THE INITIAL POSITION
C                            (THE ALTITUDE MEASURED IN FEET)
C                      ZPOS2=THE Z COORDINATE OF THE FINAL POSITION (THE
C                            ALTITUDE MEASURED IN FEET)
C                      HDN=COMPASS AZIMUTH FROM INITIAL POINT TO FINAL
C                          POINT IN RADIANS FROM NORTH.(+Y AXIS IS NORTH
C                      D2=THE STRAIGHT LINE DISTANCE FROM THE INITIAL
C                         POSITION TO THE FINAL POSITION IN STATUTE
C                         MILES
C                      VELMAG=MAGNITUDE OF THE VELOCITY VECTOR (MILES/HR
C
C**OUTPUT PARAMETERS:  VEC(3)=THE COMPONENTS OF THE VELOCITY VECTOR
C                            1=X COMPONENT (MILES PER MINUTE)
C                            2=Y COMPONENT (MILES PER MINUTE)
C                            3=Z COMPONENT (FEET PER MINUTE)
C
```

VI. USER INSTRUCTIONS

1-0 LOADING ARRAYS.

1-1. Initializing Arrays.

Subroutines LOADCU, LOADIU, and LOADRU will initialize all or part of character, integer, and real arrays, respectively, with a single value. They are written formally to initialize a single row of a two dimensional array. This structure, however, permits initializing of all or part of both one and two dimensional arrays.

As an example, consider initializing the following integer arrays to zero.

```
DIMENSION   IRY(15,10), JRY(15)
```

a. `CALL LOADIU(0,15,10,3,IRY)`

b. `CALL LOADIU(0,1,15,1,JRY)`

c. `CALL LOADIU(0,1,15,1,IRY(1,4))`

d. `CALL LOADIU(0,1,15*10,1,IRY)`

e. `CALL LOADIU(0,1,5,1,JRY(11))`

Example (a) above sets row 3 of IRY to zero according to the description of LOADIU in Section V. In example (b), the array JRY is set to zero. The parameters imply that JRY is a two dimensional array with 1 row and 15 columns. The CALL to LOADIU initializes row 1 (the only row, of course). This example demonstrates how a one dimensional array can be initialized. It works because the elements of JRY are stored contiguously in memory by FORTRAN.

Example (c) is an extension of this idea to initialize column 4 of IRY. Since FORTRAN stores two dimensional arrays by columns, column 4 is contiguous in memory and element (1,4) is the location of its beginning. Example (c) treats the fourth column as a one by 15 two dimensional array and initializes the first row (which is in fact the fourth column of IRY).

Example (d) shows how to use this idea to initialize the entire IRY array with a single call to LOADIU. For this case, IRY is treated as a one by 150 two dimensional array.

Finally, the last five elements of JRY can be initialized with the statement in example (e). Here, JRY(11) is the first element to be initialized.

2-0 COPYING INFORMATION TO ARRAYS.

The Subroutine COPYCU, COPYIU, and COPYRU will copy data from one character, integer, or real array (respectively) to another. A second set of programs, PUTCU, PUTIU, and PUTRU, permit arrays to be copied with an offset to other arrays. For example,

```
CALL  PUTIU(KRY,4,16,20,LRY)
```

will copy elements one through four of KRY to elements 16 through 19 of LRY. LRY is of length 20.

3-0 OPENING AND CLOSING FILES.

Subroutines OPENDU and OPENSU will open direct access and sequential access files, respectively. The most commonly used (or required) parameters of the FORTRAN 77 OPEN statement are passed to these programs as formal parameters (i.e., unit number, file name, status, and format). The files are opened with the FORTRAN IOSTAT parameter and the resulting error code is returned through the formal parameter IERR. Thus, if a system I/O error occurs during opening of the file, the system error code is returned from OPENDU or OPENSU.

Subroutine INQUIRU can be used to execute a FORTRAN 77 INQUIRE statement. INQUIRU returns parameters indicating if a file is opened, its file name, access method, and format. If an I/O error occurs, its value is also returned.

4-0 TEXT I/O.

UTIL contains several utilities for printing messages to an interactive terminal and to read back responses. UTIL makes use of programs in the FFIN2 package (Polito) for this purpose.

Subroutine PRMU is a general purpose program for printing messages to a terminal. The calling sequence is as follows.

```
SUBROUTINE PRMU(LFD,MSG,NPRNT)
```

NPRNT is the unit number to write to. MSG is a CHARACTER expression which is the message to be written. LFD is a CHARACTER expression of up to 35 characters. LFD is a portion of a FORMAT statement to be executed before MSG is printed. For example,

```
CALL PRMU('5X','MESSAGE',6)
```

will skip five spaces before printing MESSAGE. A more complex example is

```
CALL PRMU('///2X,&MY NAME=','POLITO',10)
```

Subroutine PAGEU will count lines on a page and execute a page eject if insufficient space remains for the next section of output.

Subroutine NBLCU will delimit the non-blank portion of a CHARACTER variable. NBLCU will examine a CHARACTER variable and return the character position of its first and last non-blank character.

Subroutine PRMSU prints a text message to a file with no trailing blanks. Subroutine NBLCU can be used to find the last non-blank character.

Character function RYNU obtains a yes or no response from the terminal and returns a single character that is one of 'Y' or 'N'. The user may respond with any abbreviation of Yes or No (e.g., Y, YE, or YES). RYNU must be declared CHARACTER *1 in any program which uses it.

Subroutine RETRYU will ask the user if he/she wishes to re-try an operation which has failed. Two alternate returns are provided: one for yes and one for no.

5-0 SEARCHING CHARACTER ARRAYS.

Function IFCU will search a character array to match it with an input character expression. In other words, if the input character expression equals the fourth element in the character array, IFCU returns the value four. Only the non-blank portion of the input expression is compared, and it is matched with the leftmost characters of the array elements. Thus, the input text may abbreviate the array elements to any unique string. IFCU returns indications of success, failure, or ambiguous input strings. IFCU performs a linear search in the character array.

Function IBFCU has the same characteristic as IFCU except it performs a binary search in the character array. The user need not arrange the array in the computer's collating sequence, however. IBFCU will do this automatically by setting values in an index array (also provided by the user). IBFCU does not physically rearrange the character array. If the character array is long and if many searches will be made, IBFCU will be much faster than IFCU.

6-0 LOCATING PROGRAM UNITS IN FORTRAN 77.

Several utilities were written for the MOPADS project to aid in management of the FORTRAN 77 source code. Subroutine FSPU was used by these utility programs. FSPU is not called from any of the execution or user interface MOPADS programs.

FSPU examines a character string to determine if it is the first or last statement of a FORTRAN 77 program unit. FSPU recognizes the following: PROGRAM, SUBROUTINE, FUNCTION (all types), BLOCK DATA, and END (FSPU assumes that a line with only the character "END" is an END statement. It does not examine a subsequent line to see if it is a continuation statement which might cause the total statement to be ENDIF).

FSPU requires "normal" separation of the above FORTRAN statements. In other words it will recognize

```
CHARACTER*1 FUNCTION RYNU
```

but not

```
CHARACTER*1FUNCTIONRYNU
```

All of the MOPADS source code was developed with the P&A LAMP utility (Sutton, Hixson), so FSPU will also recognize "*DECK", and "*COMDECK" statements. Distributed versions of the MOPADS software do not contain these statements.

7-0 MANAGING AN ARRAY STACK.

UTIL contains programs that generalize the familiar notion of maintaining a stack of values onto which numbers are "PUSH"ed and from which they are "POP"ed. The stack which is maintained by UTIL programs is a stack of arrays.

Subroutine SETVU intializes the stack storage array. The user passes this array to SETVU as a parameter. The array (named IARRAY in Section V) has M rows and N columns. N must be the maximum size that the stack will ever grow to. Integer arrays of length M-1 may be stored in the stack. The M-th row is used by the utility programs to manage the storage. The array IARRAY and the variable MFIRST must not be changed by the user between calls to these programs.

After calling SETVU once, Subroutine INSCLU is called to "push" an array of M-1 values onto the stack. INSCLU returns the column number that the input array is stored in. The user program must "remember" this column number, because the "POP" operation is essentially random access. In other words, any column of IARRAY may be "POP"ed without regard to the order of insertion.

Subroutine RMVCLU does the "POP". It clears the specified column and makes it available for a new array to be stored with INSCLU. Note that RMVCLU does not return the information stored in the "POP"ed column. The user must access this information directly from the specified column of IARRAY.

8-0 A PSEUDO-COLLATING SEQUENCE.

MOPADS has need to code single characters into integers and vice versa. The UTIL package has two programs that do this in a way that does not depend upon the computer's collating sequence. Function LPOSU accepts a single character and returns an integer code. Only the capital letters and numbers may be coded. The letters A-Z are mapped to the integers 1-26, and the digits 0-9 are mapped to the numbers 27-36, respectively

Character Function CPOSU performs the reverse operation. It accepts an integer in the range 1-36 and returns a single character A-Z, 0-9, respectively.

9-0 DISTANCE, HEADING, SPEED.

Subroutine HEADNU computes the distance between two points. Given two points, each consisting of x, y, and z coordinates, HEADNU computes the compass azimuth from the first to the second (radians from north), the straight line distance between them (miles), and the ground distance between their projections on the z=0 plane (miles).

Subroutine VECU computes the components of a velocity vector. The input data are the z coordinates of two points, the heading from one to the other, the straight line distance from the first point to the second, and the magnitude of the velocity vector from the first to the second. The outputs are the components of the velocity vector in the x, y, and z directions.

10-0 MISCELLANEOUS.

In addition to the above, UTIL contains programs to compare character strings (CEQVU), compute exponentially smoothed values (EXSMU), sum the elements of an array (ISUMU, RSUMU), and clip an integer expression (NCLIPU). The use of these programs is explained in Section V.

VII. ERROR PROCESSING

The UTIL programs have no centralized error processing. Where appropriate, each program in UTIL performs its own error processing. See Section V and VI.

VIII. COMMON VARIABLE DEFINITIONS

The UTIL programs contain a single COMMON block which is initialized in BLOCK DATA BLOCKU. BLOCKU must be loaded with any programs using UITL. The COMMON block is labeled COMØ1U.

Variable	Value	Defintion
CHARS(36)*1	The letters A-H and the digits 0-9	This character array contains the characters coded by LPOSU and CPOSU in the correct pseudo-collating order.

IX. REFERENCES

1. Polito, J. MOPADS free-format input programs (MOPADS/FFIN2) (MOPADS Vol. 5.14). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft.Bliss, TX, April 1983.

2. Sutton, R. D., Hixson, A. F. Library management program (LAMP) users guide. Pritsker & Associates, Inc., PO Box 2413, West Lafayette, IN, August 1980.

X. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

XI. CHANGE NOTICES

APPENDIX R

MOPADS FINAL REPORT:

HUMAN FACTORS MODERATOR FUNCTIONS

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are HAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

ADDITIONAL MOPADS TERMINOLOGY

SKILLS	Components of human task performance which are required for successful completion of the task.
SKILL MODERATOR FUNCTION	FORTRAN subroutines which dynamically alter simulated skill performance during the course of a simulation run.
TASK VECTOR or TASK SPECIFIC CHARACTERISTIC	A list of data that is associated with a task element rather than with the operator who performs the tasks, e.g., the energy expenditure required to perform the task elements.

I. PURPOSE

The purpose of the software described herein is to make the human operator models 'behave' more like humans when variables known to affect human performance are considered. In previous SAINT modeling efforts, parameters associated with human performance, such as time to perform and probability of successful performance, were treated simply as random variables with a known distribution and distribution parameters (e.g., normal distribution with a mean of 2.0 and a standard deviation of 0.60). We know, however, that variability around the mean of human performance is not simply random. This variability is, by and large, caused by a variety of factors which affect the human's ability to perform his tasks. Obvious examples of these factors are environmental stressors (e.g., heat, noise), operator variables (e.g., intelligence, training), and task variables (e.g., workload, task difficulty).

The problems faced when developing causal relationships between these factors affecting performance and corresponding human performance are many and varied. To be of use to a simulation modeler, these relationships must be represented mathematically. With some notable exceptions, the human factors literature is relatively weak in documenting causal relationships. In cases where causal claims are made, they are not usually represented mathematically. A classic example of this is a report describing a study where regression analysis is used to positively link human performance to some variable, yet all that is reported is the significance level of the test and not the derived regression equation.

Therefore, our first task was to reformulate as much of the relevant human factors literature as possible into these mathematical relationships. This process is outlined in MOPADS reports 5.1, 5.2, 5.5, and 5.6 in considerable detail. To briefly summarize this process, the human factors literature was reviewed and each piece of literature was categorized with respect to the type of skill which was studied. The skill taxonomy which was used is presented in Table I-1. Furthermore, the articles were categorized with respect to how skill performance was measured (i.e.,

Table I-1
SKILL CATEGORIES

1	PROBABILITY-ESTIMATION
2	TIME-ESTIMATION
3	LTM-SENSORY
4	LTM-SYMBOLIC
5	STM-SENSORY
6	STM-SYMBOLIC
7	NUMERIC-MANIPULATION
8	RECOGNITION
9	UNUSED
10	UNUSED
11	TIME-SHARING
12	DETECTION
13	FINE-MANIPULATION
14	GROSS-MANIPULATION
15	UNUSED
16	RAL-PHYSICAL-EFFORT
17	TION-TIME
18	TRACKING
19	TEAM-COORDINATION

time to complete or probability of task completion was measured). Then, those articles which provided sufficiently detailed information were used to build mathematical relationships between the skill studied and the independent variable(s) studied. Curve fitting techniques were employed and many relationships quantitatively defined.

At this point in the project, we had a number of mathematical relationships describing how 'generic' skill performance was affected by a set of independent variables. To be useful to the simulation modeler, this information had to be translated into a form which could be readily incorporated into a task network simulation.

The approach selected involved the development of a set of skill moderator functions from the literature. These skill moderator functions take, as input, the mean time or probability of performing the task (which requires use of that skill) and the current values of the independent variables which will affect performance of this skill. Then, the moderator functions reestimate the mean time or probability based upon the values of the independent variables and provide this as output. The result is an estimate of mean performance based upon the state of the system as represented by the independent variables.

The task to the modeler is reduced to relatively few activities. First, he must identify the skills involved in each task. It is anticipated that some tasks will involve more than one skill. In this case, the modeler must decide the relative importance of each skill by assigning a percentage to each of the component skills of the task (they must add up to 100).

Second, the modeler must develop a data structure scheme to maintain the independent variables. In the MOPADS context, the information is maintained in the MOPADS data base. If these moderator functions are

used in another setting, the data may be maintained in any form so long as they are accessible to the moderator function module in a standard format. To accomplish this, the analyst must build the following three subroutines to transfer the values of the independent variables to the moderator function subroutines:

1. GETEVA
2. GETOVA
3. GETTSA
4. TIMEA

More detail on the parameters which are passed to and from these subroutines will be covered in Section IV of this report.

Finally, the modeler must write a small amount of software to identify the skills involved in the task currently being executed by the model, call the moderator function subroutines for these skills, and then combine the new estimates of the mean (if multiple skills are required by the task) into a single weighted estimate.

The remainder of this report will focus on the skill moderator functions. Only Section IV will address any details of external requirements of the subroutines. It should however, be strongly emphasized that these moderator functions are not MOPADS specific. If the interfaces discussed above are developed, these moderator functions can be used in any task network simulation. Their utility is only limited by the extent to which the skill categories are appropriate for the tasks being modeled. We anticipate that most tasks which are low-level cognitive, procedural, and/or physical in nature can be appropriately modeled with these subroutines. On the other hand, tasks which are predominantly cognitive (e.g., problem solving) or require predominantly highly skilled psychomotor behavior may be better modeled with other techniques.

It is also worthy to note that these moderator functions should not be viewed as 'sacred cows not to be fooled with.' In fact, we hope that the moderator functions can continue to develop and grow with the state-of-the-art in mathematical modeling of human performance. Special care has been given to document every segment of code which affects modeled human performance so that the source of that mathematical relationship can be determined. During the course of developing these subroutines, many decisions had to be made as to which model was best and, therefore, should be included. Others may justifiably dispute these judgments. We encourage anyone wishing to change these subroutines to reflect their judgment and/or to reflect new findings in human performance to do so. We ask only that, if possible, these changes be forwarded to the authors so that we may continue to improve the fidelity of the SAIL moderator subroutines.

II. DATA STRUCTURE DESCRIPTION

Since the moderator functions are a set of more or less unrelated subprograms, they have no major internal data structure. The exception to this is the set of independent variables which are considered in determining moderated performance. However, the specific nature of where and how these variables are stored is not of concern to the moderator functions. Rather, it must be considered by those writing the interface subroutines GETEVA, GETOVA, and GETTSA to be discussed in Section IV.

In summary, these subroutines contain no COMMON blocks and all required data are passed as formal parameters.

III. OVERVIEW OF FLOW OF CONTROL

The moderator subprograms are called by user programs (in this case MOPADS). Each moderator function has the same calling sequence as described in Section VI. This standardization ensures that information is passed to and from the moderator functions in a uniform way.

The moderator functions return revised estimates of the task time or probability to complete. To do this, they may require access to the operator state vectors, the environmental state vectors, and/or various data that are particular to the task (called task vectors here). Four standard subprograms are called by the moderator functions to obtain this information. They are:

GETOVA - Get operator state vector information
GETEVA - Get environmental state vector information
GETTSA - Get task vector information
TIMEA - Get the current military time

In MOPADS, these programs are provided as database application programs and are described in MOPADS Volume 5.18. In other applications, these programs would have to be written to access the data structures created by the modular procedures for writing these interfaces as contained in Section VI.

Outputs from the moderator functions include a moderated mean, a moderated standard deviation, and a new distribution type. At this writing the moderator functions only affect the mean. This is because of a lack of data to make estimates of variability or distribution changes. The module, however, has been designed to accommodate such estimates in the future without changes to the information interface.

IV. EXTERNAL FILE USAGE

No external files are used.

V. SUBPROGRAM DESCRIPTIONS

The following pages include printouts for each of the 32 moderator subroutines. These subroutines are listed alphabetically in the following order:

1. Detection probability
2. Detection time
3. Fine manipulative ability probability
4. Fine manipulative ability time
5. General physical effort probability
6. General physical effort time
7. Gross manipulative ability probability
8. Gross manipulative ability time
9. Long term memory-sensory probability
10. Long term memory-sensory time
11. Long term memory-symbolic probability
12. Long term memory-symbolic time
13. Numeric manipulation probability
14. Numeric manipulation time
15. Probability estimation probability
16. Probability estimation time
17. Reaction time probability
18. Reaction time
19. Recognition probability
20. Recognition time
21. Short term memory-sensory probability
22. Short term memory-sensory time
23. Short term memory-symbolic probability
24. Short term memory-symbolic time
25. Team coordination probability
26. Team coordination time
27. Time estimation probability
28. Time estimation time
29. Time sharing probability
30. Timesharing time
31. Tracking probability
32. Tracking time

We should note that a number of these moderator subroutines do not contain any executable code. This

is due to either a lack of data in the human factors literature regarding that skill or our inability to locate these data within the available contract resources. Again, we encourage any individuals with information on these skills to forward this information to the authors.

The comment sections at the beginning of each subprogram describe all parameter and local variable definitions. Note that the MOPADS assigned suffix for this module is "H". All subprograms end with an "H".

```
C:
      SUBROUTINE DETEPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY-TO-DETECT MODERATORS
C          FOR AUDITORY AND/OR VISUAL TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECYOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(15), NEE(2), NTSK(2), NIV(6), P(6), XM(3)
      DIMENSION XE(2),XO(15),XTSK(2)
C
C**DEFINE LENGTH OF ENVIRONMENTAL, OPERATOR,
```

```
C  AND TASK STATE VECTORS
C
      DATA KE/2/
      DATA KO/15/
      DATA NT/2/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEE/6.1/
      DATA NEO/32.20.3.44.12.40.27.36.15.34.4.16.45.49.47/
      DATA NTSK/3.6/
C
C**DEFINE THE VECTOR ELEMENTS
C  XE(1)=NUMBER OF OPERATORS ON DUTY
C  XE(2)=AMBIENT DRY BULB TEMPERATURE IN DEGREES CENTIGRADE
C  XO(1)=THE NUMBER OF CONSECUTIVE NIGHTS FOR WHICH THE
C        DURATION OF THE OPERATOR'S SLEEP PER
C        NIGHT IS KNOWN
C  XO(2)=THE OPERATOR'S AVERAGE AMOUNT OF SLEEP PER NIGHT IN HOURS
C  XO(3)=TIME ON TASK IN HOURS
C  XO(4)=THE OBSERVER'S FIELD-OF-VIEW IN DEGREES
C  XO(5)=TARGET TYPE
C  XO(6)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C  XO(7)=TARGET/BACKGROUND CONTRAST RATIO
C  XO(8)=THE APPARENT CONTRAST/THRESHOLD CONTRAST RATIO
C  XO(9)=OBSERVER'S SEARCH AREA IN DEGREES
C  XO(10)=AIRCRAFT ALTITUDE ABOVE GROUND IN FEET
C  XO(11)=OPERATOR'S DAYS ON DUTY
C  XO(12)=USE OF BINOCULARS
C  XO(13)=OBSERVER OFFSET IN NAUTICAL MILES
C  XO(14)=LINE RESOLUTION ON A CRT
C  XO(15)=TARGET LOCATION ON SCREEN
C  XTSK(1)=TASK MODALITY
C  XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C
C**RETRIEVE VALUES FOR OPERATOR, ENVIRONMENTAL AND TASK STATE VECTORS
C
      CALL GETOVA (IDO, N, NEO, KO, XO)
      CALL GETEVA (IDE, N, NEE, KE, XE)
C
C**NTSK CONTAINS ELEMENT NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT,ITASK, NTS, NTSK, NT, XTSK)
C
C**INITIALIZE NME, NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XTSK(1)=TASK MODALITY
C
      IF (XTSK(1).EQ.1.0)GO TO 1
      IF (XTSK(1).EQ.10.0)GO TO 2
      IF (XTSK(1).EQ.11.0)GO TO 30
C
C**THE TARGET IS AUDITORY
C**XO(1)=THE NUMBER OF CONSECUTIVE NIGHTS FOR WHICH THE DURATION OF THE OPERA-
C        TOR'S SLEEP PER NIGHT IS KNOWN
C**XO(2)=THE OPERATOR'S AVERAGE AMOUNT OF SLEEP PER NIGHT IN HOURS
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 127
C
    1 NME=NME+1
      NIV(NME)=2
      IF (XO(1).EQ.1.0) THEN
```

```
         P(NME)=0.630-0.079*EXP(-0.133*XO(2)**2)
       ELSE
         P(NME)=0.434+0.058*XO(2)-0.004*(XO(2)**2)
       ENDIF
C
C**XO(3)=TIME ON TASK IN HOURS
C**XMIN=TIME ON TASK IN MINUTES
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 154
C
       NME=NME+1
       NIV(NME)=1
       XMIN=XO(3)*60.0
       P(NME)=0.937-0.005*XMIN+0.00005*(XMIN**2)
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR AUDITORY DETECTION
C
       CALL ABSPMH(DIST,NIV,NME,P,XM)
       GO TO 999
C
C**THE TARGET IS VISUAL
C**XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C**NPOS=THE OBSERVER-TO-TARGET POSITION AS AN INTEGER VARIABLE
C
     2 NPOS=NINT(XTSK(2))
       GO TO (3,4,5,28,29)NPOS
C
C**THIS IS A GROUND-TO-GROUND,VISUAL DETECTION TASK
C**XMIN=TIME ON TASK IN MINUTES
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 154
C
     3 NME=NME+1
       NIV(NME)=1
       XMIN=XO(3)*60.0
       P(NME)=0.876-0.009*XMIN+0.0001*(XMIN**2)
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR GROUND-TO-GROUND,VISUAL DETECTIONN
C
       CALL ABSPMH(DIST,NIV,NME,P,XM)
       GO TO 999
C
C**THIS IS AN AIR-TO-GROUND,VISUAL DETECTION TASK
C**XO(4)=THE OBSERVER'S FIELD-OF-VIEW IN DEGREES
C**DATA FROM SCANLAN, 1976, PAGE 67
C
     4 NME=NME+1
       NIV(NME)=1
       P(NME)=0.556+0.022*XO(4)-0.002*(XO(4)**2)
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR AIR-TO-GROUND VISUAL DETECTION
C  TASK
C
       CALL ABSPMH(DIST,NIV,NME,P,XM)
       GO TO 999
C
C**THIS IS A GROUND-TO-AIR, VISUAL DETECTION TASK
C**XO(5)=TARGET TYPE
C**NTARG=TARGET TYPE AS AN INTEGER VARIABLE
C**XO(6)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C**XKM=HORIZONTAL RANGE TO TARGET IN KILOMETERS
C**DATA FROM WRIGHT, 1966, PAGES 16, 17, AND 13.
C
     5 NTARG=NINT(XO(5))
       XKM=XO(6)*1.852
       IF (XO(5).GT.10.0)GO TO 17
       NME=NME+1
       NIV(NME)=2
```

```
      GO TO (6,7,8,9,10,11,12,13,14,15)NTARG
C
C**TARGET IS AN F-4C
C
    6 P(NME)=-0.401+1.447*(-0.002*XKM**2)
      GO TO 17
C
C**TARGET IS AN F-100
C
    7 P(NME)=0.071+0.986*EXP(-0.009*XKM**2)
      GO TO 17
C
C**TARGET IS A T-33
C
    8 P(NME)=0.034+0.980*EXP(-0.013*XKM**2)
      GO TO 17
C
C**TARGET IS ANOTHER TYPE OF JET
C
    9 P(NME)=0.057+0.99?*EXP(-0.008*XKM**2)
      GO TO 17
C
C**TARGET IS A U-1A
C
   10 P(NME)=-0.350+1.421*EXP(-0.003*XKM**2)
      GO TO 17
C
C**TARGET IS A U-6A
C
   11 P(NME)=0.008+1.143*EXP(-0.016*XKM**2)
      GO TO 17
C
C**TARGET IS ANOTHER TYPE OF PROPELLER AIRCRAFT
C
   12 P(NME)=-0.011+1.071*EXP(-0.009*XKM**2)
      GO TO 17
C
C**TARGET IS AN O-1A
C
   13 P(NME)=0.010+1.121*EXP(-0.022*XKM**2)
      GO TO 17
C
C**TARGET IS AN OH-23
C
   14 P(NME)=0.010+0.941*EXP(-0.019*XKM**2)
      GO TO 17
C
C**TARGET IS ANOTHER TYPE OF HELICOPTER
C
   15 P(NME)=0.019+0.937*EXP(-0.030*XKM**2)
C
C**XO(7)=TARGET/BACKGROUND CONTRAST RATIO
C**XO(8)=THRESHOLD TARGET/BACKGROUND CONTRAST RATIO
C**CCT=THE APPARENT CONTRAST/THRESHOLD CONTRAST RATIO
C**DATA FROM SUGARMAN, HAMMILL, AND DEUTSCHMAN, 1972, PAGE B-2
C
   17 NME=NME+1
      NIV(NME)=2
      CCT=XO(7)/XO(8)
      P(NME)=1.011-1.1736*EXP(-1.574*CCT**2)
C
C**XO(9)=OBSERVER'S SEARCH AREA IN DEGREES
C**XO(10)=AIRCRAFT ALTITUDE ABOVE GROUND IN FEET
C**XKYARD=HORIZONTAL DISTANCE TO TARGET IN THOUSANDS OF YARDS
C**DATA FROM WOKOUN, 1960, PAGES 25 TO 35
C
```

```
      NME=NME+1
      NIV(NME)=2
      XKYARD=XO(6)**2.025
      IF (XO(10).LT.1500.0)GO TO 22
      IF (XO(9).GT.180.0)GO TO 18
      IF (XO(9).LE.180.0.AND.XO(9).GT.90.0)GO TO 19
      IF (XO(9).LE.90.0.AND.XO(9).GT.45.0)GO TO 20
      IF (XO(9).LE.45.0)GO TO 21
C
C**TARGET ALTITUDE IS AT LEAST 1500 FEET
C**OBRERVER SEARCH AREA GREATER THAN 180 DEGREES
C
   18 P(NME)=0.049+0.783*EXP(-0.145*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS AT LEAST 500 FEET
C**OBSERVER SEARCH AREA 91 TO 180 DEGREES
C
   19 P(NME)=0.058+0.795*EXP(-0.165*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS AT LEAST 1500 FEET
C**OBSERVER SEARCH AREA 46 TO 90 DEGREES
C
   20 P(NME)=0.033+0.668*EXP(-0.057*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS AT LEAST 1500 FEET
C**OBSERVER SEARCH AREA LESS THAN 46 DEGREES
C
   21 P(NME)=-0.115+0.961*EXP(-0.056*XKYARD**2)
      GO TO 27
   22 IF (XO(9).GT.180.0)GO TO 23
      IF (XO(9).LE.180.0.AND.XO(9).GT.90.0)GO TO 24
      IF (XO(9).LE.90.0.AND.XO(9).GT.45.0)GO TO 25
      IF (XO(9).LE.45.0)GO TO 26
C
C**TARGET ALTITUDE IS LESS THAN 1500 FEET
C**OBSERVER SEARCH AREA GREATER THAN 180 DEGREES
C
   23 P(NME)=0.019+0.733*EXP(-0.082*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS LESS THAN 1500 FEET
C**OBSERVER SEARCH AREA 91 TO 180 DEGREES
C
   24 P(NME)=0.030+0.709*EXP(-0.105*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS LESS THAN 1500 FEET
C**OBSERVER SEARCH AREA 46 TO 90 DEGREES
C
   25 P(NME)=0.039+0.607*EXP(-0.039*XKYARD**2)
      GO TO 27
C
C**TARGET ALTITUDE IS LESS THAN 1500 FEET
C**OBSERVER SEARCH AREA LESS THAN 46 DEGREES
C
   26 P(NME)=0.046+0.790*EXP(-0.055*XKYARD**2)
C
C**XO(11)=OPERATOR'S DAYS ON DUTY
C**DATA FROM SUGARMAN, HAMMILL, AND DEUTCHMAN, 1972, PAGE C-6
C
   27 NME=NME+1
      NIV(NME)=1
      P(NME)=0.739+0.807*EXP(-1.0*XO(11)**2)
```

```
C
C**XO(12)=USE OF BINOCULARS
C**DATA FROM WRIGHT, 1966, PAGE 14
C
      NM2=NME+1
      NIV(NME)=2
      IF (XO(12).EQ.0.0) THEN
         P(NME)=0.04+1.0328*EXP(-0.009*XKM**2)
      ELSE
         P(NME)=0.065+0.997*EXP(-0.011*XKM**2)
      ENDIF
C
C**XO(13)=OBSERVER OFFSET IN NAUTICAL MILES
C**XMETER=OBSERVER OFFSET IN METERS
C**DATA FROM WRIGHT, 1966, PAGE 16
C
      NME=NME+1
      NIV(NME)=2
      XMETER=XO(13)*1852.0
      IF (XMETER.LT.650.0)THEN
         P(NME)=0.051+0.969*EXP(-0.009*XKM**2)
      ELSE
         IF (XMETER.LT.1400.0)THEN
            P(NME)=0.075+0.987*EXP(-0.011*XKM**2)
         ELSE
            P(NME)=0.007+1.044*EXP(-0.010*XKM**2)
         ENDIF
      ENDIF
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR GROUND-TO-AIR,
C  VISUAL DETECTION TASKS
C
      CALL A**PMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-AIR, VISUAL DETECTION TASK
C
   28 NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR AIR-TO-AIR,
C  VISUAL DETECTION TASKS
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS TASK INVOLVES DETECTING TARGETS ON A DISPLAY
C**XO(14)=LINE RESOLUTION ON A CRT
C**DATA FROM DATMAN, 1965, PAGE 9
C
   29 NME=NME+1
      NIV(NME)=1
      P(NME)=0.509+0.0002*XO(14)
C
C**XE(1)=NUMBER OF OPERATORS ON DUTY
C**DATA FROM BERGUM AND LEHR, 1962, PAGE 12
C
      NME=NME+1
      NIV(NME)=2
      IF (XE(1).EQ.1.0)THEN
         P(NME)=1.40-0.640*XO(3)+0.130*XO(3)**2
      ELSE
         P(NME)=1.0-0.041*EXP(0.549*XO(3)**2)
      ENDIF
C
```

```
C**AUDIO=PERCENT CORRECT DETECTION OF SCREEN
C**DATA FROM SIEGEL AND FEDERMAN, 1973, PAGE 12
C
      NME=NME+1
      NIV(NME)=1
      IF (XO(15).EQ.1.0)P(NME)=0.68
      IF (XO(15).EQ.2.0)P(NME)=0.89
      IF (XO(15).EQ.3.0)P(NME)=0.76
      IF (XO(15).EQ.4.0)P(NME)=0.77
      IF (XO(15).EQ.5.0)P(NME)=0.90
      IF (XO(15).EQ.6.0)P(NME)=0.81
      IF (XO(15).EQ.7.0)P(NME)=0.70
      IF (XO(15).EQ.8.0)P(NME)=0.80
      IF (XO(15).EQ.9.0)P(NME)=0.75
C
C**XE(2)=AMBIENT DRY BULB TEMPERATURE IN DEGREES CENTIGRADE
C**DATA CITED IN HANCOCK, IN PRESS, FIGURE 4
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=0.727+0.0148*XE(2)-0.0002*XE(2)**2
C
C**CALL ABSPMH(DIST,NIV,NME,P,XM)
C
      CALL ABSPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THE TARGET IS AUDITORY AND VISUAL
C**XMIN=TIME ON TASK IN MINUTES
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 154
C
   30 NME=NME+1
      NIV(NME)=1
      XMIN=XO(3)*60.0
      P(NME)=0.952-0.002*XMIN+0.00003*XMIN**2
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR DETECTING AUDITORY
C  AND VISUAL TARGETS
C
      CALL ABSPMH(DIST,NIV,NME,P,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE DETETH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 9.10 OF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME-TO-DETECT MODERATORS
C          FOR VISUAL DETECTION TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NMF - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(19), NEE(4), NTSK(1), NIV(10), T(10), XM(3)
      DIMENSION XE(4),XO(19),XTSK(1)
C
C**DEFINE LENGTH OF ENVIRONMENTAL, OPERATOR,
```

```
C  ADD DATA STATE VECTORS
C
      DATA KE/4/
      DATA KO/19/
      DATA NT/1/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEE/4,1,8,6/
      DATA NEO/3,12,40,44,19,27,37,38,39,17,50,5,,49,53,14,20,54,55,47/
      DATA NTSK/6/
C
C**DEFINE THE VECTOR ELEMENTS
C  XE(1)=AMBIENT NOISE LEVEL IN DB
C  XE(2)=DRY BULB TEMPERATURE IN DEGREES CENTIGRADE
C  XE(3)=VAPOR PRESSURE IN MB
C  XE(4)=NUMBER OF OPERATORS ON DUTY
C  XO(1)=OPERATOR'S TIME-ON-TASK IN HOURS
C  XO(2)=TARGET TYPE
C  XO(3)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C  XO(4)=OPERATOR'S FIELD OF VIEW IN DEGREES
C  XO(5)=TARGET BACKGROUND COMPLEXITY
C  XO(6)=TARGET/BACKGROUND CONTRAST RATIO
C  XO(7)=TARGET HEIGHT IN FEET
C  XO(8)=TARGET WIDTH IN FEET
C  XO(9)=TARGET DEPTH IN FEET
C  XO(10)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C  XO(11)=BACKGROUND NONTARGET HEIGHT IN FEET
C  XO(12)=BACKGROUND NONTARGET WIDTH IN FEET
C  XO(13)=DISPLAY RESOLUTION IN LINES
C  XO(14)=DISTANCE TO DISPLAY IN FEET
C  XO(15)=TARGET COLOR
C  XO(16)=NUMBER OF BACKGROUND CHARACTERS
C  XO(17)=DISPLAY HEIGHT IN FEET
C  XO(18)=DISPLAY WIDTH IN FEET
C  XO(19)=LOCATION OF TARGET ON DISPLAY
C  XTSK(1)=THE OBSERVER-TO-TARGET POSITION
C
C**RETRIEVE VALUES FOR OPERATOR, ENVIRONMENTAL AND TASK STATE VECTORS
C
      CALL GETOVA (IDO, N, NEO, KO, XO)
      CALL GETEVA (IDE, N, NEE, KE, XE)
C
C**NTSK CONTAINS ELEMENT NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT,ITASK, NTS, NTSK, NT, XTSK)
C
C**INITIALIZE NME, NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XH
C**XTSK(1)=THE OBSERVER-TO-TARGET POSITION
C**NPOS=THE OBSERVER-TO-TARGET POSITION AS AN INTEGER VARIABLE
C
      NPOS=NINT(XTSK(1))
      GO TO (1,4,6,7,8)NPOS
C
C**THIS IS A GROUND-TO-GROUND, VISUAL DETECTION TASK
C**XO(1)=OPERATOR'S TIME-ON-TASK IN HOURS
C**XO(2)=TARGET TYPE
C**DATA FROM AINSWORTH AND BISHOP, 1971, PAGE 16
C
    1 IF (XO(2).LT.11.0.OR.XO(2).GT.13.0)GO TO 2
```

```
      NME=NME+1
      NIV(NME)=2
C
C**TARGET IS A TANK
C
      IF (XD(2).EQ.11.0)T(NME)=(25.333-0.167*XD(1))/60.0
C
C**TARGET IS A JEEP
C
      IF (XD(2).EQ.12.0)T(NME)=(28.000-0.800*XD(1))/60.0
C
C**TARGET IS TROOPS
C
      IF (XD(2).EQ.13.0)T(NME)=(34.000-1.000*XD(1))/60.0
C
C**XD(3)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C**XYARD=HORIZONTAL RANGE TO TARGET IN YARDS
C**DATA FROM KOBRICK, 1979, FIGURE 2
C
    2 XYARD=XD(3)*2025.367
      IF (XD(2).LT.11.0.OR.XD(2).G..25.0)GO TO 3
      IF (XD(2).GT.13.0.AND.XD(2).LT.25.0)GO TO 3
      IF (XD(2).EQ.13.0)GO TO 3
      NME=NME+1
      NIV(NME)=2
C
C**TARGET IS A TANK
C
      IF (XD(2).EQ.11.0)T(NME)=(2.115-1.6056*EXP(-0.0001*XYARD**2))/60.0
C
C**TARGET IS A JEEP
C
      IF (XD(2).EQ.12.0)T(NME)=(1.619-1.5699*EXP(-0.0004*XYARD**2))/60.0
C
C**TARGET IS AN ARMORED PERSONNEL CARRIER
C
      IF (XD(2).EQ.14.0)T(NME)=(0.063+0.014*XYARD)/60.0
C
C**TARGET IS A TRUCK
C
      IF (XD(2).EQ.15.0)T(NME)=(0.042+0.013*XYARD)/60.0
C
C**TARGET IS A SOLDIER
C
      IF (XD(2).EQ.25.0)T(NME)=(1.581+0.0016*EXP(0.0007*XYARD**2))/60.0
C
C**CALL ABSTMH OR RELTMH TO DERIVE SKILL MODERATORS FOR GROUND-TO-GROUND,
C  VISUAL DETECTION TASKS
C
    3 IF (NME.EQ.0)THEN
         NME=1
         NIV(NME)=1
         T(NME)=DIST(1)
         CALL RELTMH(DIST,NIV,NME,T,XM)
      ELSE
         CALL ABSTMH(DIST,NIV,NME,T,XM)
      ENDIF
      GO TO 999
C
C**THIS IS AN AIR-TO-GROUND,VISUAL DETECTION TASK
C**XD(4)=THE OBSERVER'S FIELD-OF-VIEW IN DEGREES
C**DATA FROM SCANLAN, 1976, PAGE 68
C
    4 NME=NME+1
      NIV(NME)=1
      T(NME)=(332.250-249.106*EXP(-0.002*XD(4)**2))/60.0
```

```
C
C**XO(5)=TARGET BACKGROUND COMPLEXITY
C**XO(6)=TARGET/BACKGROUND CONTRAST RATIO
C**DATA FROM SCANLAN, 1976, PAGE 22
C
      IF (XO(2).LT.11.0.OR.XO(2).GT.15.0)GO TO 5
      IF (XO(2).GT.11.0.AND.XO(2).LT.14.0)GO TO 5
      NME=NME+1
      NIV(NME)=3
C
C**TARGET IS A TANK
C
      IF (XO(2).EQ.11.0)T(NME)=(-55.327+32.149*XO(5)+32.538*XO(6))/60.0
C
C**TARGET IS AN ARMORED PERSONNEL CARRIER
C
      IF (XO(2).EQ.14.0)T(NME)=(81.303-2.300*XO(5)-36.077*XO(6))/60.0
C
C**TARGET IS A TRUCK
C
      IF (XO(2).EQ.15.0)T(NME)=(-22.331+30.799*XO(5)-0.385*XO(6))/60.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR AIR-TO-GROUND.
C  VISUAL DETECTION TASK
C
    5 CALL ABSTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS A GROUND-TO-AIR, VISUAL DETECTION TASK
C**XO(7)=TARGET HEIGHT IN FEET
C**XO(8)=TARGET WIDTH IN FEET
C**XO(9)=TARGET DEPTH IN FEET
C**XO(10)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C**XFEET=SLANT RANGE TO TARGET IN FEET
C**XYARD=AVERAGE OF THE TARGET'S THREE PLANAR AREAS IN SQUARE YARDS
C**XMARC=ANGLE THE TARGET SUBTENDS IN MINUTES OF ARC
C**DATA FROM HAMMILL, 1981, PAGE D-8
C
    6 NME=NME+1
      NIV(NME)=4
      XFEET=XO(10)*6076.1
      IF (XFEET.EQ.0.0)XFEET=0.00001
      XSYARD=((XO(7)*XO(8))+(XO(8)*XO(9))+(XO(7)*XO(9)))/3.0)/9.0
      XMARC=(XSYARD*((3000.0/XFEET)**2))/0.2964
      IF (XMARC.GT.450.0)XMARC=450.0
C
C**THE FOLLOWING LINE WAS INSERTED BECAUSE THE APPLE II HAS A LIMIT
C  ON THE SIZE OF EXPONENTS
C
      T(NME)=(0.422+61.84*EXP(-0.105*XMARC**2))/60.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR GROUND-TO-AIR,
C  VISUAL DETECTION TASKS
C
      CALL ABSTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-AIR, VISUAL DETECTION TASK
C
    7 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR AIR-TO-AIR,
C  VISUAL DETECTION TASKS
C
```

```
      CALL RELTMV(DIST,NIV NME,T,XM)
      GO TO 999
C
C**THIS TASK INVOLVES DETECTING TARGETS ON A DISPLAY
C**XO(11)=BACKGROUND NONTARGET HEIGHT IN FEET
C**XO(12)=BACKGROUND NONTARGET WIDTH IN FEET
C**XBINCH=DIAMETER OF BACKGROUND NONTARGETS IN INCHES
C**XTINCH=DIAMETER OF THE TARGET IN INCHES
C**XINCH=DIFFERENCE BETWEEN NONTARGET AND TARGET DIAMETERS
C**DATA FROM BLOOMFIELD, BECKWITH, EMERICK, MARMUREK, TEI,
C  AND TROUB, 1978, PAGE 2
C
    8 NME=NME+1
      NIV(NME)=2
      IF (XO(11).GT.XO(12))THEN
         IF (XO(12).EQ.0.0)XO(12)=0.00001
         XBINCH=XO(11)/XO(12)
      ELSE
         XBINCH=XO(12)/12.0
      ENDIF
      IF (XO(7).GT.XO(8))THEN
         XTINCH=XO(7)/12.0
      ELSE
         XTINCH=XO(8)/12.0
      ENDIF
      XINCH=XBINCH-XTINCH
      IF (XINCH.EQ.0.0)XINCH=0.00001
      T(NME)=(1/(XINCH**2))/60.0
C
C**XO(13)=DISPLAY RESOLUTION IN LINES ON A CRT
C**XO(14)=DISTANCE TO DISPLAY IN FEET
C**DATA FROM SCANLAN, 1976, PAGE 18
C
      NME=NME+1
      NIV(NME)=4
      IF (XO(14).EQ.0.0)XO(14)=0.00001
      XSYARD=((XO(7)*XO(8))+(XO(8)*XO(9))+(XO(7)*XO(9)))/3.0)/9.0
      XMARC=(XSYARD*((3000.0/XO(14))**2))/0.2964
      T(NME)=(105.324-2.579*XMARC-0.045*XO(13))/60.0
C
C**XO(15)=TARGET COLOR
C**XO(16)=NUMBER OF BACKGROUND CHARACTERS
C**DATA FROM BLOOMFIELD, BECKWITH, EMERICK,
C  MARMUREK, TEI, AND TROUB, 1978, PAGE 16
C
      IF (XO(15).LE.2.0)THEN
C
C**TARGET IS WHITE OR TAN
C
         NME=NME+1
         NIV(NME)=2
         T(NME)=(9.942-2.491*XO(15)+0.0004*XO(16))/60.0
      ENDIF
C
C**XO(17)=DISPLAY HEIGHT IN FEET
C**XO(18)=DISPLAY WIDTH IN FEET
C**XSINCH=SEARCH AREA IN SQUARE INCHES
C**DATA FROM BLOOMFIELD, BECKWITH, EMERICK,
C  MARMUREK, TEI, AND TROUB, 1978, PAGE 14
C
      XSINCH=(XO(17)*XO(18))/144.0
      IF (XO(15).LE.2.0)THEN
         NME=NME+1
         NIV(NME)=3
         T(NME)=(-2.555+1.366*XO(15)+0.020*XSINCH)/60.0
      ENDIF
```

```
C:
      SUBROUTINE PHASPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          FINE MANIPULATIVE ABILITY TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(1), NEE(1), NIV(1), P(1), XM(3)
      DIMENSION XO(1), XE(1)
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
```

```
      UNIN MEU/3/
      DATA MEE/7/
      DATA NIV/2/
      DATA KE,KO/1.1/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C     XO(1)=TIME ON TASK IN HOURS
C     XE(1)=VERTICAL VIBRATION IN HZ
C
C**RETRIEVE THE VALUES FOR THE STATE VECTORS
C
      CALL GETOVA (IDO,N,MEO,KO,XO)
      CALL GETEVA (IDE,N,MEE,KE,XE)
C
C**INITIALIZE NME
C
      NME=0
C
C**EVALUATE XM
C**DATA FROM PFEIFFER ET AL (1979)
C
      NME=NME+1
      IF(XE(1).GT.2.0) THEN
C
C** VIBRATION IS PRESENT
C
         P(NME)=1.0-((1.0-DIST(1))*(.718+.823*EXP(-.013*XO(1))))
C
      ELSE
C
C** NO VIBRATION IS PRESENT
C
         P(NME)=1.0-((1.0-DIST(1))*(1.23-.365*EXP(-.036*XO(1))))
      ENDIF
C
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE PHASTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE FINE MANIPULATIVE ABILITY
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C::
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NI(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(3), NEE(1), NIV(1), T(1), XM(3)
      DIMENSION XE(1), XO(3)
C
C**DEFINE THE LENGTH OF STATE VECTORS
C
```

```
      DATA KE,KO/1,3/
C
C**DEFINE ELEMENT NUMBERS
C
      DATA NEE/1/
      DATA NEO/67,66,65/
      DATA NIV/1/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C   XE(1)=AMBIENT (DRY BULB) TEMPERATURE
C   XO(1)=SKIN TEMPERATURE WHEN THE OPERATOR CHANGED ENVIRONMENTS
C         (OLD SKIN TEMPERATURE)
C   XO(2)=TIME IN THE CURRENT TEMPARATURE
C   XO(3)=CURRENT SKIN TEMPERATURE (COMPUTED FROM XO(1) AND XO(2))
C
C
C**RETRIEVE VALUES FOR THE STATE VECTORS
C
      CALL GETEVA(IDE,N,NEE,KE,XE)
      CALL GETOVA(IDO,N,NEO,KO,XO)
      NME=0
C
C
C**EVALUATE XM
C
      NME=NME+1
C
C**DATA FROM SIEGEL ET AL. 1979
C**FINAL SKIN TEMPERATURE EVALUATION FROM MCINTYRE (1980)
C**RATE OF SKIN TEMPERATURE CHANGE (1-EXP(.05*XO(2)) IS HYPOTHESIZED
C
C  EVALUATE SKIN TEMPERATURE (XO(3))
C
      XO(3)=XO(1)+((26.15+.235*XE(1))-XO(1))*(1-EXP(.05*XO(2)))
C
C  EVALUATE TIME BASED UPON SKIN TEMPERATURE
C
      T(1)=DIST(1)*(1.0/(1.012-.607*EXP(-.003*XO(3)**2)))
C
C  ENSURE THAT HE DOES NOT GO FASTER THAN AVERAGE BECAUSE HIS
C     HANDS ARE WARM
C
      IF (T(1).LT.DIST(1)) T(1)=DIST(1)
C
C**CALL RELTMM TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMM(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE SPEFPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPAD3/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          GENERAL PHYSICAL EFFORT TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), P(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE GENERAL
C**    PHYSICAL EFFORT TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE GPEFTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE GENERAL PHYSICAL EFFORT
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR .DO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(4), NEE(2), NTSK(1), NIV(3), T(3), XM(3)
      DIMENSION XE(2), XO(4), XTSK(1)
C
C**DEFINE THE LENGTH OF STATE VECTORS
C
```

```
      DATA XE,XO,NI/2,4,1/
C
C**DEFINE ELEMENT NUMBERS
C
      DATA NEE/1,8/
      DATA NEO/8,9,24,28/
      DATA NTSK/1/
      DATA NME,NIV/3,2,3,2/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C   XE(1)=AMBIENT (DRY BULB) TEMPERATURE
C   XE(2)=ATMOSPHERIC VAPOR PRESSURE IN MB
C   XO(1)=LENGTH OF OPERATOR'S PREVIOUS WORK SESSION
C   XO(2)=LENGTH OF OPERATOR'S PREVIOUS REST SESSION
C   XO(3)=HOURS OF SLEEP THE OPERATOR RECEIVED IN MOST RECENT SLEEP
C   XO(4)=DAYS (OR PART OF A DAY) SINCE THE OPERATOR LAST SLEPT
C   XTSK(1)=KCAL/MINUTE REQUIRED BY THE TASK
C
C
C**RETRIEVE VALUES FOR THE STATE VECTORS
C
      CALL GETEVA(IDE,N,NEE,KE,XE)
      CALL GETOVA(IDO,N,NEO,KO,XO)
      CALL GETTSA(ITASK,NTS,NTSK,NT,XTSK)
C
C
C**EVALUATE RM
C
C
C**FROM MCINTYRE (1980), PAGE 324 RELATING TO PHYSIOLOGICAL HEAT
C    EXPOSURE LIMIT (PHEL)
C**RELATIONSHIP BETWEEN TIME ON TASK, PHEL, AND TIME
C    TO PERFORM THE TASK WAS HYPOTHESIZED
C
C
C  COMPUTE METABOLIC WORK RATE FROM KCAL/MIN (APPROXIMATION)
C
      RM=3.0+32.1*XTSK(1)
C
C  COMPUTE WET BULB GLOBE TEMPERATURE
C
      WBGT=.567*XE(1) + .216*XE(2) + 3.38
C
C  ENSURE THAT RM AND WBGT ARE WITHIN ACCEPTABLE BOUNDS FOR THE
C     PHEL FORMULA TO BE VALID
C
      IF(RM.GT.85.0 .AND. RM.LT.150.0 .AND. WBGT.GT.31.0 .AND.
     +    WBGT.LT.50.)  THEN
         PHEL=(17.25*10.0**8 - 12.97*RM**10.0**6 + 18.61*(RM**2)
     +        *10.0**3) * (WBGT**(-.536))
C
C   AS THE OPERATOR APPROACHES HIS PHEL, HIS PERFORMANCE TIME SHOULD
C    INCREASE SLIGHTLY (CONSIDERING HIS RECENT REST)
C
         T(1)=DIST(1)*(1.0+(XO(1)-XO(2))/PHEL)
      ELSE
         T(1)=DIST(1)
      ENDIF
C
C
C** MODEL FROM PFEIFFER ET AL (1979)
C**  THE LINK BETWEEN REST REQUIREMENT AND PERFORMANCE IS HYPOTHESIZED
C
C  COMPUTE THE REST REQUIREMENT
C
      RESTR=((XO(1)-XO(2))*(XTSK(1)-4.5))/(XTSK(1)-1.5)
```

```
      IF (RESTR.LT.0.0) RESTR=0.0
      T(2)=DIST(2)*(1.0+.015*RESTR)
C
C
C** DATA FROM SIEGEL ET AL (1979)
C
C   COMPUTE THE FATIGUE LEVEL BASED UPON THE TIME SINCE THE
C     OPERATOR LAST SLEPT
C
      IF (XD(4).LT.0.21) FTG=0.366*XD(4)
      IF (XD(4).GE.0.2) .AND. XD(4).LE.0.83) FTG=-0.3+1.648*XD(4)
      IF (XD(4).GT.0.83) FTG=0.82+0.09*XD(4)
C
C   INCREASE THE FATIGUE LEVEL IF THE OPERATOR'S LAST SLEEP WAS
C     LESS THAN 8 HOURS
C
      IF(XD(3).GT.8.0) THEN
         FATRED=1.0
      ELSE
         FATRED=0.125*XD(3)
      ENDIF
      IF (FATRED.LT.0.001) FATRED=0.001
      FTG=FTG/FATRED
C
C  ENSURE THAT THE FATIGUE MEASURE DOES NOT EXCEED 1.0
C
      IF (FTG.GT.1.0) FTG=1.0
C
C  INCREASE THE OPERATOR'S TASK PERFORMANCE TIME
C
      T(3)=DIST(1)*(1.0+FTG)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE GRAPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          GROSS MANIPULATIVE TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), P(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(I)=0
C
C**EVALUATE IN
C**ND DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE ERRORS
C**    MANIPULATIVE TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPRM TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPRM(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE BRNATH(IDO, IDE, M, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE GROSS MANIPULATION
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       M - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(I)=U
C
C**EVALUATE IM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    GROSS MANIPULATION
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTIM TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTIM(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE LMSYPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          LONG TERM MEMORY/SENSORY TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(1) - SPILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       ND - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NI - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NED(ND) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(NE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NIN(NI) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NED(2), NIV(1), P(1), XM(3)
      DIMENSION ID(2)
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NED /98,64/
```

```
      DATA NIV/2/
      DATA KO/2/
C
C** THE VECTOR ELEMENTS ARE AS FOLLOWS
C     XO(1)=NUMBER OF TIMES THAT THE OPERATOR HAS PREVIOUSLY PERFORMED
C           THE TASK
C     XO(2)=OPERATOR'S SENSE OF DIRECTION
C
C
C**RETRIEVE THE VALUES OF THE STATE VECTORS
C
      CALL GETOVA (IDO,N,NEO,KO,XO)
C
C**INITIALIZE NME
C
      NME=0
C
C**EVALUATE XM
C
C**DATA FROM SIEGEL ET AL. 1979
C
      IF (XO(2).EQ.1) THEN
C
C   OPERATOR HAS A GOOD SENSE OF DIRECTION
C
         P(NME)=DIST(1)*(.414-.745*EXP(-.226*XO(1)**2))
      ELSE
C
C   OPERATOR HAS A POOR SENSE OF DIRECTION
C
         P(NME)=DIST(1)
      ENDIF
C
C**CALL RELPMI TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMI(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE LMSYTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE LONG TERM MEMORY/SENSORY
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(1) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       NO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(NO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(NE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(1) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(1) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
```

```
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    LONG TERM MEMORY/SYMBOLIC
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE LMSBPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          LONG-TERM MEMORY/SYMBOLIC TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(2), NEE(2), NIV(2), P(2), XM(3)
      DIMENSION XO(2), XE(2)
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
```

```
      DATA NEU/50.63/
      DATA NEE/4.9/
      DATA NIV/2.2/
      DATA KE,KO/2.2/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C     XO(1)=NUMBER OF TIMES THE OPERATOR HAS PREVIOUSLY PERFORMED
C           THE TASK
C     XO(2)=DAYS SINCE THE TASK WAS LAST PERFORMED
C     XE(1)=NOISE LEVEL
C     XE(2)=NOISE PREDICTABILITY
C
C**RETRIEVE THE VALUES FOR THE STATE VECTORS
C
      CALL GETOVA (IDO,N,NEO,KO,XO)
      CALL GETEVA (IDE,N,NEE,KE,XE)
C
C**INITIALIZE NME
C
      NME=0
C
C**EVALUATE XM
C**DATA FROM SIEGEL ET AL 1979
C
      NME=NME+1
      IF(XE(2).GT.0.0) THEN
C
C** NOISE IS UNPREDICTABLE
C
         P(NME)=DIST(1)*(1.01-0.0029*XE(1))
C
      ELSE
C
C** NOISE IS UNPREDICTABLE
C
         P(NME)=DIST(1)*(0.95-0.0043*XE(1))
      ENDIF
C
C** DATA FROM SIEGEL 1979
C
C   NOTE: WE ASSUME THAT IF THE HUMAN HAS PERFORMED THE TASK
C         CORRECTLY TEN TIMES, HE IS WELL TRAINED
C
      NME=NME+1
      IF(XO(1).GE.10.0) THEN
C
C**  OPERATOR IS WELL TRAINED
C
        P(NME)=DIST(1)*(.016+.356*EXP(-.155*XO(2)))
      ELSE
C
C**  OPERATOR IS POORLY TRAINED
C
        P(NME)=DIST(1)*(.069+0.87*EXP( .15*XO(2)))
      ENDIF
C
C
C**DATA FROM DAVEY(1973)
C
      NME=NME+1
      IF (XTSK(2).LT.2.0) THEN
         P(NME)=DIST(1)*(1+.025*XTSK(2))
      ELSE
         P(NME)=DIST(1)*(1.05-.025*XTSK(2))
      ENDIF
C
```

```
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE LMSBTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE LONG TERM MEMORY/SYMBOLIC
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(I)=0
C
C**EVALUATE NN
C***ND DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    LONG TERM MEMORY/SYMBOLIC
      NPE=NPE+1
      NIV(NPE)=1
      T(NPE)=DIST(I)
C
C**CALL FELTVH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL FELTVH(DIST,NIV,NPE,T,NN)
      RETURN
      END
```

```
C:
      SUBROUTINE NUMAPH(IDO. IDE. N. DIST. ITASK, NTS. XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          NUMERICAL MANIPULATIVE TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                    (1) MEAN
C                    (2) STANDARD DEVIATION
C                    (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C::
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N). IDE(N). DIST(3). ITASK(NTS)
      DIMENSION NIV(1). P(1). XM(3)
C
C**INITIALIZE NME.NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE NUMERIC
C**    MANIPULATIVE TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE HUMATH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:HOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE NUMERIC MANIPULATION
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(2), NEE(1), NIV(2), T(2), XM(3)
      DIMENSION XE(1), XO(2)
C
C**DEFINE THE LENGTH OF STATE VECTORS
C
```

```
      DATA NE,NO,NT/1,2,0/
C
C**DEFINE ELEMENT NUMBERS
C
      DATA NEE/6/
      DATA NEO/4,3/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C   XE(1)=MOISE LEVEL
C   XO(1)=DAYS OF DUTY
C   XO(2)=TIME ON TASK
C
C
C**RETRIEVE VALUES FOR THE STATE VECTORS
C
      CALL GETEVA(IDE,N,NEE,XE,XE)
      CALL GETOVA(IDO,N,NEO,XO,XO)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=2
      NIV(2)=1
      NIV(3)=1
C
C**EVALUATE XM
C
C
C**DATA FROM SIEGEL ET AL. 1979
C
      NME=NME+1
      TD=TIME(0)
      IF (TD.GE.1000) THEN
         TD=(TD-1000)/100
      ELSE
         TD=(TD+1400)/100
      ENDIF
      T(NME)=DIST(1)*(1.0+123.45/((TD*(1.65-.095*XO(1)))+
     +  (TD**2*(-.07+.005*XO(1)))+(120.9-.0085*XO(1))))
C
C
C**FROM MCCORMICK (EXACT RELATIONSHIPS ARE HYPOTHESIZED)
C
      NME=NME+1
      IF (XE(1).GT.80.0) THEN
         T(NME)=DIST(1)*2.0**((XE(1)-80)/30.0)
      ELSE
         T(NME)=DIST(1)
      ENDIF
C
C**HYPOTHESIZED RELATIONSHIP-LAUGHERY,1984
C
      NME=NME+1
      IF(XO(2).LT.8.0) THEN
         T(NME)=DIST(1)
      ELSE
         T(NME)=DIST(1)*((XO(2)-8.0)*0.05+1.0)
      ENDIF
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C
      SUBROUTINE PRESPH(IDO, IDE, M, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C           PROBABILITY ESTIMATION TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       M - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C                 XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                         MEAN
C                 XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                         STANDARD DEVIATION
C                 XM(3) = DERIVED DISTRIBUTION TYPE
C
C**
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), P(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE JM
C***NO DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE PROBABILITY
C**    ESTIMATION TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPWH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPWH(DIST,NIV,NME,P,JM)
      RETURN
      END
```

```
C
      SUBROUTINE PRESTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE PROBABILITY ESTIMATION
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE IM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    PROBABILITY ESTIMATION
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMM TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMM(DIST,NIV,NME,T,IM)
      RETURN
      END
```

```
C:
      SUBROUTINE REACPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 3.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY-TO-COMPLETE MODERATORS
C          FOR AUDITORY AND/OR VISUAL REACTION TIME TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS), XM(3)
      DIMENSION NIV(1), P(1)
C
C**INITIALIZE NME,NIV
C
```

```
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**THERE WERE NO DATA DESCRIBING REACTION TIME PROBABILITIES
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPRH TO DERIVE SKILL MODERATORS FOR REACTION TIME TASKS
C
      CALL RELPRH(DIST,NIV,NME,P,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE REACTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 3.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME-TO-COMPLETE MODERATORS
C          FOR AUDITORY AND/OR VISUAL REACTION TIME TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEE(1), NEO(4), NTSK(6), NIV(9), T(9), XM(3)
      DIMENSION XE(1), XO(4), XTSK(6)
C
C**DEFINE LENGTH OF ENVIRONMENTAL, OPERATOR, AND TASK STATE VECTORS
```

```
C
      DATA KE/1/
      DATA KO/4/
      DATA NT/6/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEE/7/
      DATA NEO/12,58,59,22/
      DATA NTSK/3,10,7,8,5,9/
C
C**DEFINE THE VECTOR ELEMENTS
C  XE(1)=VIBRATION IN HZ
C  XO(1)=TARGET TYPE
C  XO(2)=OPERATOR EXPERIENCE
C  XO(3)=SIGNAL PROBABILITY
C  XO(4)=SIGNALS PER MINUTE
C  XTSK(1)=TASK MODALITY
C  XTSK(2)=NUMBER OF ALTERNATIVE RESPONSES
C  XTSK(3)=DISTANCE TO SWITCH/KEY IN FEET
C  XTSK(4)=WIDTH OF SWITCH/KEY IN FEET
C  XTSK(5)=RESPONSE MODE
C  XTSK(6)=NUMBER OF DISPLAYS MONITORED
C
C**RETRIEVE VALUES FOR OPERATOR AND TASK STATE VECTORS
C
      CALL GETEVA (IDE, N, NEE, HZ, XE)
      CALL GETOVA (IDO, N, NEO, KO, XO)
C
C**NTSK CONTAINS ELEMENT NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT,ITASK, NTS, NTSK, NT, XTSK)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XTSK(1)=TASK MODALITY
C
      IF (XTSK(1).EQ.1.0)GO TO 1
      IF (XTSK(1).EQ.10.0)GO TO 2
      IF (XTSK(1).EQ.11.0)GO TO 3
C
C**THE TARGET IS AUDITORY
C
   1  NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
C  AUDITORY REACTION-TIME TASKS
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THE TARGET IS VISUAL
C**XTSK(2)=NUMBER OF ALTERNATIVE RESPONSES
C
    2 IF (XTSK(2).EQ.1)THEN
C
C**THIS IS A SIMPLE, VISUAL, REACTION-TIME TASK
C**XTSK(3)=DISTANCE TO SWITCH/KEY IN FEET
C**INCHD=DISTANCE TO SWITCH/KEY IN INCHES
```

```
C**XTSK(4)=WIDTH OF SWITCH/KEY IN FEET
C**XINCHW=WIDTH OF SWITCH/KEY IN INCHES
C**XID=INDEX OF DIFFICULTY
C**DATA FROM FITTS AND PETERSON, 1964, PAGE 110
C
          NME=NME+1
          NIV(NME)=2
          XINCHD=XTSK(3)/12.0
          XINCHW=XTSK(4)/12.0
          XID=(ALOG((2*XINCHD)/XINCHW))/0.693
          T(NME)=((74.0*XID)-57.0)/1000.0
C
C**DATA FROM FITTS AND PETERSON, 1964, PAGE 107
C
          NME=NME+1
          NIV(NME)=2
          T(NME)=(261.0+(5.4*XID))/1000.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR
C  SIMPLE, VISUAL, REACTION-TIME TASKS
C
          CALL ABSTMH(DIST,NIV,NME,T,XM)
          GO TO 999
        ENDIF
C
C**THIS IS A CHOICE, VISUAL, REACTION-TIME TASK
C**XTSK(5)=RESPONSE MODE
C
      IF (XTSK(5).EQ.10.0)THEN
C
C**TASK REQUIRES A VOICE RESPONSE
C**XO(1)=TARGET TYPE
C
          IF (XO(1).EQ.56.0)THEN
C
C**TASK REQUIRES VOICE RESPONSE TO A LIGHT
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 81
C
            NME=NME+1
            NIV(NME)=3
            T(NME)=(0.438+0.054*XTSK(2))/60.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR CHOICE, VISUAL,
C  REACTION-TIME TASKS WHICH REQUIRE A VOICE RESPONSE TO A LIGHT
C
            CALL ABSTMH(DIST,NIV,NME,T,XM)
            GO TO 999
          ENDIF
          IF (XO(1).EQ.57.0)THEN
C
C**TASK REQUIRES VOICE RESPONSE TO A DIGIT
C
            NME=NME+1
            NIV(NME)=3
            T(NME)=(0.419+0.(OLD)1*EXP(1.551*XTSK(2)))/60.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR CHOICE, VISUAL,
C  REACTION-TIME TASKS WHICH REQUIRE A VOICE RESPONSE TO A DIGIT
C
            CALL ABSTMH(DIST,NIV,NME,T,XM)
            GO TO 999
          ENDIF
C
C**TASK REQUIRES A VOICE RESPONSE TO A NON-LIGHT, NON-DIGIT STIMULUS
C
          NME=NME+1
```

```
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR CHOICE, VISUAL,
C  REACTION-TIME TASKS WHICH REQUIRE A VOICE RESPONSE TO A
C  NON-LIGHT, NON-DIGIT STIMULUS
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
    ELSE
C
C**TASK REQUIRES A MANUAL RESPONSE
C
      IF (XO(1).EQ.56.0)THEN
C
C**TASK REQUIRES A MANUAL RESPONSE TO A LIGHT
C
        NME=NME+1
        NIV(NME)=3
        T(NME)=(0.606-0.658*EXP(-0.309*XTSK(2)))/60.0
      ENDIF
      IF (XO(1).EQ.57.0)THEN
C
C**TASK REQUIRES A MANUAL RESPONSE TO A DIGIT
C
        NME=NME+1
        NIV(NME)=3
        T(NME)=(0.308+0.047*XTSK(2))/60.0
C
C**XE(1)=VIBRATION IN HZ
C**DATA FROM WOLDSTAT, BITTNER, AND BUIGNARD, 1982
C
        NME=NME+1
        NIV(NME)=1
        T(NME)=(1.800-0.167*XE(1)+0.004*XE(1)**2)/60.0
      ENDIF
    ENDIF
C
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 103
C
    NME=NME+1
    NIV(NME)=1
    T(NME)=(0.720-0.791*EXP(-0.238*XTSK(2)))/60.0
C
C**XO(2)=OPERATOR EXPERIENCE
C**XO(3)=SIGNAL PROBABILITY
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 87
C
    NME=NME+1
    NIV(NME)=2
    T(NME)=(0.492-0.00006*XO(2)-0.167*XO(3))/60.0
C
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 89
C
    NME=NME+1
    NIV(NME)=1
    T(NME)=(0.419-0.117*XO(3))/60.0
C
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 82
C
    NME=NME+1
    NIV(NME)=2
    X2=ALOG(XTSK(2))/0.693
    A=0.425*X2+0.295
    XK=-0.07*ALOG10(XTSK(2))-0.029
    T(NME)=(XK*ALOG10(XO(2))+A)/60.0
```

```
C
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 84
C
      NME=NME+1
      NIV(NME)=2
      A=0.190*X2+0.140
      XK=-0.017*X2-0.001
      T(NME)=(XK*ALOG10(XD(2)+A))/60.0
C
C**DATA FROM TEICHNER AND KREBS, 1974, PAGE 86
C
      NME=NME+1
      NIV(NME)=2
      T(NME)=(0.586+0.010*XTBK(2)-0.0000001*XD(2))/60.0
C
C**XD(4)=SIGNALS PER MINUTE
C**XTBK(6)=NUMBER OF DISPLAYS MONITORED
C**DATA FROM GOLDSTEIN AND DURTMAN, 1978, PAGE 606
C
      NME=NME+1
      NIV(NME)=2
      T(NME)=(-0.091+0.001*XD(4)+0.074*XTBK(6))/60.0
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR
C  FOR CHOICE, VISUAL, REACTION-TIME TASKS
C
      CALL ABSTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS A REDUNDANT, VISUAL AND AUDITORY REACTION-TIME TASK
C
    3 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR REDUNDANT VISUAL
C  AND AUDITORY, REACTION-TIME TASKS
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE RECOPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:HOPADS/VOLUME 9.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY-TO-RECOGNIZE MODERATORS
C          FOR AUDITORY AND/OR VISUAL RECOGNITION TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEE(1), NEO(10), NTSK(2), NIV(3), P(3), XE(1), XM(3)
      DIMENSION XO(10), XTSK(2)
C
C**DEFINE LENGTH OF ENVIRONMENTAL, OPERATOR, AND TASK STATE VECTORS
```

```
C
      DATA KE/1/
      DATA KO/10/
      DATA NT/2/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEE/4/
      DATA NEO/56,57,37,38,39,17,40,12,16,45/
      DATA NTSK/3,6/
C
C**DEFINE THE VECTOR ELEMENTS
C  XE(1)=AMBIENT NOISE LEVEL IN DB
C  XO(1)=TARGET NOISE LEVEL IN DB
C  XO(2)=TARGET DURATION IN MINUTES
C  XO(3)=TARGET HEIGHT IN FEET
C  XO(4)=TARGET WIDTH IN FEET
C  XO(5)=TARGET DEPTH IN FEET
C  XO(6)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C  XO(7)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C  XO(8)=TARGET TYPE
C  XO(9)=BINOCULAR USAGE
C  XO(10)=OBSERVER OFFSET IN NAUTICAL MILES
C  XTSK(1)=TASK MODALITY
C  XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C
C**RETRIEVE VALUES FOR OPERATOR AND TASK STATE VECTORS
C
      CALL GETEVA (IDE, N, NEE, KE, XE)
      CALL GETOVA (IDO, N, NEO, KO, XO)
C
C**NTSK CONTAINS ELEMENT NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT,ITASK, NTS, NTSK, NT, XTSK)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XTSK(1)=TASK MODALITY
C
      IF (XTSK(1).EQ.1.0)GO TO 1
      IF (XTSK(1).EQ.10.0)GO TO 2
      IF (XTSK(1).EQ.11.0)GO TO 19
C
C**THE TARGET IS AUDITORY
C**XE(1)=AMBIENT NOISE LEVEL IN DB
C**XO(1)=TARGET NOISE LEVEL IN DB
C**XO(2)=TARGET DURATION IN MINUTES
C**SN=SIGNAL-TO-NOISE RATIO
C**DATA FROM FLEISHMAN, 1975, PAGE 1143
C
    1 SN=XO(1)/XE(1)
      NME=NME+1
      NIV(NME)=3
      P(NME)=0.563+0.005*XO(2)-0.021*SN
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR
C  AUDITORY RECOGNITION
C
      CALL ABSPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
```

```
C**THE TARRET IS VISUAL
C**XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C**NPOS=THE OBSERVER-TO-TARGET POSITION AS AN INTEGER VARIABLE
C
    2 NPOS=NINT(XTSK(2))
      GO TO (3,4,5,17,18)NPOS
C
C**THIS IS A GROUND-TO-GROUND, VISUAL RECOGNITION TASK
C
    3 NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPHH TO DERIVE SKILL MODERATORS FOR
C  GROUND-TO-GROUND, VISUAL RECOGNITION
C
      CALL RELPHH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-GROUND, VISUAL RECOGNITION TASK
C**XO(3)=TARGET HEIGHT IN FEET
C**XO(4)=TARGET WIDTH IN FEET
C**XO(5)=TARGET DEPTH IN FEET
C**XO(6)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C**XFEET=SLANT RANGE TO TARGET IN FEET
C**XSYARD=THE AVERAGE OF THE TARGET'S THREE PLANAR AREAS IN SQUARE YARDS
C**XSMIL=ANGLE (IN SQUARE MILS) THAT THE TARGET SUBTENDS
C**DATA FROM FRANKLIN AND WHITTENBURG, 1965, PAGE 65
C
    4 XFEET=XO(6)*6076.1
      XSYARD=(((XO(3)*XO(4))+(XO(3)*XO(5))+(XO(4)*XO(5)))/3.0)/9.0
      XSMIL=XSYARD*((3000.0/XFEET)**2)
      NME=NME+1
      NIV(NME)=4
      P(NME)=0.91-0.264*EXP(-0.00002*XSMIL**2)
C
C**CALL ASSPHH TO DERIVE SKILL MODERATORS FOR
C  AIR-TO-GROUND, VISUAL RECOGNITION TASKS
C
      CALL ASSPHH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS A GROUND-TO-AIR, VISUAL RECOGNITION TASK
C**XO(7)=HORIZONTAL RANGE TO TARGET IN NAUTICAL MILES
C**XO(8)=TARGET TYPE
C**XKM=HORIZONTAL RANGE TO TARGET IN KILOMETERS
C**DATA FROM WRIGHT, 1966, PAGES 13, 16, AND 17
C
    5 XKM=XO(7)*1.852
      IF (XO(8).GT.10.0)GO TO 16
      NME=NME+1
      NIV(NME)=2
      NTARG=NINT(XO(8))
      GO TO (6,7,8,9,10,11,12,13,14,15)NTARG
C
C**TARGET IS AN F-4C
C
    6 P(NME)=0.043+1.092*EXP(-0.058*XKM**2)
      GO TO 16
C
C**TARGET IS AN F-100
C
    7 P(NME)=0.009+1.214*EXP(-0.073*XKM**2)
      GO TO 16
C
C**TARGET IS A F-33
```

```
C
      8 P(NME)=0.025+1.145*EXP(-0.055*XKM**2)
        GO TO 16
C
C**TARGET IS ANOTHER TYPE OF JET
C
      9 P(NME)=0.01+1.106*EXP(-0.052*XKM**2)
        GO TO 16
C
C**TARGET IS A U-1A
C
     10 P(NME)=0.05+0.994*EXP(-0.03*XKM**2)
        GO TO 16
C
C**TARGET IS A U-6A
C
     11 P(NME)=0.008+0.787*EXP(-0.062*XKM**2)
        GO TO 16
C
C**TARGET IS ANOTHER TYPE OF PROPELLER AIRCRAFT
C
     12 P(NME)=0.068+0.931*EXP(-0.068*XKM**2)
        GO TO 16
C
C**TARGET IS AN O1-A
C
     13 P(NME)=0.010+0.813*EXP(-0.074*XKM**2)
        GO TO 16
C
C**TARGET IS AN OH-23
C
     14 P(NME)=0.010+0.813*EXP(-0.074*XKM**2)
        GO TO 16
C
C**TARGET IS ANOTHER TYPE OF HELICOPTER
C
     15 P(NME)=0.009+0.928*EXP(-0.101*XKM**2)
C
C**XO(9)=BINOCULAR USAGE
C**DATA FROM WRIGHT, 1966, PAGE 14
C
     16 NME=NME+1
        NIV(NME)=2
        IF (XO(9).EQ.0.0)THEN
C
C**BINOCULARS WERE NOT WORN
C
           P(NME)=0.029+0.725*EXP(-0.079*XKM**2)
        ELSE
C
C**BINOCULARS WERE WORN
C
           P(NME)=0.042+1.03*EXP(-0.058*XKM**2)
        ENDIF
C
C**XO(10)=OBSERVER OFFSET IN NAUTICAL MILES
C**XM=OBSERVER OFFSET IN METERS
C**DATA FROM WRIGHT, 1966, PAGE 15
C
        NME=NME+1
        NIV(NME)=2
        XM=XO(10)*1852
        P(NME)=0.840+0.00005*XM-0.097*XKM
C
C**CALL ABSPMH TO DERIVE SKILL MODERATORS FOR
C  GROUND-TO-AIR, VISUAL RECOGNITION
```

```
C
      CALL ABSPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-AIR, VISUAL RECOGNITION TASK
C
   17 NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR
C  AIR-TO-AIR, VISUAL RECOGNITION TASKS
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS A VISUAL, DISPLAY-TARGET RECOGNITIION TASK
C
   18 NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR VISUAL,
C  DISPLAY-TARGET RECOGNITION
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      GO TO 999
C
C**THIS IS A REDUNDANT AUDITORY/VISUAL TARGET RECOGNITION TASK
C
   19 NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR
C  REDUNDANT AUDITORY/VISUAL TARGET RECOGNITION
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE RECOTH(IDO. IDE, N, DIST, ITASK. NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME-TO-RECOGNIZE MODERATORS
C          FOR AUDITORY AND/OR VISUAL RECOGNITION TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDS
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N). IDE(N). DIST(3). ITASK(NTS)
      DIMENSION NEO(4). NTSK(2). NIV(3). T(3). XM(3)
      DIMENSION XO(4). XTSK(2)
C
C**DEFINE LENGTH OF ENVIRONMENTAL. OPERATOR. AND TASK STATE VECTORS
```

```
C
      DATA KO/4/
      DATA NT/2/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEO/17,19,45,12/
      DATA NTSK/3,6/
C
C**DEFINE THE VECTOR ELEMENTS
C  XO(1)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C  XO(2)=TARGET BACKGROUND COMPLEXITY
C  XO(3)=AIRCRAFT SPEED IN KNOTS
C  XO(4)=TARGET TYPE
C  XTSK(1)=TASK MODALITY
C  XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C
C**RETRIEVE VALUES FOR OPERATOR AND TASK STATE VECTORS
C
      CALL GETOVA (IDO, N, NEO, KO, XO)
C
C**NTSK CONTAINS ELEMENT NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT,ITASK, NTS, NTSK, NT, XTSK)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XTSK(1)=TASK MODALITY
C
      IF (XTSK(1).EQ.1.0)GO TO 1
      IF (XTSK(1).EQ.10.0)GO TO 2
      IF (XTSK(1).EQ.11.0)GO TO 8
C
C**THE TARGET IS AUDITORY
C
    1 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
C  AUDITORY RECOGNITION
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THE TARGET IS VISUAL
C**XTSK(2)=THE OBSERVER-TO-TARGET POSITION
C**NPOS=THE OBSERVER-TO-TARGET POSITION AS AN INTEGER VARIABLE
C
    2 NPOS=NINT(XTSK(2))
      GO TO (3,4,5,6,7)NPOS
C
C**THIS IS A GROUND-TO-GROUND, VISUAL RECOGNITION TASK
C
    3 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
C  GROUND-TO-GROUND, VISUAL RECOGNITION
C
```

```
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-GROUND, VISUAL RECOGNITION TASK
C**XO(1)=SLANT RANGE TO TARGET IN NAUTICAL MILES
C**XFEET=SLANT RANGE TO TARGET IN FEET
C**XO(2)=TARGET BACKGROUND COMPLEXITY
C**DATA FROM BIEDEMAN, GOMER, AND LEVINE, 1980,
C  PAGES 104, 158
C
    4 XFEET=XO(1)*6076.1
      NME=NME+1
      NIV(NME)=2
      T(NME)=(-13.812+0.001*XFEET+1.132*XO(2))/60.0
C
C**XO(3)=AIRCRAFT SPEED IN KNOTS
C**XFTSEC=AIRCRAFT SPEED IN FEET/SECOND
C**DATA FROM BIEDEMAN, GOMER, AND LEVINE, 1980,
C  PAGES 104, 158
C
      XFTSEC=XO(3)*1.688
      NME=NME+1
      NIV(NME)=1
      T(NME)=(-6.348+0.001*XFEET-0.010*XFTSEC)/60.0
C
C**XO(4)=TARGET TYPE
C**DATA FROM BIEDEMAN, GOMER, AND LEVINE, 1980,
C  PAGES 104, 158
C
C**TARGET IS A TANK
C
      IF (XO(4).EQ.11.0)THEN
         NME=NME+1
         NIV(NME)=2
         T(NME)=(-8.990+0.001*XFEET)/60.0
      ENDIF
C
C**TARGET IS A HALF-TRACK
C
      IF (XO(4).EQ.17.0)THEN
         NME=NME+1
         NIV(NME)=2
         T(NME)=(-12.57+0.001*XFEET)/60.0
      ENDIF
C
C**TARGET IS A TRUCK
C
      IF (XO(4).EQ.15.0)THEN
         NME=NME+1
         NIV(NME)=2
         T(NME)=(-13.220+0.001*XFEET)/60.0
      ENDIF
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR
C  AIR-TO-GROUND, VISUAL RECOGNITION TASKS
C
      CALL ABSTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS A GROUND-TO-AIR, VISUAL RECOGNITION TASK
C
    5 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
```

```
C  GROUND-TO-AIR, VISUAL RECOGNITION
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS AN AIR-TO-AIR, VISUAL RECOGNITION TASK
C
    6 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
C  AIR-TO-AIR, VISUAL RECOGNITION TASKS
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS A VISUAL, DISPLAY-TARGET RECOGNITIION TASK
C
    7 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR VISUAL
C  DISPLAY-TARGET RECOGNITION
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      GO TO 999
C
C**THIS IS A REDUNDANT AUDITORY/VISUAL TARGET RECOGNITION TASK
C
    8 NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR
C  REDUNDANT AUDITORY/VISUAL TARGET RECOGNITION
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE STSYPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          SHORT-TERM MEMORY/SENSORY TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(1), NTSK(2), NIV(2), P(2), XM(3)
      DIMENSION XO(1), XTSK(2)
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
```

```
      DATA NEO/3/
      DATA NTSK/11.1/
      DATA NIV/2.1/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C     XO(1)=TIME ON TASK
C     XTSK(1)=NUMBER OF ITEMS KEPT IN SHORT TERM MEMORY
C     XTSK(2)=KCAL/MINUTE REQUIRED OF THE TASK
C
C**RETRIEVE THE VALUES FOR THE STATE VECTORS
C
      CALL GETOVA (IDO,N,NEO,KO,XO)
      CALL GETTSA(1,ITASK,NTS,NTSK,NT,XTSK)
C
C**INITIALIZE NME
C
      NME=0
C
C**EVALUATE XM
C**DATA FROM SIEGEL ET AL 1979
C
      NME=NME+1
      P(NME)=DIST(1)*((1.11-.2*XTSK(1))+(-.095+.22*XTSK(1))*
     +        EXP((-.044-.043*XTSK(1))*XO(1)))
C
C
C**DATA FROM DAVEY(1973)
C
      NME=NME+1
      IF (XTSK(2).LT.2.0) THEN
         P(NME)=DIST(1)*(1+.025*XTSK(2))
      ELSE
         P(NME)=DIST(1)*(1.05-.025*XTSK(2))
      ENDIF
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE STSYTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE SHORT-TERM MEMORY/SEMSORY
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    SHORT-TERM MEMORY/SCANNING
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMM TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMM(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE STSSPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:HOPADS/VOLUME 5.10 OF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          SHORT-TERM MEMORY/SYMBOLIC TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(1), NTSK(2), NIV(2), P(2), XM(3)
      DIMENSION XO(1), XTSK(2)
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
```

```
      DATA NEU/3/
      DATA NTSK/11,1/
      DATA NIV/2,1/
C
C**THE VECTOR ELEMENTS ARE AS FOLLOWS:
C     XI(1)=TIME ON TASK
C     XTSK(1)=NUMBER OF ITEMS KEPT IN SHORT TERM MEMORY
C     XTSK(2)=KCAL/MINUTE REQUIRED OF THE TASK
C
C**RETRIEVE THE VALUES FOR THE STATE VECTORS
C
      CALL GETOVA (IDC,N,NED,KD,XO)
      CALL RETTBA(1,ITASK,NTS,NTSK,NT,XTSK)
C
C**INITIALIZE NME
C
      NME=0
C
C**EVALUATE XM
C**DATA FROM SIEGEL ET AL 1979
C
      NME=NME+1
      P(NME)=DIST(1)*((1.11-.28*XTSK(1))+(-.095+.22*XTSK(1))*
     +        EXP(-.044-.043*XTSK(1))*XO(1)))
C
C
C**DATA FROM DAVEY(1973)
C
      NME=NME+1
      IF (XTSK(2).LT.2.0) THEN
         P(NME)=DIST(1)*(1+.025*XTSK(2))
      ELSE
         P(NME)=DIST(1)*(1.05-.025*XTSK(2))
      ENDIF
C
C**CALL RELPNH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPNH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE STRSTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C:*MODULE:HUM : FACTOR:S MODERATOR FUNCTIONS
C**REFERENCE:NAPADS/VOLUME 5.10 DF
C
C:*PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE SHORT-TERM MEMORY/SYMBOLIC
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(I)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    SHORT-TERM MEMORY/SYMBOLIC
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(I)
C
C**CALL RELTMM TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMM(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C
      SUBROUTINE TEAMPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          TEAM COORDINATION TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR TASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C**
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), P(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE
C**    TEAM COORDINATION TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1.
C
C**CALL RELPSH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPSH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C
      SUBROUTINE TEAMTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE TEAM COORDINATION
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE X4
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**    TEAM COORDINATION
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(1)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE THESPH(IDO. IDE. N. DIST. ITASK. NTS. XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY OF COMPLETING
C          TASKS WHICH REQUIRE THE ESTIMATION OF TIME
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NT3) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N). IDE(N). DIST(3). ITASK(NTS)
      DIMENSION NIV(1). P(1). XM(3)
C
C**INITIALIZE NME.NIV
C
      NME=0
```

```
      NIV(1)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT PROBABITY OF COMPLETING TASKS WHICH
C**    REQUIRE THE ERTIMATION OF TIME
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE TMERTH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE
C          TASKS WHICH REQUIRE THE ESTIMATION OF TIME
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       N1   NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NIV(1), T(1), XM(3)
C
C**INITIALIZE NME,NIV
C
      NME=1
```

```
      NIV(I)=0
C
C**EVALUATE XM
C**NO DATA WERE FOUND TO PREDICT TIME TO COMPLETE TASKS WHICH REQUIRE
C**THE ESTIMATION OF TIME
      NME=NME+1
      NIV(NME)=1
      T(NME)=DIST(I)
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO CUMPLETE
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE TIMEPH(IDO. IDE. N. DIST. ITASK. NTS. XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY TO COMPLETE
C          TIMESHARED TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N). IDE(N). DIST(3). ITASK(NTS)
      DIMENSION NIV(). P(1). XM(3)
C
C**INITIALIZE NME.NIV
C
      NME=0
```

```
      NIV(1)=0
C
C$$EVALUATE XM
C$$NO DATA WERE FOUND TO PREDICT PROBABILITY-TO-COMPLETE TIMESHARED TASKS
C
      NME=NME+1
      NIV(NME)=1
      P(NME)=DIST(1)
C
C$$CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY-TO-COMPLETE
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
      RETURN
      END
```

```
C:
      SUBROUTINE TIMETH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:HOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME TO COMPLETE FOR
C          TIMESHARED TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NUMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTSK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NTSK2(NT2) - WEIGHTS OF THE SKILL CATEGORIES USED IN THIS TASK
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME-TO-COMPLETE CALCULATED FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(1), NIV(1), NTSK(1), NTSK2(1), T(1), XM(3)
      DIMENSION XO(1), XTSK(1), XTSK2(1)
C
C**DEFINE LENGTH OF THE OPERATOR STATE VECTOR
```

```
C
      DATA KO/1/
      DATA NT/1/
      DATA NT2/1/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEO/3/
      DATA NTSK/3/
C
C**DEFINE NEEDED SKILL CATEGORY NUMBERS
C
      DATA NTSK2/11/
C
C**DEFINE THE VECTOR ELEMENTS
C  XO(1)=TIME ON TASK IN HOURS
C  XTSK(1)=TASK MODALITY
C  XTSK2(1)=DETECTION WEIGHTING
C
C**RETRIEVE VALUES FOR OPERATOR STATE VECTOR
C
      CALL GETOVA (IDO, N, NEO, KO, XO)
C
C**NTSK CONTAINS NUMBERS OF THE TASK SPECIFIC HUMAN FACTORS DATA
C
      IOPT=1
      CALL GETTSA (IOPT, ITASK, NTS, NTSK, NT, XTSK)
C
C**NTSK2 CONTAINS WEIGHTS OF THE SKILL CATEGORIES
C
      IOPT=2
      CALL GETTSA (IOPT, ITASK, NTS, NTSK2, NT2, XTSK2)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**PERFORMANCE DECREMENTS FOR TIMESHARED TASKS IS TASK DEPENDENT
C**XTSK(1)=TASK MODALITY
C**XTSK2(1)=DETECTION WEIGHTING
C
      IF (XTSK2(1).EQ.11.0.AND.XTSK(1).EQ.10.0)THEN
C
C**AT LEAST TWO VISUAL DETECTION TASKS ARE BEING TIMESHARED
C**XO(1)=TIME ON TASK IN HOURS
C**XMIN=TIME ON TASK IN MINUTES
C**DATA FROM LEVINE, ET AL., 1971, PAGE 19
C
        NME=NME+1
        NIV(NME)=1
        XMIN=XO(1)/60.0
        T(NME)=-0.058+0.011*XMIN-0.0001*XMIN**2
C
C**CALL ABSTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
        CALL ABSTMH(DIST,NIV,NME,T,XM)
      ELSE
C
C**THE TASKS DO NOT INVOLVE VISUAL DETECTION
C
        NME=NME+1
        NIV(NME)=1
        T(NME)=DIST(1)
C
```

```
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TIME TO COMPLETE
C
          CALL RELTMH(DIST,NIV,NME,T,XM)
      ENDIF
      RETURN
      END
```

```
C:
      SUBROUTINE TRACPH(IDO, IDE, N, DIST, ITASK, NTS, XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS PROBABILITY-TO-COMPLETE MODERATORS
C          FOR TRACKING TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C**
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NI - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NIASK(NI) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       P(I) - PROBABILITY TO COMPLETE CALCULATED FROM MODERATOR
C            EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N), IDE(N), DIST(3), ITASK(NTS)
      DIMENSION NEO(3), NIV(3), P(3), XM(3)
      DIMENSION XO(3)
C
C**DEFINE LENGTH OF THE OPERATOR STATE VECTOR
```

```
C
      DATA KO/3/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEO/24,60,3/
C
C**DEFINE THE VECTOR ELEMENTS
C  XO(1)=DAYS WITHOUT SLEEP
C  XO(2)=NUMBER OF REST PERIODS
C  XO(3)=TIME ON TASK IN HOURS
C
C**THE FOLLOWING IS A DUMMY STATEMENT FUNCTION FOR TIMEA
C  TIMEA IS A PRITSKER FUNCTION WHICH RETURNS CURRENT TIME
C  IN MILITARY CLOCK FORMAT
C
      TIMEA(X)=900.0*X
C
C**RETRIEVE VALUES FOR OPERATOR STATE VECTOR
C
      CALL SETOVA (IDO, N, NEO, KO, XO)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XO(1)=DAYS WITHOUT SLEEP
C**DATA FROM PFEIFFER, SIEGEL, TAYLOR, AND SHULER, 1979, PAGE 27
C
      NME=NME+1
      NIV(NME)=2
      RM=-69.835+57.666*XO(1)+0.010*TIMEA(1.0)
      P(NME)=RM*DIST(1)
C
C**XO(2)=NUMBER OF REST PERIODS
C**XO(3)=TIME ON TASK IN HOURS
C**DATA FROM SIEGEL, PFEIFFER, KOPSTEIN, WILSON, AND OZKAPTAN, 1979, PAGE 93
C
      NME=NME+1
      NIV(NME)=2
      RM=1.072-0.029*XO(2)+0.03**XO(3)
      P(NME)=RM*DIST(1)
C
C**DATA FROM SIEGEL, PLATZER, AND LANTERMAN, 1967, PAGE 13
C
      NME=NME+1
      NIV(NME)=1
      RM=1.477-6.683*EXP(-0.008*XO(3)**2)
      P(NME)=RM*DIST(1)
C
C**CALL RELPMH TO DERIVE SKILL MODERATORS FOR PROBABILITY TO BE ON TARGET
C
      CALL RELPMH(DIST,NIV,NME,P,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE TRACTH(IDO. IDE. N. DIST. ITASK. NTS. XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:HOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE RETURNS TIME DELAY MODERATORS
C          FOR TRACKING TASKS
C
C**INPUT PARAMETERS:
C       IDO(N) - ADDRESSING INFORMATION FOR THE OPERATOR
C       IDE(N) - ADDRESSING INFORMATION FOR THE ENVIRONMENT
C       ITASK(NTS) - ADDRESSING INFORMATION FOR THE TASK
C       N - INDEX FOR IDO AND IDE
C       NTS - INDEX FOR ITASK
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       KO - NUMBER OF OPERATOR INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       KE - NUMBER OF ENVIRONMENTAL INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NT - NUMBER OF TASK INDEPENDENT VARIABLES USED IN
C            THIS MODERATOR FUNCTION
C       NEO(KO) - ELEMENT NUMBERS OF VARIABLES IN THE OPERATOR
C            STATE VECTOR
C       NEE(KE) - ELEMENT NUMBERS OF VARIABLES IN THE ENVIRONMENT
C            STATE VECTOR
C       NTASK(NT) - ELEMENT NUMBERS OF VARIABLES IN THE TASK VECTOR
C       NME - NUMBER OF MODERATOR EQUATIONS
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C            EQUATION I
C       T(I) - TIME DELAY FROM MODERATOR EQUATION I
C
C**DIMENSION ARRAYS
C
      DIMENSION IDO(N). IDE(N). DIST(3). ITASK(NTS)
      DIMENSION NEO(1). NIV(1). T(1). XM(3)
      DIMENSION XO(1)
C
C**DEFINE LENGTH OF THE OPERATOR STATE VECTOR
C
```

```
      DATA KO/1/
C
C**DEFINE NEEDED ELEMENT NUMBERS
C
      DATA NEO/50/
C
C**DEFINE THE VECTOR ELEMENTS
C  XO(1)=OPERATOR'S EXPERIENCE
C
C**RETRIEVE VALUES FOR OPERATOR STATE VECTOR
C
      CALL GETOVA (IDO, N, NEO, KO, XO)
C
C**INITIALIZE NME,NIV
C
      NME=0
      NIV(1)=0
C
C**EVALUATE XM
C**XO(1)=OPERATOR'S EXPERIENCE
C**DATA FROM SHIPLEY, 1979, PAGE 25
C
      NME=NME+1
      NIV(NME)=1
      RM=(1.435-0.072*XO(1)+0.005*XO(1)**2)/1000.0
      T(NME)=RM*DIST(1
C
C**CALL RELTMH TO DERIVE SKILL MODERATORS FOR TRACKING TIME DELAY
C
      CALL RELTMH(DIST,NIV,NME,T,XM)
  999 RETURN
      END
```

```
C:
      SUBROUTINE ABSPMH(DIST,NIV,NME,P,XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE CALCULATES VALUES FOR THE SKILL
C          MODERATORS FROM ABSOLUTE PROBABILITIES
C
C**INPUT PARAMETERS:
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C                EQUATION I
C       NME - NUMBER OF MODERATOR EQUATIONS
C       P(I) - PROBABILITY CALCULATED FROM
C              MODERATOR EQUATION I
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       DP - DERIVED PROBABILITY
C       TNIV - TOTAL NUMBER OF INDEPENDENT VARIBLES
C
C**DIMENSION ARRAYS
C
      DIMENSION DIST(3),NIV(50),P(50),XM(3)
C
C**INITIALIZE DP AND TNIV TO ZERO
C
      DP=0.0
      TNIV=0.0
C
C**THE DERIVED PROBABILITY IS THE AVERAGE OF THE P(I)S
C  WEIGHTED BY THE NIV(I)S
C
      DO 1 I=1,NME
        DP=DP+(P(I)*NIV(I))
        TNIV=TNIV+NIV(I)
    1 CONTINUE
      DP=DP/TNIV
C
C**IF THE DERIVED PROBABILITY IS NEGATIVE,
C  MAKE THE FINAL PROBABILITY 0.0
C
      IF (DP.LT.0.0)THEN
```

```
         XM(1)=DIST(1)
      ELSE
C
C**IF THE DERIVED PROBABILITY EXCEEDS 1.0.
C  MAKE THE FINAL PROBABILITY 1.0
C
         IF (DP.GT.1.0)THEN
            XM(1)=1-DIST(1)
         ELSE
C
C**OTHERWISE, MAKE THE FINAL PROBABILITY
C  EQUAL THE DERIVED PROBABILITY
C
            XM(1)=DP-DIST(1)
         ENDIF
      ENDIF
C
C**ASSIGN VALUES FOR XM(2) AND XM(3)
C
    2 XM(2)=0.0
      XM(3)=DIST(3)
C
C**THE FOLLOWING PRINT STATEMENTS HAVE BEEN
C  INSERTED TO AID IN DEBUGGING THE MODERATOR
C  SUBROUTINES
C
      DO 1000 I=1,NME
      WRITE(6,1001)I,P(I),NIV(I)
 1001 FORMAT(/'THE PROBABILITY DERIVED FROM MODERATOR EQUATION ',
     +I3,' = ',F5.3/'.  THE NUMBER OF INDEPENDENT VARIABLES WAS ',
     +I3)
 1000 CONTINUE
      RETURN
      END
```

```
C:
      SUBROUTINE ABSTMH(DIST,NIV,NME,T,XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE CALCULATES VALUES FOR THE SKILL
C          MODERATORS FROM ABSOLUTE TIME-TO-COMPLETE
C
C**INPUT PARAMETERS:
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C                EQUATION I
C       NME - NUMBER OF MODERATOR EQUATIONS
C       T(I) - TIME-TO-COMPLETE CALCULATED FROM
C              MODERATOR EQUATION I
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C**
C**LOCAL VARIABLES:
C       DT - DERIVED TIME-TO-COMPLETE
C       TNIV - TOTAL NUMBER OF INDEPENDENT VARIBLES
C
C**DIMENSION ARRAYS
C
      DIMENSION DIST(3),NIV(50),T(50),XM(3)
C
C**INITIALIZE DT AND TNIV TO ZERO
C
      DT=0.0
      TNIV=0.0
C
C**THE DERIVED TIME-TO-COMPLETE IS THE AVERAGE OF THE T(I)S
C  WEIGHTED BY THE NIV(I)S
C
      DO 1 I=1,NME
         DT=DT+(T(I)*NIV(I))
         TNIV=TNIV+NIV(I)
    1 CONTINUE
      DT=DT/TNIV
C
C**IF THE DERIVED TIME-TO-COMPLETE IS NEGATIVE,
C  MAKE THE FINAL TIME-TO-COMPLETE 0.1
C
      IF (DT.LT.0.0)THEN
```

```
            XM(1)= DIST(1) + 0.1
         ELSE
C
            XM(1)=D1-DIST(1)
         ENDIF
C
C**ASSIGN VALUES FOR XM(2) AND XM(3)
C
      2  XM(2)=0.0
         XM(3)=DIST(3)
C
         RETURN
         END
```

```
      SUBROUTINE RELPMH(DIST,NIV,NME,P,XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE CALCULATES VALUES FOR THE SKILL
C          MODERATORS FROM ABSOLUTE PROBABILITIES
C
C**INPUT PARAMETERS:
C       DIST(3) - NOMINAL TASK PARAMETERS
C                     (1) MEAN
C                     (2) STANDARD DEVIATION
C                     (3) DISTRIBUTION TYPE
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C                EQUATION I
C       NME - NUMBER OF MODERATOR EQUATIONS
C       P(I) - PROBABILITY CALCULATED FROM
C              MODERATOR EQUATION I
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       DP - DERIVED PROBABILITY
C
C**DIMENSION ARRAYS
C
      DIMENSION DIST(3),NIV(50),P(50),XM(3)
C
C**INITIALIZE DP TO ZERO
C
      DP=0.0
C
C**THE DERIVED PROBABILITY IS A WEIGHTED AVERAGE OF
C  THE INDIVIDUAL P(I)S BASED UPON THE CONCEPT OF
C  DIMINISHING RETURNS
C
      DO 1 I=1,NME
         DP=DP + (P(I)-DIST(1))/I
    1 CONTINUE
      DP=DIST(1) + DP
C
C**IF THE DERIVED PROBABILITY IS NEGATIVE,
C  MAKE THE FINAL PROBABILITY 0.0
C
      IF (DP.LT.0.0)THEN
         XM(1)=-DIST(1)
      ELSE
C
```

```
C**IF THE DERIVED PROBABILITY EXCEEDS 1.0,
C  MAKE THE FINAL PROBABILITY 1.0
C
          IF (DP.GT..0)THEN
             XM(1)=1-DIST(1)
          ELSE
C
C**OTHERWISE, MAKE THE FINAL PROBABILITY
C  EQUAL THE DERIVED PROBABILITY
C
             XM(1)=DP-DIST(1)
          ENDIF
       ENDIF
C
C**ASSIGN VALUES FOR XM(2) AND XM(3)
C
      XM(2)=0.0
      XM(3)=DIST(3)
C
      RETURN
      END
```

```
      SUBROUTINE RELTMH(DIST,NIV,NME,T,XM)
C
C**MODULE:HUMAN FACTORS MODERATOR FUNCTIONS
C**REFERENCE:MOPADS/VOLUME 5.10 DF
C
C**PURPOSE:THIS SUBROUTINE CALCULATES VALUES FOR THE SKILL
C          MODERATORS FROM ABSOLUTE TIMES
C
C**INPUT PARAMETERS:
C       DIST(3) - NOMINAL TASK PARAMETERS
C                 (1) MEAN
C                 (2) STANDARD DEVIATION
C                 (3) DISTRIBUTION TYPE
C       NIV(I) - NUMBER OF INDEPENDENT VARIABLES IN MODERATOR
C                EQUATION I
C       NME - NUMBER OF MODERATOR EQUATIONS
C       T(I) - PROBABILITY CALCULATED FROM
C              MODERATOR EQUATION I
C
C**OUTPUT PARAMETERS:
C       XM(I) - SKILL CATEGORY MODERATOR VALUES
C               XM(1) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       MEAN
C               XM(2) = DIFFERENCE BETWEEN DERIVED AND BASELINE
C                       STANDARD DEVIATION
C               XM(3) = DERIVED DISTRIBUTION TYPE
C
C:
C**LOCAL VARIABLES:
C       DT - DERIVED TIME
C
C**DIMENSION ARRAYS
C
      DIMENSION DIST(3),NIV(50),T(50),XM(3)
C
C**INITIALIZE DT TO ZERO
C
      DT=0.0
C
C**THE DERIVED TIME IS A WEIGHTED AVERAGE OF
C  THE INDIVIDUAL T(I)S BASED UPON THE CONCEPT OF
C  DIMINISHING RETURNS
C
      DO 1 I=1,NME
         DT=DT + (T(I)-DIST(1))/I
    1 CONTINUE
      DT=DIST(1) + DT
C
C**IF THE DERIVED TIME IS NEGATIVE,
C  MAKE THE FINAL TIME 0.1
C
      IF (DT.LT.0.0)THEN
         XM(1)=-DIST(1)  + 0.1
      ELSE
C
```

```
C
C**OTHERWISE, MAKE THE FINAL TIME
C  EQUAL THE DERIVED TIME
C
          XM(1)=DT-DIST(1)
        ENDIF
C
C**ASSIGN VALUES FOR XM(2) AND XM(3)
C
    2 XM(2)=0.0
      XM(3)=DIST(3)
C
      RETURN
      END
```

VI. USER INSTRUCTIONS

This section contains instructions on writing the utility subprograms GETOVA, GETEVA, GETTSA, and TIMEA. In addition, it also explains how to call the moderator functions.

SUBROUTINE GETOVA (IDO, N, NEO, KO, XO)

IDO(N) - An N-dimensional vector of address information for the operator state vector. IDO has meaning with respect to the data structure created by the module to store the operator state vector. For example, if the operators are numbered 1,2,3,...; then IDO(1) could equal the operator number. In MOPADS, IDO defines the data base address of the operator.

NEO(KO) -NEO specifies the indexes of elements of the operator state vector being requested. For example, if KO=2, NEO(1)=6, and NEO(2)=12, then elements 6 and 12 of the operator state vector are requested. See Table VI-1 for the definition of these elements.

XO(KO) - GETOVA returns the values of the elements specified in NEO in this vector. For the example above, if element #6 is 24.22 and element #12 is 1.6, then XO(1) would equal 24.22 and XO(2) would equal 1.6.

SUBROUTINE GETEVA (IDE, N, NEE, KE, XE)

IDE(N) - An N-dimensional vector of address information for the environmental state vector. It is analogous to IDO in GETEVA.

Table VI-1

OPERATOR STATE VECTOR VARIABLES

1. Core temperature (degrees C)
2. Clo value of operator's clothes
3. Time spent on task today (hours)
4. Consecutive days of duty
5. Number of dimensions on which objects can be discriminated
6. Number of fire units for which the operator is responsible
7. Percentage recovery since last work period
8. Hours worked in the previous work session
9. Hours between previous and current work session
10. Flash intensity in foot-lambert seconds
11. Target speed in knots
12. Target type (see last page of table for definitions)
13. Target size (determined by target type)
14. Target color (white=1, tan=2, green=3, blue=4, red=5, yellow=6)
15. Search area (degrees)
16. Use of binoculars (1=yes, 0=no)
17. Slant range to target in n. mi.
18. Target Trajectory (degrees relative to north)
19. Target background complexity (low=1, medium=2, high=3)
20. Number of display background characters
21. Number of messages requiring operator attention
22. Signals per minute
23. Hours worked per week
24. Days without sleep
25. Consecutive days of night duty
26. Number of tasks performed at once
27. Target/background contrast ratio
28. Average hours sleep received per night over 3 day period
29. Objective function (not used by moderators)
30. Number of goals considered (not used by moderators)
31. Target brightness in foot lamberts
32. Number of consecutive nights with known hours sleep

Table VI-1 (continued)

33. Sky/ground ratio in foot lamberts
34. Aircraft altitude in feet
35. Meteorological range (visibility) in n. mi.
36. Threshold of operator's object/background contrast ratio
37. Target height in feet
38. Target width in feet
39. Target depth in feet
40. Horizontal range to target (n. miles)
41. Number of resolution elements to resolve the target
42. Number of areas with suspected targets
43. Target speed (knots)
44. Field of view operator is searching (degrees)
45. Observer offset (nautical miles)
46. Unused
47. Display target location (defined by a 3x3 matrix numbered from left to right, then down)
48. Target displacement from observer in degrees (left=negative, right=positive)
49. Display resolution in number of lines
50. Average height of nontargets in view in feet
51. Average width of nontargets in view in feet
52. Average depth of nontargets in view in feet
53. Distance to display in feet
54. Display height in feet
55. Display width in feet
56. Target noise level in dB
57. Target duration in minutes
58. Number of times the operator has performed this task before)
59. Signal probability
60. Number of rest periods per work shift
61. Task error factor (not used in moderator subroutines)
62. Task element error factor (not used in moderator subroutines)
63. Days since the task was last practiced
64. Operator's sense of direction (good=1, poor=0)
65. Skin temperature (degrees centigrade)
66. Minutes in current ambient temperature
67. Previous skin temperature prior to changing environments

Table VI-1 (continued)
TARGET TYPE CODES

CODE NO.	NAME	COUNTRY	MISSION
1	F4C	USA	
2	F100	USA	
3	T33	USA	
4	OTHER JET		
5	U1A	USA	
6	U6A	USA	
7	OTHER PROP		
8	01A	USA	
9	OH23	USA	
10	OTHER HELICOPTER		
11	TANK		
12	JEEP		
13	TROOP		
14	APC		
15	TRUCK		
16	ZERO		
17	HALFTRACK		
18	F14	USA	
19	F15	USA	
20	F16	USA	
21	F18	USA	
22	MIG21	USSR	
23	MIG23	USSR	
24	MIG25	USSR	
25	SOLDIER(FOOT)	ANY	
26	MIG-27	USSR	
27	SU-17	USSR	
28	QIANG Ji-5	PRC	Ground Attack
29	R-2358	FRANCE	Military surveillance
30	MIRAGE 3E	FRANCE	Fighter
31	MIRAGE F1	FRANCE	Fighter
32	MIRAGE 2000	FRANCE	Fighter
33	MIRAGE 4000	FRANCE	Fighter
34	MIRAGE 4	FRANCE	Bomber
35	MIRAGE 5	FRANCE	Ground Support
36	MIRAGE 50	FRANCE	Fighter
37	AW.660	GREAT BRITAIN	Military Transport
38	698	GREAT BRITAIN	Bomber
39	HS748	GREAT BRITAIN	Military Transport
40	HS.780	GREAT BRITAIN	Military Transport
41	P.1099	GREAT BRITAIN	Ground Attack

Table VI-1 (continued)

42	IAI202	ISRAEL	Military Transport
43	MIRAGE 3C	ISRAEL	Fighter
44	KfirC2	ISRAEL	Fighter
45	G.222	ITALY	Military Transport
46	F1045	ITALY	Interceptor
47	MB.326K	ITALY	Strike
48	S.M.1019E	ITALY	Military STOL
49	F-1	JAPAN	Fighter
50	C.207A	SPAIN	Military Transport
51	SF-5A	SPAIN	Fighter
52	HA-220	SPAIN	Ground Attack
53	35XD	SWEDEN	Ground Attack
54	JA37	SWEDEN	Fighter
55	J-1	YUGOSLAVIA	Strike
56	Light (e.g.blinking)		
57	Digit (digit on a display)		
58	MIG-17	USSR	
59	MIG-19	USSR	
60	SU-7	USSR	
61	SU-9PM	USSR	
62	SU-11	USSR	
63	SU-15	USSR	
64	SU-19	USSR	
65	SU-20	USSR	
66	YAK-28P	USSR	
67	YAK-36	USSR	
68	TU-28?	USSR	
69	IL-28	USSR	
70	M-4	USSR	
71	TU-16	USSR	
72	TU-20	USSR	
73	TU-22	USSR	
74	TU-26	USSR	
75	IL-38	USSR	
76	TU-126	USSR	
77	MIG-15	USSR	
78	L-29	USSR	
79	L-39	USSR	
80	J-5	USSR	
81	J-6	USSR	
82	J-7	USSR	
83	J-8	USSR	
84	H-5	USSR	
85	H-6	USSR	
86	CJ-6	USSR	
87	Y-11	USSR	
88	MIRAGE III	FRANCE	

NEE(KE) - Element numbers of the environmental state vector. This is analogous to NEO in GETOVA. For a list of the environmental state vector, see Table VI-2.

XE(KE) - GETEVA returns the values specified by NEE in XE. XE is analogous to XO in GETOVA.

SUBROUTINE GETTSA (IOPT, ITASK, NTS, NTSK, NT, XTSK)

IOPT - Option
- 0 - Requesting task time information
- 1 - Requesting task vector information
- 2 - Requesting skills information
- 3 - Requesting system resource information

ITASK - An NTS dimensional vector of address information for the task data. It is analogous to IDO in GETOVA.

NTSK(NT) - The desired element numbers. There are four cases:
- IOPT=0 - NTSK is not used
- IOPT=1 - NTSK contains element numbers of the task specific categories
- IOPT=2 - NTSK contains skill numbers
- IOPT=3 - NTSK contains resource numbers

XTSK(NT) - The values of variables requested in NTSK above. There are four cases:
- IOPT=0 - XTSK(1) to XTSK(3) are distribution type, mean, and standard deviation.
- IOPT=1 - The values of the task vector elements requested in NTSK.
- IOPT=2 - The weights of the skill categories specified by the location in the vector (e.g., XTSK(5) would be the weight for skill number 5.
- IOPT=3 - The values of resource parameters specified in NTSK.

Table VI-2

ENVIRONMENTAL STATE VECTOR VARIABLES

1. Dry bulb temperature in degrees centigrade
2. Relative humidity
3. Air movement rate (miles/hour)
4. Noise level in dB
5. Work area illumination in foot-lamberts
6. Number of operators on duty
7. Vibration as defined by vertical movement in Hz
8. Ambient vapor pressure in mb
9. Noise predictability (consistent=0, inconsistent=1)

Table VI-3 contains definitions of the data in the task vector. Resource requirements are not used in the current MOPADS implementation and, therefore, are not utilized by the moderator functions. However, we felt that this option was important for growth.

FUNCTION TIMEA()

TIMEA returns the current military time accurate to the nearest minute. TIMEA ranges in value from 0000 to 2359. TIMEA is a REAL function.

All of the moderator functions have the following calling sequence:

```
SUBROUTINE xxxxxxH (IDO, IDE, N, DIST, ITASK,
                    NTS, XM)
```

The parameters in this calling sequence are defined as follows:

IDO(N), IDE(N) - Addressing information for the operator and environmental state vectors, respectively. These vectors are then used to call GETOVA and GETEVA.

DIST(3) - Nominal (i.e., baseline) task parameters.
- DIST(1)=mean
- DIST(2)=standard deviation
- DIST(3)=distribution type (see Table VI-4)

ITASK(NTS) - Addressing information for the task. ITASK is used for calling GETTSA.

Table VI-3

TASK VECTOR VARIABLES

1. Kcal/minute required by the task
2. Number of tasks which might be performed after this one
3. Primary stimulus mode
4. Secondary stimulus mode
5. Response mode
6. Observer/target position (1= ground to ground,
   2= air to ground,
   3= ground to air,
   4= air to air,
   5= at a display
7. Distance to control in feet
8. Control width in feet
9. Number of displays that must be monitored
10. Number of controls that might be actuated
11. Number of items which must be kept in short term memory

Table VI-3 (continued)

TASK SPECIFIC DATA

Column	Description	Default
7	Kilocalories/minute	1
8	Number of branches out	1
9	Stimulus Mode 1 A two-digit number 10's digit - 0 - no visual 1 - visual 1's digit - 0 - no auditory 1 - auditory	10
10	Stimulus Mode 2 A three-digit number as above 100's digit- 0 - no olfactory 1 - olfactory 10's digit - 0 - no kinesthetic 1 - kinesthetic 1's digit - 0 - no tactile 1 - tactile	00
11	Response Mode A two-digit number 10's digit - 0 - no vocal 1 - vocal 1's digit - 0 - no tactile 1 - tactile	1
12	Observer - Target Position 1 - ground to ground 2 - air to ground 3 - ground to air 4 - air to air 5 - at a display	3
13	Control Distance (distance from operator to control device in feet)	1
14	Control Width (width of the control device in feet)	0.1
15	Number of Displays	1
16	Number of Alternatives (number of possible controls that may need to be actuated	1

Table VI-4

DISTRIBUTION TYPES

1	CONSTANT
2	NORMAL
3	UNIFORM
4	ERLANG-1 (exponential)
5	LOGNORMAL
6	NOT USED
7	BETA
8	GAMMA

XM(3) - Moderated output
XM(1)=Change to the nominal mean. In other words, the moderated mean is DIST(1)+XM(1)
XM(2)=Change to the nominal standard deviation
XM(3)=New distribution type

As mentioned previously, in the current MOPADS implementation, XM(2)=0 and XM(3)=DIST(3). Only the mean is currently moderated.

VII. ERROR PROCESSING

No error processing occurs in the moderator subroutines.

VIII. COMMON VARIABLE DEFINITIONS

No common variables are used by these subroutines. All variables are passed as subroutine parameters or are local variables.

IX. REFERENCES

Laughery, K. R. A data base for quantitative human performance modeling(MOPADS Vol. 5.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981.

Laughery, K. R. Moderating task performance in MOPADS (MOPADS Vol. 5.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Laughery, K. R. & Ditzian, J. L. The underlying person model behind HOMO (human operator model) (MOPADS Vol. 5.5). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1981.

Laughery, K. R. & Ditzian, J. L. MOPADS literature review (MOPADS Vol. 5.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1982.

Polito, J. MOPADS data base application programs (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft.Bliss, TX, 1983.

X. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc. (5)
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.81, Ft.Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245,The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Rm. 108
Boulder, CO 80301

XI. CHANGE NOTICES

APPENDIX S

MOPADS FINAL REPORT:

MOPADS FREE-FORMAT SYNTAX PROCESSOR (MOPADS/FFSP)

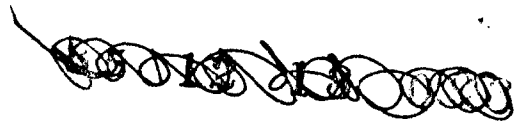


TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

TABLE OF CONTENTS
(Continued)

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS AND TERMINOLOGY

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

2-0 Other Terminology

COMMAND	A keyword entered by the user that specifies some action to be taken. Commands may be abbreviated to the letters that will make them unique from all the other commands.
COMMAND MODES	The manner in which a command may be entered: either regular, concise, or command-name-only. In regular mode the user will enter the command followed by a list of prompts and their responses. In concise mode the user enters the command followed by a list of responses. The command-name-only mode is the simplest to use since the user only enters the command and is then prompted for all the remaining information.
COMMAND SUBROUTINE	The subroutine that performs the logic of the command and processes responses to the command's prompts.
DEFAULT VALUES	The value that will be used for a response. A value that is used when the user elects not to explicitly enter a value for a response.
ENUMERATED VALUES	A discrete set of allowable responses for a prompt.
LOWER BOUND/ UPPER BOUND	The minimum/maximum value for a response.
MIXED RESPONSE	A response made up of more than one type (integer, real, or character).
PROMPT	A keyword requiring a value or string as a response.
PROMPT-RESPONSE-GROUPS	The sets of prompts followed by their responses.

RESPONSE	The value entered for a prompt.
RESPONSE GROUPS	The set of responses for a prompt.
RESPONSE TYPE	The type of a response: either a real, integer, character or mixed.
RESPONSE UNIVERSE TYPE	Specifies the characteristics of the values for responses. The response universe may be a range of values or an enumerated set.
UPDATED DEFAULTS	Default values that are overwritten when the user explicitly enters a new value. The new value will become the default value.

I. PURPOSE

1-0 GENERAL DESCRIPTION OF FREE-FORMAT SYNTAX PROCESSOR (FFSP) ROUTINES

This FORTRAN 77 program module is used to perform interactive free-form syntax processing of MOPADS user commands. This module checks each field in the command for both arithmetic and logical validity. The responses are returned from the FFSP programs in three response output arrays (one for character responses, one for integer responses, and one for real responses). If an error occurs in the processing of a command and/or its prompts and responses, one of three actions may occur. The user may be asked to re-enter the information, the command may be cancelled and control returned to the point where the FFSP routines were called, or in extreme cases execution will terminate.

The FFSP module is a generalized processor of the syntax described below. It is used by the MOPADS user interface to process MOPADS user commands. Since it is a general processor, however, it can be used to write user interfaces in non-MOPADS contexts.

2-0 COMMAND MODES

The commands may be entered in any one of three forms:

1. command-name-only
2. regular mode (default mode), and
3. concise mode.

The three forms reflect the relative experience the user may have with a particular command. The user that has very little experience with a command may enter that command in the command-name-only form. After gaining some experience with a command and its prompts the user will begin to enter the command in regular mode. When the user becomes proficient with a particular command he/she will begin using the command in concise mode. A description of the three command forms follows.

2-1. Command-Name-Only.

This form is illustrated in Figure I-1 where upper case letters are typed by the computer and lower case letter are typed by the user. The user simply enters the command and the system will prompt the user for any additional information needed (HAIR[BLONDE]= is a prompt).

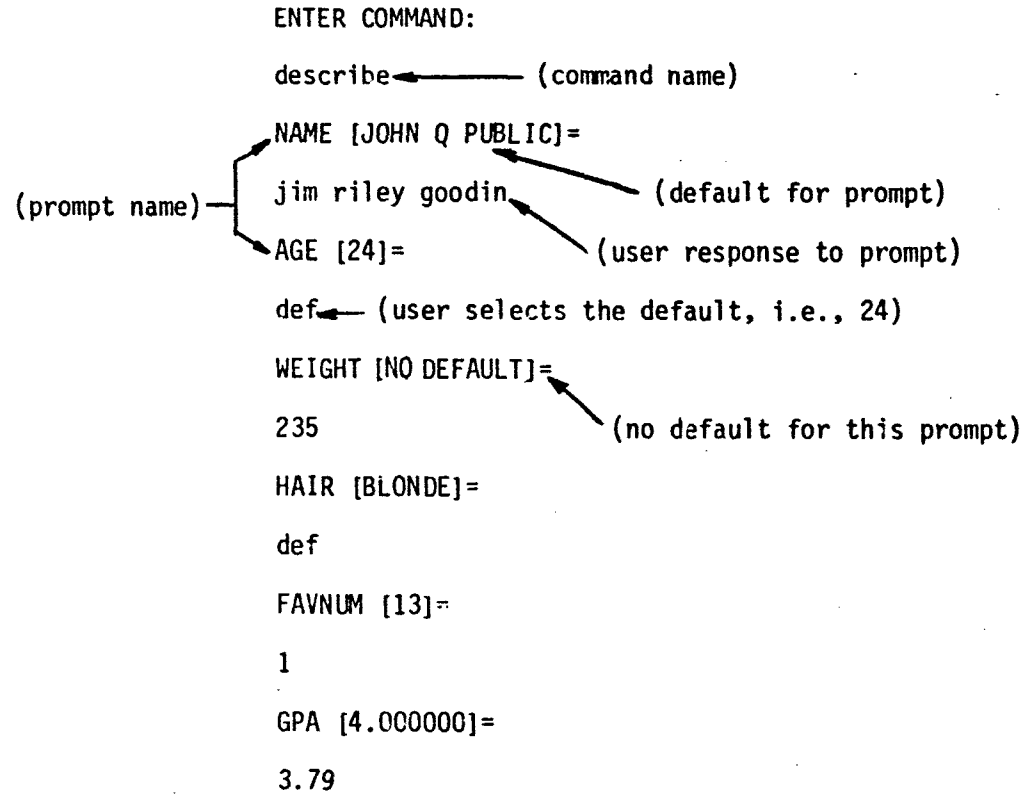


Figure I-1. Example of Command In The Command-Name-Only Form.

The information printed between the brackets "[.....]" following a prompt represents the current default values for that prompt. If the user enters "DEF" as his response, the value(s) in the brackets will become the response to the prompt. If the text "NO DEFAULT" appears between the brackets, the user must enter a response for this prompt other than the "DEF" response.

2-2. Regular Mode.

Figure I-2 illustrates the use of this form. Once again the upper case letters represent information typed by the computer and lower case letters represent user supplied information. In each of the examples in Figure I-2 the user enters the command followed by a complete or partial list of prompts and responses.

In the first example, the user entered the command and the entire list of prompts and responses. The prompt-response groups may be entered in any order and are separated from one another with a slash (/). The next two examples illustrate the case where the user enters the command followed by a partial list of prompts and responses. In the second example in Figure I-2 the last non-blank character entered by the user is a dash (-). The dash instructs the system to prompt for all unentered responses. In the third example, no dash is typed at the end of the command line. This instructs the system to prompt only for responses that have no defaults and to accept the default values for all other responses.

2-3. Concise Mode.

This form is illustrated in Figure I-3. The user enters the command followed by a partial or complete set of responses. The response list must be entered in a preset order which is the same order the prompts appear at the terminal when using the command-name-only form. The user enters only the responses, with the different response groups separated by slashes.

The first example in Figure I-3 illustrates the case where the user entered the command followed by a complete list of responses. In this example, regular mode is the default mode; therefore, the user must precede any commands entered in concise mode with a "C-". This instructs the system that the following command and responses are in concise mode. The dash at the end of the second example has the same effect as in regular mode; the system will prompt for any unentered responses. If the user leaves off the dash, as in the last example, the system will use the default value(s) for any unentered responses with default value(s).

```
EXAMPLE 1

        ENTER COMMAND:      describe/hair=black/name=john,h,doe/age=37/weight=185/favnum=2/gpa=3.78

EXAMPLE 2

        ENTER COMMAND:      describe/name=john h doe/favnum=def/weight=185/-
                            AGE [24]=
                            37
                            HAIR [BLONDE]=
                            black
                            GPA [4.000000]=
                            3.79

EXAMPLE 3

        ENTER COMMAND:      describe/weight=185/name=john h doe/
```


Figure I-2. Examples of Commands In The Regular Mode Form.

```
EXAMPLE 1

        c-describe/john h doe/37/231/black/def/3.18

EXAMPLE 2

        describe/john,h,doe/37/231/-
        HAIR [BLONDE] =
        black
        FAVNUM [13]=
        16
        GPA  [4.000000]=
        3.18

EXAMPLE 3

        describe/john h doe/37/231
```


Figure I-3. Examples of Commands In The Concise Mode Form.

3-0 FFSP SYNTAX RULES

The following rules will formalize the previous discussion of how syntax is processed by FFSP.

3-1. The command-name-only form of a command may be used in regular or concise modes by typing only the command name.

3-2. Blank responses may be entered where character responses are required. To enter a blank response type " " (including the quotes). At least one blank must be entered between the quote marks.

3-3. A command may be cancelled at any time by typing CANC for any prompt or response. You can not use CANC as an abbreviation to a prompt.

3-4. The user may elect to use the default value(s) by typing DEF for any response in a response list up to one field past the last response in the list.

3-5. Slashes (/) must be used to separate one prompt-response group from another. Blanks or commas may be used to separate all other fields. The equal sign should be used to separate prompts from their responses; however, it is not required.

3-6. Command and prompt names may be abbreviated to any non-ambiguous string of characters. For example, if there were two commands, DESIGN and DESCRIBE, they can be abbreviated DESI and DESC respectively. The commands may be abbreviated in longer forms. For example, the user may enter DESC, DESCR, DESCRI, DESCRIB, or DESCRIBE for the command DESCRIBE.

3-7. If a command line in regular or concise mode is ended with more than one dash, the last dash will signify to the system to prompt the user for all the unentered responses. Other dashes will then be considered as part of a response.

3-8. Any multiple combination of commas and blanks is treated as a single separator. For example,

NAME = BILL WOLF and NAME = BILL , WOLF

are equivalent.

3-9. If the user enters an incorrect response or misuses the syntax, FFSP will explain the error and prompt interactively for all remaining responses.

3-10. It is possible to specify the default mode to be regular or concise. This defualt mode can be overridden for individual commands by preceding the command name with "R-" (for regular mode) or "C-"(for concise mode).

II. DATA STRUCTURE

1-0 INTRODUCTION

The data structure used by the FFSP programs may be considered in two parts, 1) input data structure and 2) output data structure. The input data is passed to the FFSP subroutines through the subroutine parameters and contains the information describing the prompts and responses. The output data is generated by the FFSP programs. It contains the user's responses to the prompts.

2-0 INPUT DATA STRUCTURE

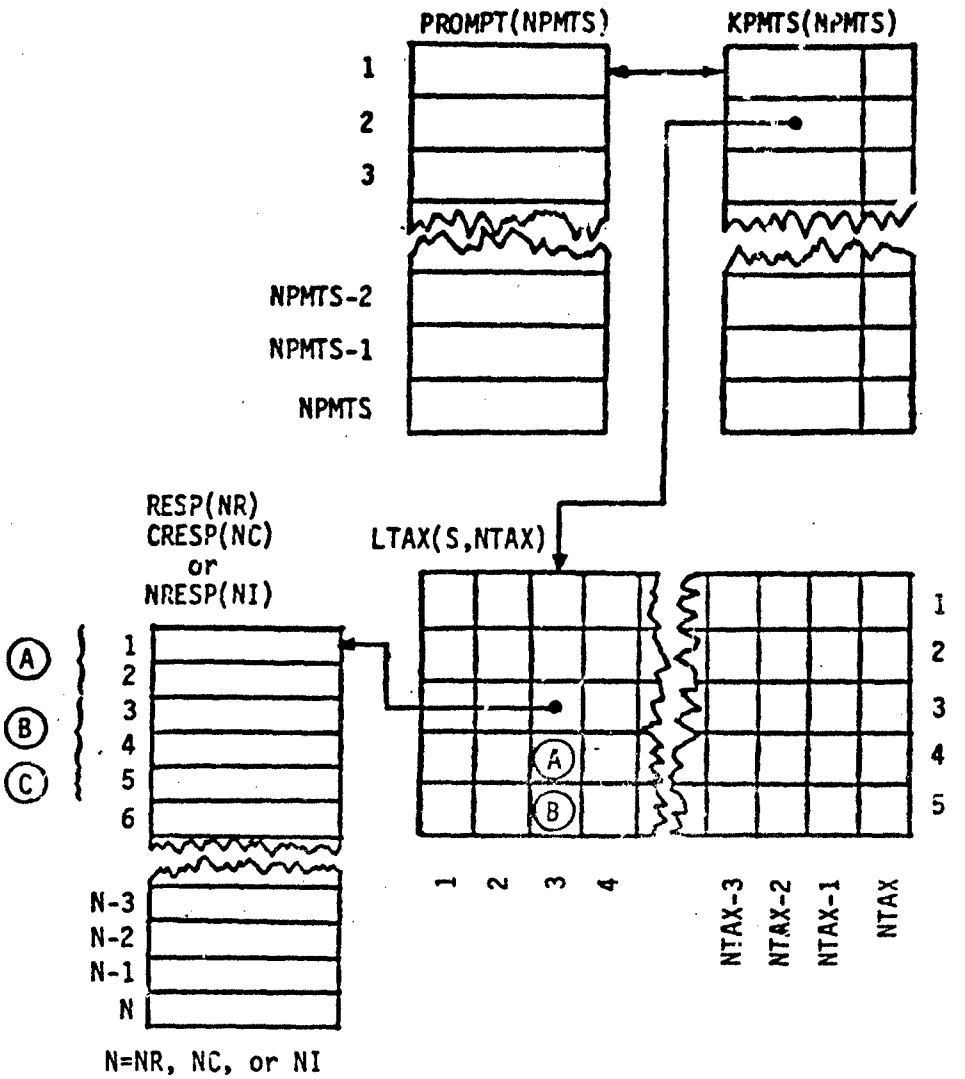
The input data structure is contained in arrays passed to FFSP as formal subprogram parameters. The data arrays and their relationships to one another are illustrated in Figure II-1. The following subsections discuss their form and contents.

2-1. PROMPT(NPMTS) is a CHARACTER array used for storing prompt names for a command. The dimension of this array, NPMTS, is passed to the FFSP programs as a formal parameter. The order that the prompt names are stored in PROMPT will be the order the system will print them and accept their responses while in the command-name-only form.

2-2. KPMTS(NPMTS,2) is an INTEGER array containing two columns. There is a correspondence between the rows of KPMTS and the elements of the PROMPT array. The information in row 3 of the KPMTS array pertains to the third prompt in PROMPT. Column 1 of any row contains a pointer to a column of the LTAX array. Column 2 is working storage used internally by the FFSP programs to dynamically keep track of which responses have been entered.

2-3. LTAX(5,NTAX) is an INTEGER array with information pertaining to the prompts in the PROMPT array. The information describes the expected responses and points to information stored in the response universe arrays (RESP, NRESP, CRESP). The contents of a column of the LTAX array are as follows:

Row 1	Response type
Row 2	Response universe type
Row 3	Pointer to the response universe array of the type indicated by the value in Row 1 (RESP,NRESP, or CRESP).
Row 4	The number of default values stored in the response universe array.
Row 5	Number of enumerated values stored in response universe array.



(A) - Number of default values

(B) - Two elements of response universe array used for storing upper and lower bound values

(C) - Number of enumerated values

Figure II-1. Input Data Structure.

Row 1 indicates response type using the values shown in Table II-1. The length of the list, n, is also contained in the response type value. For example, the response type value for a list of six real values would be 2006. If the value of n is 0, the list is of unspecified length. The mixed response type (values >4000 and <5000) permits the command designer to allow for responses of more than one type. For example, say the command designer wants a response to consist of two character strings followed by an integer and another character string. The value for this response type would be 4003. The "3" indicates that the mixed response consists of three response type values. The next three columns of the LTAX array would contain the information concerning the response lists that make up the mixed response (the values 3002, 1001, and 3001 would be in row 1 of these columns). The mixed response type can not have an undefined length; therefore, a value of 4000 will always be invalid. In addition to this, a mixed response list can not contain any response lists of undefined length (no 1000, 2000, or 3000 responses type values in a mixed response list). Examples of a mixed response prompts are given in Section VI.

Row 2 of the LTAX array is a value from Table II-2. This code value represents the type of response universe the response must be from. Values >100 signify responses with defaults that are updated every time a new entry is made (the new response becomes the current default value(s)). It is invalid to use response universe type values >100 with response lists of unspecified length.

Row 3 of the LTAX array has the pointer to the response universe array, RESP, NRESP, or CRESP depending on the response type specified in row 1. For example, if the response type value is 2005, the pointer would point to the location in the RESP array where the response universe information begins. If the response type value is >4000, the value in row 3 will be zero since each individual response list in the mixed response will have its own response universe.

Row 4's value indicates how many default values are stored in the response universe array for a response. For defined length lists, this value must equal the last digit in the response type value or be zero (no default value(s) for the response). For example, if the response type value is 2003, the value in row 4 must be 3 or 0. If the response universe was specified to be >100, the value in row 4 must not be 0 (storage must be available to save updated default values).

Row 5's value indicates how many enumerated values (valid values for a response) are stored in the response universe array for a response. This value must be 0 unless the response universe type value was 4 or 104. If a response is from a set of enumerated values, then the default values, if there are any, must also be from the enumerated set.

Table II-1

Response Type Values

Value	Response Must Be a List of n
1000 + n	integers
2000 + n	reals
3000 + n	character strings
4000 + n	mixed types

NOTE: If n = 0 the response list is of unspecified length. n must not be zero if the response type value is is 4000 + n.

Table II-2

Response Universe Type Values

Value	Response Universe Type
1 or 101	response has an upper and a lower bound
2 or 102	response has an upper bound only
3 or 103	response has a lower bound only
4 or 104	response must be equal to an enumerated value
5 or 105	no limitations on the value of the response

2-4. RESP(NR), NRESP(NI), CRESP(NC) are the RESPONSE UNIVERSE arrays used to store default values, enumerated values, and range boundaries for the responses. RESP is used to store real number response universe values. NRESP is used to store integer number response universe values, and CRESP is used to store character string response universe values.

All the values stored in a response universe array for a particular response are contained in a contiguous block of elements. Two typical blocks of elements are shown in Figure II-2. The minimum number of elements in a block is two, one for each of the range boundaries whether they exist or not.

The dimension of the response universe arrays (NR, NI, NC) are passed as formal subroutine parameters to the FFSP subprograms.

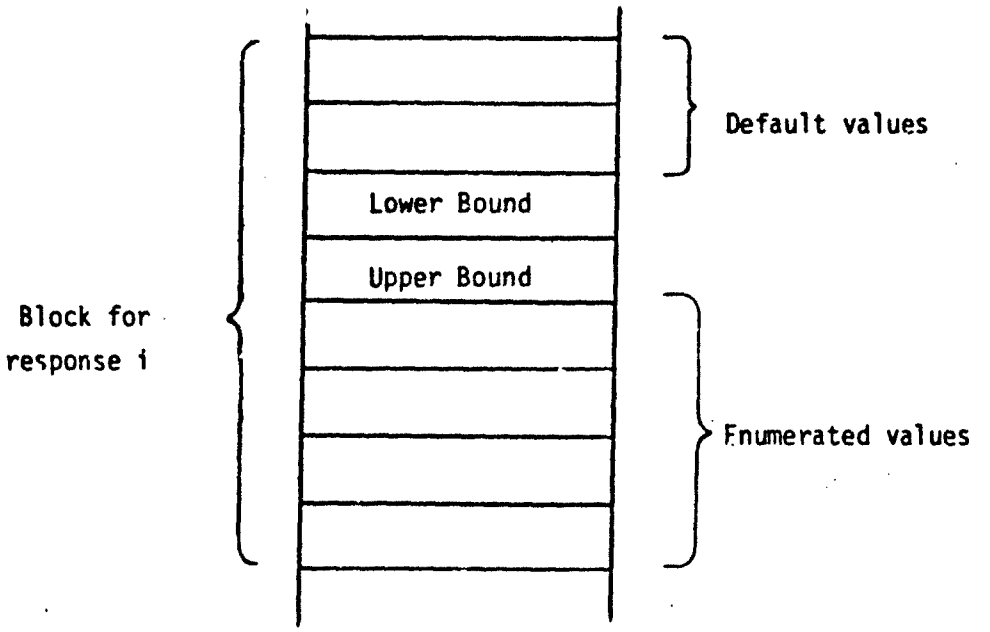
3-0 OUTPUT DATA STRUCTURE

The output data from the FFSP routines is loaded into variables in the COMØ1S common block. The data arrays that make up the output data structure are illustrated in Figure II-3. The next few subsections describe these variables.

3-1. INDEXS(NDEX,4) is an INTEGER array used to store pointers to the prompts and the response output arrays (ROUTS, IOUTS, and COUTS). Each row represents a different response list that was entered. The mixed response lists will occupy more than one row of the INDEXS array (one row for each response list that makes up the mixed response type). The response type values greater than 4000 will not appear in the INDEXS array. Only the components of the mixed response types are contained in INDEXS. The definitions of each of the columns of this array are explained below.

Column 1	Response type
Column 2	Location in the response output array (of the type indicated by the response type in column 1) of the first response in the response list.
Column 3	Location in the response output array of the last response in the response list.
Column 4	The prompt number. (The position in the PROMPT array.)

Block for a response with 2 default values and 4 enumerated values.



Block for response with 1 default value and 0 enumerated values.

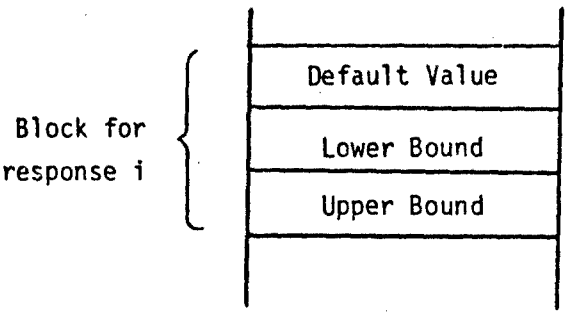
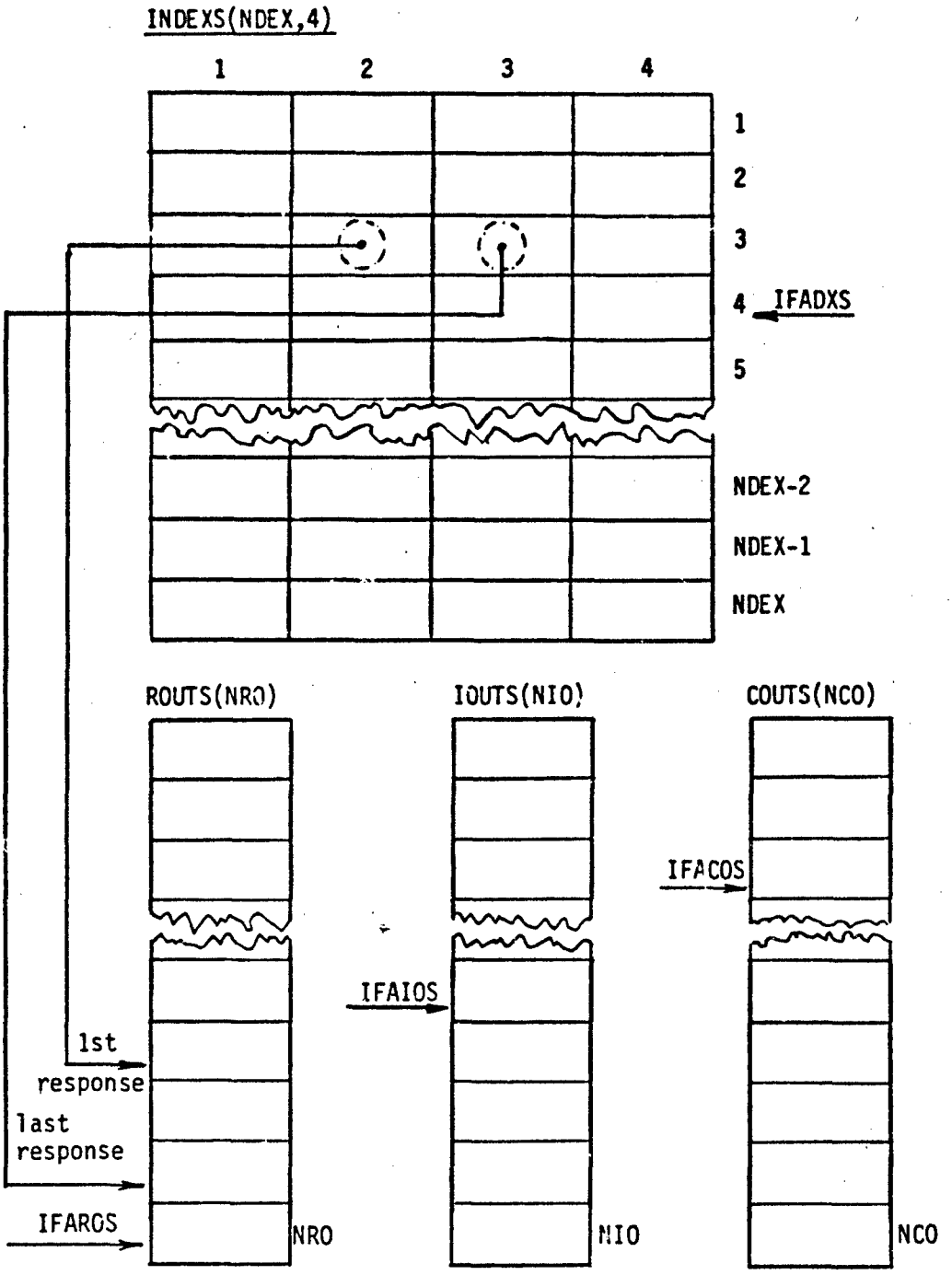


Figure II-2. Block of Elements in a Response Universe Array.



IFAROS, IFAIOS, IFACOS - First available storage location for each type of response.

Figure II-3. Cutput Data Structure.

3-2. ROUTS(NRO),IOUTS(NIO),COUTS(NCO) are the RESPONSE OUTPUT arrays. These arrays contain all the responses entered by the user for a command. The dimensions of these arrays (NRO, NIO, NCO) are set with a PARAMETER statement in every routine with the COMØ1S common block.

3-3. IFAROS,IFAIOS, and IFACOS are POINTERS to the next available locations for storing responses in ROUTS, IOUTS, and COUTS respectively.

III. FLOW OF CONTROL

1-0 DESCRIPTION OF FFSP FLOW OF CONTROL

The flow of control between the MOPADS user interface and the FFSP programs is illustrated in Figure III-1. There are six logical entry points to the FFSP programs (RCMDS, SPROCS, ACCESS, HELPS, GETSUS, and INITS). If any of the FFSP subroutines detect an error in the data structure (not user input errors), execution will be terminated.

NOTE

It is important to extensively debug the data structure in the routines that call the FFSP programs.

The FFSP subroutines are supported by the FFIN2 programs (Polito), and the MOPADS utility programs (Goodin, Polito).

The primary entry point to the actual syntax processor is subroutine SPROCS. There are three logical paths through the syntax processor (see Figure III-1) depending on the form in which the command was entered. If the system encounters "CANC" for a prompt or response, control is returned to the command subroutine with an indication that the command was cancelled.

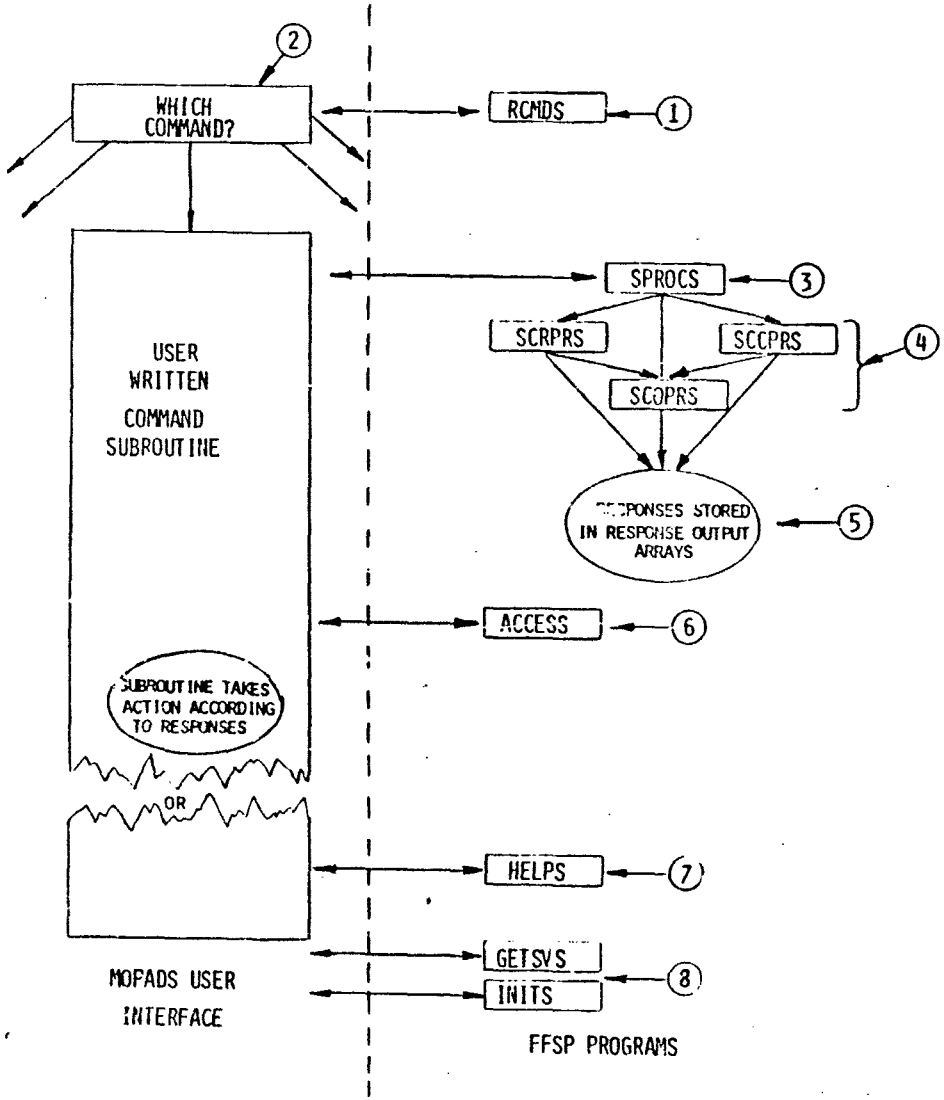


Figure III-1. Flow of Control Using the FFSP Programs.

(1) The user reads the command name with subroutine RCMDS.

(2) The MOPADS user interface decides which command was entered. The correct command subroutine is then called.

(3) The command subroutine issues a call to SPROCS to decode the prompts and load the response output arrays.

(4) One of three subroutines may be called from SPROCS depending on the form of the command. SCRPRS is called if the command is in regular mode. SCCPRS is called is the command is in concise mode. And SCOPRS is called if the command-name-only form was used or if an error occurs in SCRPRS or SCCPRS.

(5) The response output arrays have been loaded with the user responses and control is returned to the command subroutine.

(6) The command subroutine issues calls to FFSP subroutine ACCESS to retrieve responses so the appropriate action may be taken.

(7) If the HELP command is entered, subroutine HELPS is called. HELPS displays information about the command, the prompts, and the responses.

(8) INITS is an initialization program that must be called prior to using FFSP. GETSVS retrieves values of various FFSP options.

IV. FILES

1-0 FFSP FILES

There are two files used by the FFSP routines, 1) the interactive terminal input file and, 2) the interactive terminal output file. The input file is used to enter commands, prompts, responses, and corrections to the syntax processor. The output file is used to output prompts, error messages and general information to the user.

The default unit numbers for the input and output files are 5 and 6, respectively. Their values are assigned to the variables NREADS (input) and NPRNTS (output) of the COMØ1S common block. This assignment is performed in the BLOCKS block data subprogram with a DATA statement. These values may be changed with a call to the INITS subroutine. The following example demonstrates how the values for the input and output unit numbers may be changed to 9 and 15, respectively.

EXAMPLE:

```
LIN  =  9
LOUT =  15
CALL INITS (LIN, LOUT, ' ')
```

V. SUBPROGRAMS

1-0 INTRODUCTION

The subprograms that make up the FFSP module may be divided into two groups:

1. Those that process syntax, and
2. Service or utility routines.

2-0 SYNTAX PROCESSOR ROUTINES

These routines contain the logic to decode user commands into the response output arrays. These routines are discussed in the following subsections.

2-1. Subroutine SPROCS is the primary entry point to the syntax processor. It analyzes the command entered by the user and decides which form (command-name-only, regular mode, concise mode) the command is in. One of the following routines will be called by SPROCS depending on the command form:

1.	SCOPRS	command-name-only
2.	SCRPRS	regular mode
3.	SCCPRS	concise mode

After control is returned to SPROCS and no errors are detected, subroutine UPDTS is called to update any updated default responses. Figure V-1 shows the calling sequence and defines the parameters of SPROCS.

2-2. Subroutine SCOPRS is used to decode the commands entered in the command-name-only form. It is also called if an error is detected in a command entered in regular mode or concise mode. If an error occurs in a response, the system will re-prompt for a valid response. Figure V-2 shows the calling sequence and defines the parameters of SCOPRS.

2-3. Subroutine SCRPRS is used to decode the commands entered in regular mode. If an error is detected in one of the prompt-response groups, all the prompt-response groups after and

```
      SUBROUTINE SPROCS (PROMPT,NPMTS,LTAX,NTAX,RESP,NR,NRESP,NI,
     *                   CRESP,NC,KPMTS,ICNCL)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE DETERMINES WHICH TYPE OF COMMAND WAS
C            ENTERED (REGULAR,CONCISE,COMMAND NAME ONLY) AND CALLS THE
C            APPROPRIATE SYNTAX PROCESSING ROUTINE.
C
C**INPUT PARAMETERS:   PROMPT(NPMTS)=ARRAY OF PROMPT NAMES
C                      NPMTS=NUMBER OF PROMPTS FOR A PARTICULAR COMMAND
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**INPUT/OUTPUT PAR:   KPMTS(NPMTS,2)=ARRAY OF POINTERS TO LTAX ARRAY
C                                     (COLUMN 1) AND A FLAG (COLUMN 2)
C                                     TO INDICATE WHEN A RESPONSE HAS
C                                     BEEN ENTERED
C
C**OUTPUT PARAMETERS:  ICNCL=0 IF COMMAND WAS NOT CANCELLED
C                           =1 IF COMMAND WAS CANCELLED
C
```

Figure V-1. SUBROUTINE SPROCS.

```
      SUBROUTINE SCOPRS (ICODE,PROMPT,NPMTS,LTAX,NTAX,RESP,NR,
     *                   NRESP,NI,CRESP,NC,KPMTS)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS CALLED WHEN THE USER ENTERS A COMMAND
C            NAME ONLY WITH NO PROMPTS OR RESPONSES GIVEN.THIS
C            SUBROUTINE MAY ALSO BE CALLED IF THE USER ENTERS A PARTIAL
C            LIST OF RESPONSES AND ENDS THE COMMAND WITH A DASH.THIS
C            SUBROUTINE PROMPTS THE USER FOR ALL RESPONSES THAT HAVE
C            NOT BEEN ENTERED OR HAVE BEEN ENTERED INCORRECTLY (OR
C            ENTERED AFTER AN INCORRECT ENTRY.
C
C**INPUT PARAMETERS:   ICODE=CODE TO IDENTIFY UNDER WHAT CONDITION
C                           SUBROUTINE WAS CALLED (=0 COMMAND NAME
C                           ONLY ENTERED,=1 PARTIAL LIST OF RESPONSES
C                           ENTERED AND RESPONSES REMAIN THAT HAVE NO
C                           DEFAULT VALUES,OR AN ERROR OCCURRED
C                           ENTERING A RESPONSE)
C                       PROMPT(NPMTS)=CHARACTER ARRAY OF PROMPT NAMES
C                       NPMTS=NUMBER OF PROMPT NAMES IN PROMPT ARRAY FOR
C                           A PARTICULAR COMMAND
C                       LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                    1=RESPONSE TYPE
C                                    2=RESPONSE UNIVERSE TYPE
C                                    3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                    4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                      UNIVERSE ARRAY
C                                    5=NUMBER OF ENUMERATED VALUES IN
C                                      RESPONSE UNIVERSE ARRAY
C                       RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                       NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                       CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**INPUT/OUTPUT PAR:   KPMTS(NPMTS,2)=ARRAY OF POINTERS TO LTAX ARRAY
C                                     (COLUMN 1) AND A FLAG (COLUMN 2)
C                                     TO INDICATE WHEN A RESPONSE HAS
C                                     BEEN ENTERED
C
```

Figure V-2. SUBROUTINE SCOPRS.

including the one with the error, will not be accepted. Control is passed to subroutine SCOPRS, and the system will prompt the user for all un-entered responses. Figure V-3 shows the calling sequence and defines the parameters of SCRPRS.

2-4. Subroutine SCOPRS is used to decode the commands entered in concise mode. If an error is detected in one of the response lists, all the response lists after and including the one with the error will not be accepted. Control is passed to subroutine SCOPRS, and the system will prompt the user for all unentered responses. Figure V-4 shows the calling sequence and defines the parameters of SCOPRS.

2-5. Subroutines GETCHS, GETINS, GETRLS and MLTGTS are used to retrieve the responses entered by the user. GETCHS is used to retrieve character response lists. GETINS is used to retrieve integer response lists. GETRLS is used to retrieve real response lists. MLTGTS is used to call the necessary routines (GETCHS, GETINS, and/or GETRLS) to retrieve mixed responses. These routines use subroutines CHCVLS, CHIVLS, CHRVLS, and CHFLDS to check fields for validity and special case entries such as "DEF" and "CANC." Figure V-5 shows the calling sequences and defines the parameters of GETCHS, GETINS, GETRLS, and MLTGTS.

2-6. Subroutines CHCVLS, CHIVLS, and CHRVLS are used to check the validity of responses retrieved by GETCHS, GETINS, and GETRLS respectively. These routines may check a response to see if it exceeds the upper and/or lower bounds on the response. The responses may also be checked to see if they equal an enumerated value. Figure V-6 shows the calling sequences and defines the parameters of CHCVLS, CHIVLS, and CHRVLS.

2-7. Subroutine CHFLDS is used to check responses for the correct type (character, integer, or real) and to detect the special case responses "DEF" and "CANC." This routine has five alternate returns for special conditions or errors that are detected. Figure V-7 shows the calling sequence and defines the parameters of CHFLDS.

2-8. Subroutine CHDSHS is used to detect a dash at the end of a command-response list. If the character is detected, a flag is set on (=1) and the dash is replaced with a blank. Figure V-8 shows the calling sequence and defines the parameters of CHDSHS.

2-9. Subroutines LDDFVS and MLTLDS are used to load the default value(s) into the response output arrays. Subroutine LDDFVS is called when the user enters "DEF" for a response or he/she has elected to take some of the default responses. MLTLDS is called from LDDFVS when the system is loading the default values for a mixed response type. Figure V-9 shows the calling sequences and defines the parameters of LDDFVS and MLTLDS.

```
      SUBROUTINE SCRPRS (PROMPT,NPMTS,LTAX,NTAX,RESP,NR,NRESP,NI,
     *                   CRESP,NC,KPMTS)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE PERFORMS SYNTAX PROCESSING ON COMMANDS
C            ENTERED IN REGULAR MODE.IF THE LAST NON-BLANK CHARACTER IS
C            A DASH,THE SYSTEM WILL PROMPT THE USER FOR ALL UNENTERED
C            RESPONSES INCLUDING THOSE WITH DEFAULTS.IF THE DASH IS LEFT
C            OFF,.HE SYSTEM PROMPTS FOR RESPONSES WITHOUT DEFAULTS.
C
C**INPUT PARAMETERS:   PROMPT(NPMTS)=CHARACTER ARRAY OF PROMPT NAMES
C                      NPMTS=NUMBER OF PROMPTS FOR A PARTICULAR COMMAND
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY.
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE
C
C**INPUT/OUTPUT PAR:   KPMTS(NPMTS,2)=ARRAY CONTAINS A POINTER TO THE
C                                     LTAX ARRAY (COLUMN 1),AND A FLAG
C                                     TO INDICATE IF THE RESPONSE HAS
C                                     BEEN ENTERED
C
```

Figure V-3. SUBROUTINE SCRPRS.

```
      SUBROUTINE SCCPRS (PROMPT,NPMTS,LTAX,NTAX,RESP,NR,NRESP,NI,
     *                   CRESP,NC,KPMTS)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:   THIS SUBROUTINE PERFORMS SYNTAX PROCESSING ON COMMANDS
C             ENTERED IN CONCISE MODE.IF THE LAST NON-BLANK CHARACTER IS
C             A DASH,THE SYSTEM WILL PROMPT THE USER FOR ALL UNENTERED
C             RESPONSES INCLUDING THOSE WITH DEFAULTS.IF THE DASH IS LEFT
C             OFF,THE SYSTEM PROMPTS FOR RESPONSES WITHOUT DEFAULTS.
C
C**INPUT PARAMETERS:   PROMPT(NPMTS)=ARRAY OF PROMPT NAMES FOR A
C                                    PARTICULAR COMMAND
C                      NPMTS=NUMBER OF PROMPTS FOR A PARTICULAR COMMAND
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**INPUT/OUTPUT PAR:   KPMTS(NPMTS,2)=ARRAY OF POINTERS TO LTAX ARRAY
C                                     (COLUMN 1) AND A FLAG (COLUMN 2)
C                                     TO INDICATE WHEN A RESPONSE HAS
C                                     BEEN ENTERED.
C
```

Figure V-4. SUBROUTINE SCCPRS.

```
      SUBROUTINE GETCHS (MCODE,INFRSP,CRESP,NC,IPTPMT,IDEF,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO READ CHARACTER FIELDS AND LOAD
C            THEM INTO THE CHARACTER RESPONSE OUTPUT ARRAY.
C
C**INPUT PARAMETERS:   MCODE=0 PROMPT FOR RESPONSE WAS SYSTEM GENERATED
C                         =1 PROMPT FOR RESPONSE WAS USER GENERATED
C                      INFRSP(5)=RESPONSE INFORMATION ARRAY
C                              1=RESPONSE TYPE
C                              2=RESPONSE UNIVERSE TYPE
C                              3=POINTER TO RESPONSE UNIVERSE ARRAY
C                              4=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                ARRAY FOR STORING DEFAULTS TO THE
C                                PROMPT
C                              5=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                ARRAY FOR STORING ENUMERATED VALUES
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C                      IPTPMT=POINTER TO PROMPT ARRAY
C
C**INPUT/OUTPUT PAR:   IDEF=-1 ON INPUT IF RESPONSE IS FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF ERROR DETECTED
C                           IN THE RESPONSE
C                          =0 ON INPUT IF RESPONSE IS NOT FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF DEFAULT WAS NOT
C                           TAKEN AND RESPONSE WAS ACCEPTED
C                          =1 ON OUTPUT IF DEFAULT VALUE(S) WERE TAKEN
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN IS TAKEN WHEN AN
C                      ERROR OCCURS PROCESSING A RESPONSE WHEN
C                      MCODE.NE.0.
C
```

Figure V-5. SUBROUTINES GETCHS, GETINS, GETRLS, and MLTGTS.

```
      SUBROUTINE GETINS (MCODE,INFRSP,NRESP,NI,IPTPMT,IDEF,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO READ INTEGER FIELDS AND LOAD
C            THEM INTO THE INTEGER RESPONSE OUTPUT ARRAY.
C
C**INPUT PARAMETERS:   MCODE=0 PROMPT FOR RESPONSE WAS SYSTEM GENERATED
C                          =1 PROMPT FOR RESPONSE WAS USER GENERATED
C                      INFRSP(5)=RESPONSE INFORMATION ARRAY
C                              1=RESPONSE TYPE
C                              2=RESPONSE UNIVERSE TYPE
C                              3=POINTER TO RESPONSE UNIVERSE ARRAY
C                              4=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                ARRAY FOR STORING DEFAULTS TO THE
C                                PROMPT
C                              5=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                ARRAY FOR STORING ENUMERATED VALUES
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      IPTPMT=POINTER TO PROMPT ARRAY
C
C**INPUT/OUTPUT PAR:   IDEF=-1 ON INPUT IF RESPONSE IS FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF ERROR DETECTED
C                           IN THE RESPONSE
C                          =0 ON INPUT IF RESPONSE IS NOT FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF DEFAULT WAS NOT
C                           TAKEN AND RESPONSE WAS ACCEPTED
C                          =1 ON OUTPUT IF DEFAULT VALUE(S) WERE TAKEN
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN IS TAKEN WHEN AN
C                      ERROR OCCURS PROCESSING A RESPONSE WHEN
C                      MCODE.NE.0.
C
```

Figure V-5. (continued)

```
      SUBROUTINE GETRLS (MCODE,INFRSP,RESP,NR,IPTPMT,IDEF,*)
C
C**MODULE:  MOPADS RESP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO READ REAL FIELDS AND LOAD
C            THEM INTO THE REAL RESPONSE OUTPUT ARRAY.
C
C**INPUT PARAMETERS:   MCODE=0 PROMPT FOR RESPONSE WAS SYSTEM GENERATED
C                         =1 PROMPT FOR RESPONSE WAS USER GENERATED
C                      INFRSP(5)=RESPONSE INFORMATION ARRAY
C                               1=RESPONSE TYPE
C                               2=RESPONSE UNIVERSE TYPE
C                               3=POINTER TO RESPONSE UNIVERSE ARRAY
C                               4=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                 ARRAY FOR STORING DEFAULTS TO THE
C                                 PROMPT
C                               5=NUMBER OF ELEMENTS IN RESPONSE UNIVERSE
C                                 ARRAY FOR STORING ENUMERATED VALUES
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      IPTPMT=POINTER TO PROMPT ARRAY
C
C**INPUT/OUTPUT PAR:   IDEF=-1 ON INPUT IF RESPONSE IS FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF ERROR DETECTED
C                           IN THE RESPONSE
C                          =0 ON INPUT IF RESPONSE IS NOT FROM A MIXED
C                           RESPONSE LIST.ON OUTPUT IF DEFAULT WAS NOT
C                           TAKEN AND RESPONSE WAS ACCEPTED
C                          =1 ON OUTPUT IF DEFAULT VALUE(S) WERE TAKEN
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN IS TAKEN WHEN AN
C                      ERROR OCCURS PROCESSING A RESPONSE WHEN
C                      MCODE.NE.0.
C
```

Figure V-5 (continued)

```
      SUBROUTINE MLTGTS (MCODE,LTAX,NTAX,K,IPTPMT,RESP,NR,NRESP,NI,
     *                   CRESP,NC,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO HANDLE THE MIXED
C            RESPONSE TYPE.IT CALLS THE APPROPRIATE COMBINATION OF
C            ROUTINES TO HANDLE DATA ENTRY FOR A MIXED RESPONSE TYPE
C            (CHARACTER-INTEGER,REAL-INTEGER-REAL,ETC.).
C
C**INPUT PARAMETERS:   MCODE=0 PROMPT WAS SYSTEM GENERATED
C                         =1 PROMPT WAS USER ENTERED
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      K=POINTER TO APPROPRIATE COLUMN OF LTAX
C                      IPTPMT=POINTER TO PROMPT ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN OCCURS WHEN AN
C                      ALTERNATE RETURN OCCURS FROM GETINS,GETRLS,OR
C                      GETCHS.
C
```

Figure V-5. (continued)

```
      SUBROUTINE CHCVLS (INF,CRESP,NC,CVAL,*,*,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE CHECKS RESPONSES TO SEE IF THEY ARE WITHIN
C            A SPECIFIED INTERVAL OR EQUAL TO SOME ENUMERATED VALUE.
C            (FOR CHARACTER VALUE CHECKING)
C
C**INPUT PARAMETERS:   INF(5)=RESPONSE INFORMATION ARRAY
C                           1=RESPONSE TYPE
C                           2=RESPONSE UNIVERSE TYPE
C                           3=POINTER TO RESPONSE UNIVERSE ARRAY
C                           4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                             UNIVERSE ARRAY
C                           5=NUMBER OF ENUMERATED VALUES IN
C                             RESPONSE UNIVERSE ARRAY
C                       CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C                       CVAL=VALUE TO BE TESTED
C
C**ALTERNATE RETURNS:  ONE OF THREE ALTERNATE RETURNS MAY BE SELECTED
C                      DEPENDING ON THE VALUE OF THE LOCAL VARIABLE IERR
C                      (E.G. IERR=2 TAKE RETURN 2).
C
```

Figure V-6. SUBROUTINES CHCVLS, CHIVLS, and CHRVLS.

```
      SUBROUTINE CHIVLS (INF,NRESP,NI,IVAL,*,*,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE CHECKS RESPONSES TO SEE IF THEY ARE WITHIN
C            A SPECIFIED INTERVAL OR EQUAL TO SOME ENUMERATED VALUE.
C            (FOR INTERGER VALUE CHECKING)
C
C**INPUT PARAMETERS:   INF(5)=RESPONSE INFORMATION ARRAY
C                           1=RESPONSE TYPE
C                           2=RESPONSE UNIVERSE TYPE
C                           3=POINTER TO RESPONSE UNIVERSE ARRAY
C                           4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                             UNIVERSE ARRAY
C                           5=NUMBER OF ENUMERATED VALUES IN RESPONSE
C                             UNIVERSE ARRAY
C                       NRESP(NI)=RESPONSE UNIVERSE ARRAY
C                       IVAL=VALUE TO BE CHECKED
C
C**ALTERNATE RETURNS:  ONE OF THREE ALTERNATE RETURNS MAY BE SELECTED
C                      DEPENDING ON THE VALUE OF THE LOCAL VARIABLE IERR
C                      (E.G. IERR=2 TAKE RETURN 2)
C
```

Figure V-6. (continued)

```
      SUBROUTINE CHRVLS (INF,RESP,NR,RVAL,*,*,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE CHECKS RESPONSES TO SEE IF THEY ARE WITHIN
C            A SPECIFIED INTERVAL OR EQUAL TO SOME ENUMERATED VALUE.
C            (FOR REAL VALUE CHECKING).
C
C**INPUT PARAMETERS:   INF(5)=RESPONSE INFORMATION ARRAY
C                           1=RESPONSE TYPE
C                           2=RESPONSE UNIVERSE TYPE
C                           3=POINTER TO RESPONSE UNIVERSE ARRAY
C                           4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                             UNIVERSE ARRAY FOR A PARTICULAR PROMPT
C                           5=NUMBER OF ENUMERATED VALUES IN RESPONSE
C                             UNIVERSE ARRAY FOR A PARTICULAR PROMPT
C                       RESP(NR)=RESPONSE UNIVERSE ARRAY
C                       RVAL=VALUE TO BE TESTED
C
C**ALTERNATE RETURNS:  ONE OF THREE ALTERNATE RETURNS MAY BE SELECTED
C                      DEPENDING ON THE VALUE OF THE LOCAL VARIABLE IERR
C                      (E.G. IERR=2 TAKE RETURN 2).
C
```

Figure V-6. (continued)

```
      SUBROUTINE CHFLDS (NCODE,MCODE,ND,ICOND,*,*,*,*,*)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO CHECK FIELDS TO SEE IF THEY WERE
C            CORRECTLY INPUT AS INTEGER,REAL,OR CHARACTER.
C
C**INPUT PARAMETERS:   NCODE=TYPE OF FIELD TO CHECK FOR
C                           4-REAL NUMBER
C                           5-INTEGER NUMBER
C                           6-CHARACTER DATA
C                      MCODE=0 IF PROMPT WAS SYSTEM GENERATED
C                           =1 IF PROMPT WAS USER GENERATED
C                           (NOTE:MCODE MAY BE SET TO 1 TO ALLOW THE
C                           BUFFER TO BE CHECKED TO SEE IF IT IS EMPTY)
C                      ND=NUMBER OF DEFAULT VALUES
C
C**OUTPUT PARAMETERS:  ICOND=0 IF FIELD IS ACCEPTABLE
C                           1 IF FIELD HAS ERRORS
C                           2 IF 'CANC' WAS ENTERED FOR A RESPONSE
C                           3 IF 'DEF' WAS ENTERED FOR A RESPONSE
C                           4 IF STATED DEF AND FIELD HAS NO DEFAULT
C                           5 IF NO MORE FIELDS EXIST
C
C**ALTERNATE RETURNS:  THE ALTERNATE RETURNS OCCUR WHEN ICOND IS NOT
C                      EQUAL TO 0.
C
```

Figure V-7. SUBROUTINE CHFLDS.

```
      SUBROUTINE CHDSHS (CHAR,ISET)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DETERMINE IF THE LAST NON-BLANK
C            CHARACTER ON THE LINE IS CHAR.IF SO,ISET=1 AND THE
C            CHARACTER CHAR IS REPLACED WITH A BLANK IN THE BUFFER.
C
C**INPUT PARAMETERS:   CHAR=CHARACTER TO CHECK
C
C**OUTPUT PARAMETERS:  ISET=0 IF CHAR WAS NOT THE LAST NON-BLANK
C                           CHARACTER IN THE BUFFER
C                      =1 IF CHAR WAS THE LAST NON-BLANK CHARACTER
C                           IN THE BUFFER
C
```

Figure V-8. SUBROUTINE CHDSHS.

```
      SUBROUTINE LDDFVS (NPMTS,LTAX,NTAX,RESP,NR,NRESP,NI,CRESP,NC,
     *                   KPMTS)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE LOADS THE DEFAULT VALUES INTO THE
C            UNENTERED RESPONSES THAT HAVE DEFAULTS.
C
C**INPUT PARAMETERS:   NPMTS=NUMBER OF PROMPTS FOR THE COMMAND
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**INPUT/OUTPUT PAR:   KPMTS(NPMTS,2)=ARRAY OF POINTERS TO LTAX ARRAY
C                                     (COLUMN 1) AND A FLAG (COLUMN 2)
C                                     TO INDICATE WHEN A RESPONSE HAS
C                                     BEEN ENTERED
C
```

Figure V-9. SUBROUTINES LDDFVS and MLTLDS.

```
      SUBROUTINE MLTLDS (M,J,LTAX,NTAX,RESP,NR,NRESP,NI,CRESP,NC)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LOAD THE RESPONSE OUTPUT ARRAYS
C            WITH THE DEFAULT VALUES FOR THE MIXED RESPONSE TYPE.
C
C**INPUT PARAMETERS:   M=POINTER THE PROMPT FOR WHICH THE DEFAULT VALUES
C                        ARE TO BE LOADED
C                      J=POINTER TO THE LTAX ARRAY FOR LOCATING RESPONSE
C                        INFORMATION
C                      LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                      RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
```

Figure V-9. (continued)

2-10. Subroutines DEFS and LDDEFS are used to load a buffer with default values to be printed after a prompt when the user enters a command in the command-name-only form. If no default value(s) is detected the buffer is loaded with the text "NO DEFAULT." These routines are set up to allow the buffer to be printed in several lines if necessary. The maximum number of lines that will be printed is determined by the value of the parameter DBLEN. This parameter is set in a PARAMETER (DBLEN = ...) statement in subroutine SCOPRS. Figure V-10 shows the calling sequences and defines the parameters of DEFS and LDDEFS.

2-11. Subroutine UPDTS is used to update default values that require updating every time a new response is entered for a particular prompt. This routine is called by SPROCS and updates the default values only if no errors were detected by the syntax processor routines. Figure V-11 shows the calling sequence and defines the parameters of UPDTS.

2-12. Subroutine INITS is used to initialize options for the FFSP module. These variables include the following:

1. input unit number (NREADS)
2. output unit number (NPRNTS)
3. default mode (MODES)

Figure V-12 shows the calling sequence and defines the parameters of INITS. The user should call INITS prior to using any other FFSP program. If he does not, NREADS will be 5, NPRNTS will be 6 and the default mode will be regular. INITS may be called at any time to change the values of these options.

3-0 SERVICE AND UTILITY ROUTINES

3-1. Subroutine INITS can also be called as a utility routine to change FFSP options. See Section V, 2-12.

3-2. BLOCKS, Block Data Subprogram, is used to initialize COMMON common block variables. These variables are initialized before execution begins. The variables are as follows:

1. NREADS
2. NPRNTS
3. MODES
4. INITS

See Section VIII for descriptions of these variables.

```
      SUBROUTINE DEFS (INF,RESP,NR,NRESP,NI,CRESP,NC,DEF,DBLEN,J1,IFL,
     *                 J2)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DETERMINE THE DEFAULT VALUES FOR
C            A PARTICULAR PROMPT.
C
C**INPUT PARAMETERS:   INF(5)=RESPONSE INFORMATION ARRAY
C                           1=RESPONSE TYPE
C                           2=RESPONSE UNIVERSE TYPE
C                           3=POINTER TO RESPONSE UNIVERSE ARRAY
C                           4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                             UNIVERSE ARRAY FOR A PARTICULAR RESPONSE
C                           5=NUMBER OF ENUMERATED VALUES IN RESPONSE
C                             UNIVERSE ARRAY FOR A PARTICULAR RESPONSE
C                       RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                       NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                       CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
C**INPUT/OUTPUT PAR:   DEF(DBLEN)=DEFAULT BUFFER FOR LOADING DEFAULT
C                                 VALUES TO BE PRINTED
C                       J1=INPUT AS THE LOCATION IN DEF WHERE DEFAULT
C                          VALUES CAN BE PLACED.OUTPUT AS THE NEXT
C                          AVAILABLE LOCATION FOR STORING DEFAULT VALUES
C                       IFL=NUMBER OF DEFAULT BUFFER ELEMENTS USED
C
C**OUTPUT PARAMETERS:  J2=LOCATION IN DEF WHERE THE LAST PORTION OF A
C                          DEFAULT VALUE WAS PLACED
C
```

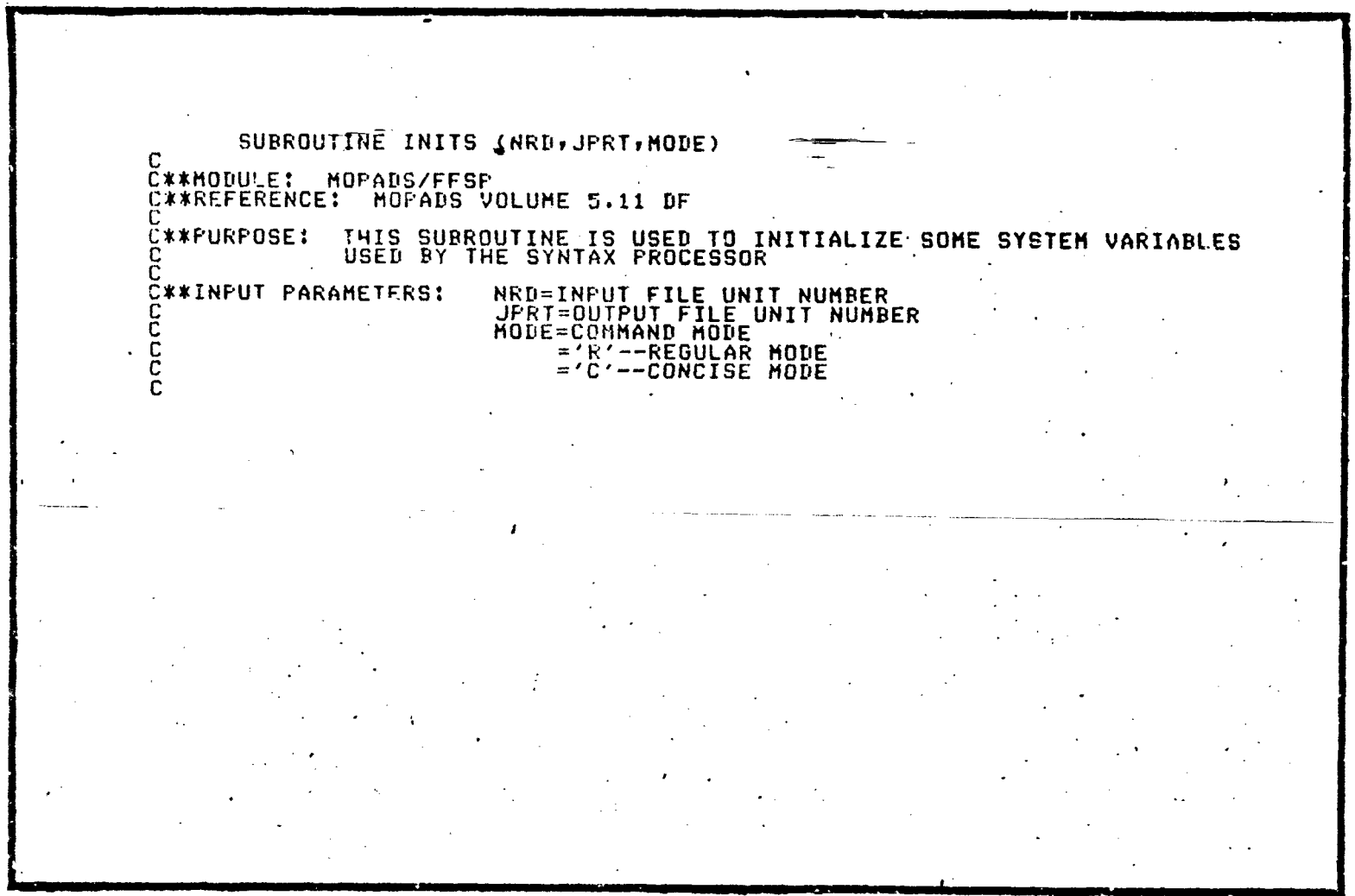
Figure V-10. SUBROUTINES DEFS and LDDEFS.

```
      SUBROUTINE LDDEFS (LTEMP,J1,J2,DEF,DBLEN,IFL)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LOAD THE DEFAULT BUFFER AND
C            EXPAND THE NUMBER OF DEFAULT BUFFERS WHEN NEEDED.
C
C**INPUT PARAMETERS:   LTEMP=TEMPORARY STORAGE FOR VALUE TO BE LOADED
C                           INTO THE DEFAULT BUFFER
C
C**INPUT/OUTPUT PAR:   J1=INPUT AS THE FIRST AVAILABLE LOCATION WHERE
C                         THE DEFAULT BUFFER MAY BE LOADED.OUTPUT AS THE
C                         AVAILABLE LOCATION
C                      J2=LAST LOCATION IN THE DEFAULT BUFFER THAT WAS
C                         LOADED
C                      DEF(DBLEN)=DEFAULT BUFFER CONTAINING ALL THE
C                                 DEFAULTS FOR A PARTICULAR PROMPT OF A
C                                 COMMAND
C                      IFL=CURRENT LENGTH OF THE DEFAULT BUFFER
C
```

Figure V-10. (continued)

```
      SUBROUTINE UPDTS (LTAX,NTAX,KPMTS,NPMTS,RESP,NR,NRESP,NI,CRESP,NC)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO REPLACE THE UPDATED DEFAULT
C            VALUES WITH THE LAST RESPONSES FOR THESE VALUES.
C
C**INPUT PARAMETERS:   LTAX(5,NTAX)=RESPONSE INFORMATION ARRAY
C                                   1=RESPONSE TYPE
C                                   2=RESPONSE UNIVERSE TYPE
C                                   3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                   4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                     UNIVERSE ARRAY
C                                   5=NUMBER OF ENUMERATED VALUES IN
C                                     RESPONSE UNIVERSE ARRAY
C                       KPMTS(NPMTS,2)=ARRAY OF POINTERS TO LTAX ARRAY
C                                      (COLUMN 1) AND A FLAG (COLUMN 2)
C                                      TO INDICATE WHEN A RESPONSE HAS
C                                      BEEN ENTERED
C
C**INPUT/OUTPUT PAR:   RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                      NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                      CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
```

Figure V-11. SUBROUTINE UPDTS.



```
      SUBROUTINE INITS (NRD,JPRT,MODE)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO INITIALIZE SOME SYSTEM VARIABLES
C            USED BY THE SYNTAX PROCESSOR
C
C**INPUT PARAMETERS:   NRD=INPUT FILE UNIT NUMBER
C                      JPRT=OUTPUT FILE UNIT NUMBER
C                      MODE=COMMAND MODE
C                           ='R'--REGULAR MODE
C                           ='C'--CONCISE MODE
C
```

Figure V-12. SUBROUTINE INITS.

3-3. Subroutine GETSVS is used to retrieve the values of FFSP options. These options are:

1. NREADS - input file number
2. NPRNTS - output file number
3. MODES - mode

Figure V-13 shows the calling sequence and defines the parameters of GETSVS.

3-4. Subroutine RCMDS is used to read the first field of the command input (the command itself). This subroutine does no validity check on the command, but returns the command name entered. Figure V-14 shows the calling sequence and defines the parameters of RCMDS.

3-5. Subroutine HELPS is used to print information about the user command. The user is asked to enter a promt name for which information is desired. The following information is printed for any prompt:

1. Prompt name (unabbreviated)
2. Response type (character, integer, real or mixed)
3. Default values, if any
4. Enumerated values, if any
5. Upper and/or lower bounds on the response

Figure V-15 shows the calling sequence and defines the parameters of HELPS.

3-6. Subroutine ACCESS is used to access the response in the response output arrays (ROUTS, IOUTS, COUTS). This routine should only be called after calling SPROCS and generating no errors while in SPROCS. Figure V-16 shows the calling sequence and defines the parameters of ACCESS.

```
      SUBROUTINE GETSVS (NRD,JPRT,MODE)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DETERMINE THE VALUES OF SOME OF
C            THE SYSTEM VARIABLES USED BY THE SYNTAX PROCESSOR.
C
C**OUTPUT PARAMETERS:  NRD=INPUT FILE UNIT NUMBER
C                      JPRT=OUTPUT FILE UNIT NUMBER
C                      MODE=COMMAND MODE
C                           ='R'--REGULAR MODE
C                           ='C'--CONCISE MODE
C
```

Figure V-13. SUBROUTINE GETSVS

```
      SUBROUTINE RCMDS (CMD)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO READ A COMMAND.
C
C**OUTPUT PARAMETERS:  CMD=CHARACTER VARIABLE USED TO HOLD COMMAND READ
C                          FROM INPUT FILE
C
```

Figure V-14. SUBROUTINE RCMDS.

```
      SUBROUTINE HELPS (CMND,PROMPT,NPMTS,KPMTS,LTAX,NTAX,RESP,NR,NRESP,
     *                  NI,CRESP,NC)
C
C**MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO PRINT INFORMATION CONCERNING THE
C            COMMANDS AND THE PROMPTS AND RESPONSES.
C
C**INPUT PARAMETERS:    CMND=COMMAND NAME
C                       PROMPT(NPMTS)=ARRAY OF PROMPT NAMES
C                       NPMTS=NUMBER OF PROMPTS FOR THE COMMAND
C                       KPMTS(NPMTS,2)=COLUMN 1 CONTAINS POINTERS TO THE
C                                      LTAX ARRAY
C                       LTAX(5,NTAX)=RESPONSE INFORMATION ARRARAY
C                                    1=RESPONSE TYPE
C                                    2=RESPONSE UNIVERSE TYPE
C                                    3=POINTER TO RESPONSE UNIVERSE ARRAY
C                                    4=NUMBER OF DEFAULT VALUES IN RESPONSE
C                                      UNIVERSE ARRAY
C                                    5=NUMBER OF ENUMERATED VALUES IN
C                                      RESPONSE UNIVERSE ARRAY
C                       RESP(NR)=REAL RESPONSE UNIVERSE ARRAY
C                       NRESP(NI)=INTEGER RESPONSE UNIVERSE ARRAY
C                       CRESP(NC)=CHARACTER RESPONSE UNIVERSE ARRAY
C
```

Figure V-15. SUBROUTINE HELPS.

```
      SUBROUTINE ACCESS (IPRMTN,NORDER,NNN,JRTCD,RSP,NNR,IRSP,NNI,CRSP,
     *                   NNC,IDFLG)

 **MODULE:  MOPADS/FFSP
C**REFERENCE:  MOPADS VOLUME 5.11 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO ACCESS THE DATA STORED IN THE
C            RESPONSE OUTPUT ARRAYS (ROUTS,IOUTS,COUTS).ONLY ONE
C            RESPONSE TYPE MAY BE ACCESSED IN ONE CALL.
C
C**INPUT PARAMETERS:    IPRMTN=PROMPT NUMBER FOR WHICH YOU WANT THE
C                              RESPONSE FOR (NUMERICAL ORDER IN THE
C                              PROMPT ARRAY)
C                       NORDER=0 IF NOT FROM A MIXED RESPONSE TYPE
C                              =N IF THE RESPONSE DESIRED IS THE N'TH
C                              RESPONSE LIST IN A MIXED RESPONSE
C                       NNN=DIMENSION OF THE RESPONSE ARRAYS
C                       JRTCD=INPUT AS THE RESPONSE TYPE DESIRED WHERE
C                                  1=INTEGER
C                                  2=REAL
C                                  3=CHARACTER
C
C**OUTPUT PARAMETERS:   RSP(NNN)=REAL RESPONSE ARRAY
C                       NNR=NUMBER OF REAL RESPONSES
C                       IRSP(NNN)=INTEGER RESPONSE ARRAY
C                       NNI=NUMBER OF INTEGER RESPONSES
C                       CRSP(NNN)=CHARACTER RESPONSE ARRAY
C                       NNC=NUMBER OF CHARACTER RESPONSES
C                       IDFLG=1 IF RESPONSE ENTERED FOR THE PROMPT WAS
C                             THE DEFAULT RESPONSE
C                            =2 IF RESPONSE ENTERED FOR THE PROMPT WAS
C                             NOT THE DEFAULT RESPONSE
C            NOTE:NNC,NNR,NNI WILL EQUAL ZERO IF A RESPONSE OF THIS TYPE
C                 IS NOT BEING ACCESSED.
C
```

Figure V-16. SUBROUTINE ACCESS.

VI. USER INSTRUCTIONS

1-0 COMMAND DATA SPECIFICATION FORM

When designing the data structure for a user command that will use the FFSP, the user should fill out the Command Data Specification form. An example of this form is shown in Figure VI-1. The next two subsections will discuss how this form is completed and used to design the data structure.

1-1. Instructions for Preparing The Command Data Specification Form.

The letters that are circled in the following instructions correlate with their placement in Figure VI-2.

1) Write the name of the command in space (A).

2) Put the name of the module in space (B).

3) Write the original or revision date in space (C).

4) Briefly explain the purpose of the command in space (D).

5) Write a prompt name in space (E).

6) Determine the response type from the following (n is the length of the list of values; if n=Ø the list is of undefined length).

Code	Response Type
1000+n	list of n integer values
2000+n	list of n real values
3000+n	list of n character values
4000+n*	mixed response

NOTE

For the mixed response type, n is the number of other response type specifications making up the response (e.g., if a mixed response is made up of 1001, 2003, 1002, and 3001 response types, the mixed response type would be equal to 4004). N can not be equal to zero for a mixed response type and none of the response types included in a mixed response type may have

VI. USER INSTRUCTIONS

1-0 COMMAND DATA SPECIFICATION FORM

When designing the data structure for a user command that will use the FFSP, the user should fill out the Command Data Specification form. An example of this form is shown in Figure VI-1. A full sized blank Command Data Specification form is provided at the end of the report. The next two subsections will discuss how this form is completed and used to design the data structure.

1-1. Instructions for Preparing The Command Data Specification Form.

The letters that are circled in the following instructions correlate with their placement in Figure VI-2.

1) Write the name of the command in space (A).

2) Put the name of the module in space (B).

3) Write the original or revision date in space (C).

4) Briefly explain the purpose of the command in space (D).

5) Write a prompt name in space (E).

6) Determine the response type from the following (n is the length of the list of values; if n=Ø the list is of undefined length).

Code	Response Type
1000+n	list of n integer values
2000+n	list of n real values
3000+n	list of n character values
4000+n*	mixed response

NOTE

For the mixed response type, n is the number of other response type specifications making up the response (e.g., if a mixed response is made up of 1001, 2003, 1002, and 3001 response types, the mixed response type would be equal to 4004). N can not be equal to zero for a mixed response type and none of the response types included in a mixed response type may have

COMMAND DATA SPECIFICATION PAGE___OF___

COMMAND_____ PURPOSE _____

MODULE _____

REVISION DATE_____

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX

Figure VI-1. Example Command Data Specification Form.

COMMAND DATA SPECIFICATION PAGE___OF___

COMMAND (A) _____ PURPOSE (D) _____

MODULE (B) _____

REVISION DATE (C) _____

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
(E)	(F)	(G)	(H)	(I)	(J)	(K)

Figure VI-2. Command Data Specification Form.

Write the code value for the response type in space (F). If the mixed response type is selected, steps 6 through 10 will be repeated for each response type making up the response.

7) Determine the response universe type from the following (where numbers greater than 100 represent responses with updated default values).

Code	Response Universe Type
1 or 101	response must be between upper and lower bound
2 or 102	response has an upper bound only
3 or 103	response has a lower bound only
4 or 104	response must be from a set of enumerated value(s)
5 or 105	any response of the type specified is acceptable

For example, if a response must be between 1 and 100, the response universe type would be 1 or 101 depending on the status desired for the default values. If a response must be either "1","3", or "5", the response universe type would be 4 or 104.

Write the code value for the response universe type in space (G).

8) Write any default value(s) in space (H). The values must conform to the response universe specified in step 7 and the response type specified in step 6 . If there are no defaults, leave this space blank.

9) If 4 or 104 was selected for the response universe type in step 7 , write the enumerated values in space (I). If any other code was entered, leave this space blank.

10) If the response universe code selected in step 6 was 1, 101, 2, 102, 3 or 103, the values for the lower and/or upper bounds must be written in spaces (J) and (K), respectively. Upper and/or lower bounds may be specified with response types of "3000+n," but the user should be aware that the character collating sequence is system dependent, and the resulting specification may not be portable.

11) Repeat steps 5 through 10 for all the prompts.

An example of a completed "Command Data Specification Form" is shown in Figure VI-3.

COMMAND DATA SPECIFICATION

PAGE 1 OF 1

COMMAND WORKINFO

PURPOSE EXAMPLE COMMAND

MODULE USER INTERFACE

REVISION DATE 11-03-82

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX
JOBTITLE	3000	5	--	--		
PAYHOUR	2001	1	10.0	--	0.0	100.0
NUMSECRT	1001	104	1	1,3,5	--	--
BLDGCODE	4004	0	--	--	--	--
	3001	4	'A'	'A','B','C','D'	--	--
	1002	1	1000 1001	--	1000	5000
	3001	4	'E'	'E','F'	--	--
	2001	102	6.0	--	--	10.0

Figure VI-3. Example of Completed Command Data Specification Form.

1-2. Creating The Actual Data Structure From The Command Data Specification Form.

The user needs to construct the arrays PROMPT, KPMTS, LTAX, NRESP, CRESP, and RESP from the information entered in the Command Data Specification form. The data structure can be created with the following steps. (Figure VI-4 is an example data structure created using the form in Figure VI-3.)

1) Load the prompt names into the PROMPT array in the same order as they appear on the form. Note the number of prompt names and equate this to the variable NPMTS (the variable name used by the FFSP routines for the number of prompt names for a given command.

2) Put the response type code values into the first row of the LTAX array.

3) Put the response universe type code values into the second row of the LTAX array.

4) Note the number of defaults for each response list and put this value into the fourth row of the LTAX array.

5) Note the number of enumerated values for each response list and put this value into the fifth row of the LTAX array.

NOTE

The third row of the LTAX array will be completed at the time the response universe arrays are created (arrays RESP, NRESP, and CRESP).

6) The KPMTS array contains pointers to the LTAX array where the information for the corresponding prompt (information in row 3 or KPMTS is for the third prompt) begins. These pointers are loaded into the first column of the array. The remainder is left uninitialized.

7) Now fill the response universe arrays (NRESP, CRESP, RESP). Scan the columns of LTAX starting at column 1 and skipping any column whose first row value is greater than 4000.

```
      DIMENSION   PROMPT(4), KPMTS(4,2), LTAX(5,8), RESP(6)
      DIMENSION   NRESP(10), CRESP(14)
      CHARACTER   CRESP* 1 , PROMPT*8

      DATA PROMPT / 'JOBTITLE',
     -              'PAYHOUR',
     -              'NUMSECRT',
     -              'BLDGCODE' /

      DATA ((LTAX(I,J),J=1,8),I=1,5)
     -     / 3000, 2001, 1001, 4004, 3001, 1002, 3001, 2001,
     -         5,    1,  104,    0,    4,    1,    4,  102,
     -         1,    1,    1,    0,    3,    7,   10,    4,
     -         0,    1,    1,    0,    1,    2,    1,    1,
     -         0,    0,    3,    0,    4,    0,    2,    0/

      DATA (KPMTS(I,1), I=1,4)  /1,2,3,4/

      DATA CRESP /' ',' ','A',' ',' ','A','B','C',
     -           'D','E',' ',' ','E','F'

      DATA NRESP /1,0,0,1,3,5,1000,1001,1000,5000/

      DATA RESP /10.0,0.0,100.0,6.0,0.0,10.0/

      NTAX   = 8
      NPMTS  = 4
      NR     = 6
      NC     = 14
      NI     = 10
```

Figure VI-4. Example Data Structure.

8) For each column of LTAX select the correct response universe array as follows:

Response Type Code (row 1 of LTAX)	Response Universe Array
1001+n	NRESP
2000+n	RESP
3000+n	CRESP

9) Note the element number of the next available element in the current response universe array. Put this number in row 3 of the current LTAX column.

10) Load values into the next several elements of the current response universe array according to the rules in Figure VI-5.

11) Repeat steps 8 through 10 for each column of LTAX.

12) Note the defined lengths of NRESP, CRESP, and RESP for use in DIMENSION statements as formal parameters when calling FFSP programs.(Note NTAX, NPMTS, NR, NC, NI in Figure VI-4.)

It is worthwhile to stop now and attempt to duplicate the contents of NRESP, CRESP, and RESP in Figure VI-4 from the specifications in Figure VI-3. This process is very easy once a little practice is obtained.

2-0 WRITING AND USING THE COMMAND SUBROUTINES

2-1. Use Of Subroutine ACCESS.

Subroutine ACCESS has the following calling sequence:

```
SUBROUTINE ACCESS(IPRMTN,NORDER,NNN,JRTCD,RSP,
                  NNR,IRSP,NNI,CRSP,NNC,IDFLG)
```

The definitions of the parameters can be found in Section V, 2-6.

This subroutine retrieves the responses for a specified prompt by searching the INDEXS array (in COMØ1S common block) for the desired prompt number. Once the number has been found, the exact location of the responses can be determined. The designer of the command subroutines must set up three additional arrays for storing responses retrieved by the call to ACCESS. These arrays are CRSP for character string responses, IRSP for integer responses, and RSP for real responses. NNN is a variable used to pass the dimension of the three response arrays to ACCESS. NNN should be set equal to the maximum number of responses of one type that may be entered for a response.

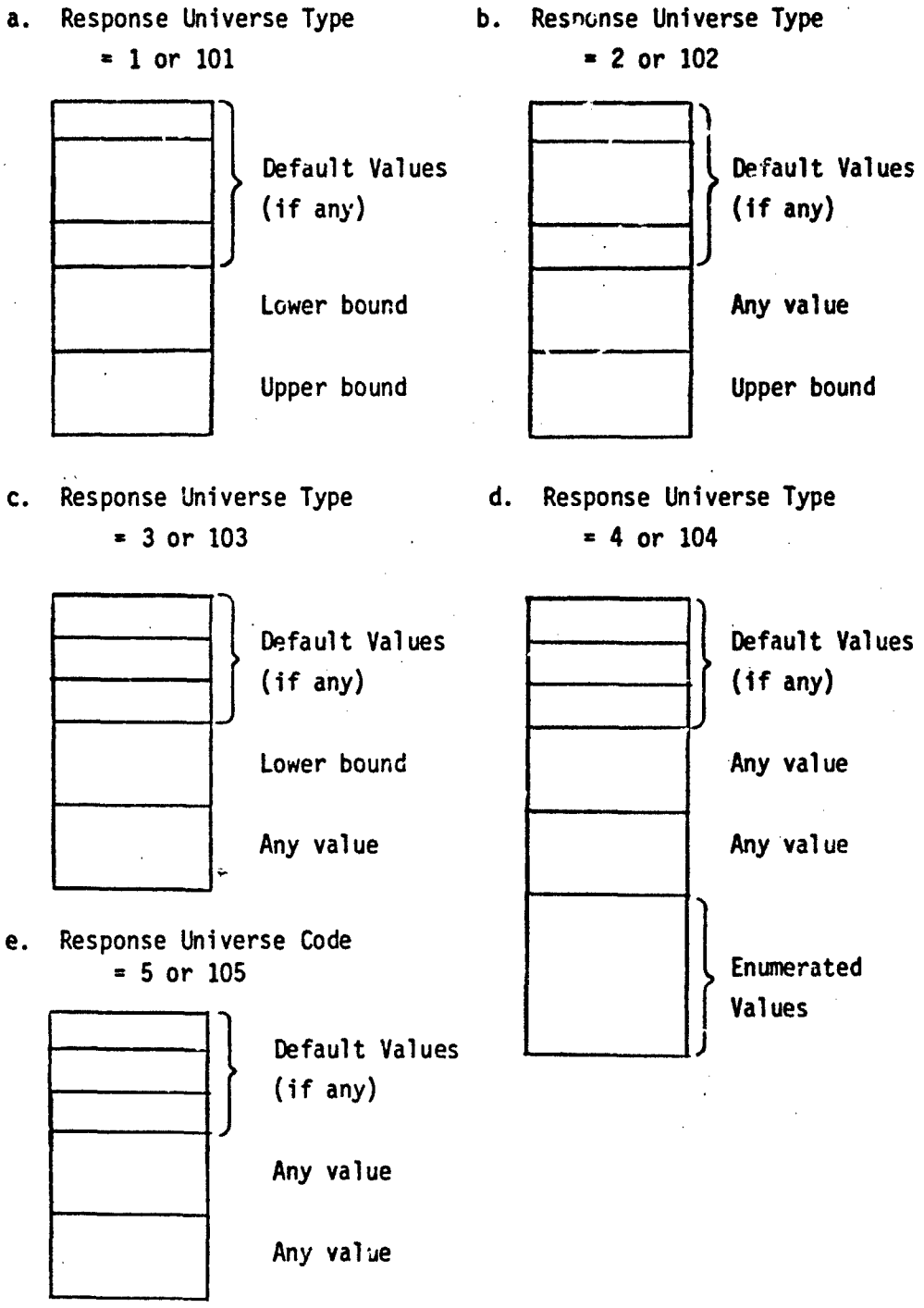


Figure VI-5. Loading Response Universe Arrays Based On The Response Universe Type.

To use the ACCESS subroutine the user must know the prompt number (the order in PROMPT array) for the response he desires, the response type (integer, real, character string), and the order of the response list (1, 2, 3, etc.) in a mixed response (if not from a mixed response the order is zero). The subroutine will return the responses in one of the response arrays (IRSP, RSP, or CRSP), depending on the response type specified. The response array with values in it will have a non-zero indicator value returned from ACCESS, the other two indicators will be equal to zero. These indicators are: NNI for the number of integer responses, NNR for the number of real responses, and NNC for the number of character string responses.

Figure VI-6 illustrates the use of the ACCESS subroutine for various cases.

2-2. Writing The Command Subroutines.

Figure VI-7 illustrates how the data structure and the calls to the FFSP routines are placed in a command subroutine. SPROCS decodes the command. ACCESS is used to retrieve the responses from the decoded command. HELPS is used to print information concerning the command. If the command has no prompts, do not issue calls to SPROCS, ACCESS, or HELPS (These routines are to be used only with commands that have prompts).

3-0 ADDITIONAL INSTRUCTIONS

3-1. How To Change Modes.

The default mode may be changed with a call to subroutine INITS. The following examples show how to change the default mode to concise mode.

```
CHARACTER      MODE*1
     .
     .
     .
MODE = 'C'
CALL INITS (0,0,MODE)
     or
CALL INITS (0,0,'C'
```

To change the default mode to regular mode the user would use 'R' in place of 'C'. For example,

```
CALL INITS (0,0,'R')
```

TO RETRIEVE THE INTEGER RESPONSE TO THE THIRD PROMPT (NOT FROM A MIXED RESPONSE) CALL ACCESS (3,0,NNN,1,RSP,NNR,IRSP,NNI,CRSP,NNC,IDFLG) AFTER THE RETURN NNC = NNR = 0 and NNI > 0
TO RETRIEVE THE FOURTH RESPONSE OF A MIXED RESPONSE FOR PROMPT NUMBER FOUR. CALL ACCESS (4,4,NNN,1,RSP,NNR,IRSP,NNI,CRSP,NNC,IDFLG) NOTE THE SECOND PARAMETER =4 TO GET THE FOURTH RESPONSE LIST FROM THE MIXED RESPONSE AFTER THE RETURN NNC = NNI = 0 and NNR > 0
TO RETRIEVE THE SECOND RESPONSE LIST OF A MIXED RESPONSE FOR PROMPT NUMBER FOUR. CALL ACCESS (4,2,NNN,3,RSP,NNR,IRSP,NNI,CRSP,NNC,IDFLG) AFTER THE RETURN NNI = NNR = 0 and NNC > 0
ADDITIONAL NOTES - NNN must have a value greater than or equal to 1 since it is the dimension for RSP,CRSP, and IRSP. If the response(s) returned by ACCESS were the default response(s), IDFLG will be returned equal to 1. If the user entered a response list and did not take the default, IDFLG will returned equal to 1.

Figure VI-6. Examples of Using Subroutine ACCESS.

```
      SUBROUTINE   WORKI(IOPT)
      DIMENSION    RSP(1),CRSP(10),NRSP(2)
      CHARACTER    CRSP*20

      DATA STATEMENTS FROM FIGURE VI-4

C
C**DECODE COMMAND
C
      IF (IOPT.EQ.1)  GO TO 900
      CALL SPROCS (PROMPT,NPMTS,LTAX,NTAX,RESP,NR,NRESP,NI,
     1             CRESP,NC,KPMTS,ICNCL)
      IF (ICNCL.EQ.1) GO TO 999
C
C**GET RESPONSE TO PROMPT ONE
C
      NNN = 10
      CALL ACCESS (1,0,NNN,3,RSP,NNR,IRSP,NNI,CRSP,NNC,IDFLG)
      .
      .
      . RESPONSE PROCESSING LOGIC
      .
C
C**GET RESPONSE TO PROMPT THREE
C
      NNN = 2
      CALL ACCESS (3,0,NNN,1,RSP,NNR,IRSP,NNI,CRSP,NNC,IDFLG)
      .
      .
      . RESPONSE PROCESSING LOGIC
      .
      ) SIMILAR PROCESSING OF OTHER PROMPTS
      GO TO 999
C
C**HELP COMMAND WAS ENTERED
  900 CALL HELPS ('WORKINFO',PROMPT,NPMTS,KPMTS,LTAX,NTAX,
     1            RESP,NR,NRESP,NI,CRESP,NC)
      GO TO 999
C
  999 RETURN
      END
```

Figure VI-7. Example of Command Subroutine for Example Command from Figure VI-3.

3-2. How To Change The Input And Output Unit Numbers.

The default input and output unit numbers established with the block data subprogram BLOCKS may be changed by a call to subroutine INITS. The following example shows how to change the input and output unit numbers to 9 and 10 respectively.

```
CALL INITS (9,10,' ')
```

3-3. How To Retrieve The Values For The Mode And Input And Output Unit Numbers.

These values may be retrieved with subroutine GETSVS. For example to retrieve the value of the mode in JMODE and the values of the input and output unit numbers in IN and IOUT respectively, use the following call to GETSVS,

```
CHARACTER  JMODE *1
     :
     :
CALL GETSVS (IN,IOUT,JMODE)
```

VII. ERROR PROCESSING

1-0 FFSP ERRORS

There are three types of errors that may be detected by the FFSP subroutines. The error message that is printed is preceded by an error type identifier. These identifiers are as follows:

1. I/O ERROR --
2. DATA STRUCTURE ERROR--
3. ACCESS ERROR --

1-1. I/O errors are generated in the syntax processor routines that decode the user commands. These errors are generated by invalid user input (response) values. The user will be prompted to re-enter the responses if one of these errors occur. These are the least severe errors. FFSP automatically corrects for these problems, and execution is never terminated because of an I/O error.

1-2. Data structure errors are generated when errors in the user defined data structure are detected. These errors are created by situations such as invalid response type code, invalid response universe type codes, and by exceeding the dimensions of some array. Data structure errors should occur only during the debugging phase of developing a program that uses FFSP. These are severe errors and FFSP will terminate execution if it detects an invalid input data structure.

1-3. ACCESS errors are generated in the ACCESS subroutine by illegal parameter values or by exceeding the dimension of an array. These errors will also cause execution to terminate. Errors in the use of the ACCESS subroutine should occur only during the debugging phase of program development.

VIII. COMMON VARIABLE DEFINITIONS

1-0 FFSP COMMON BLOCKS

The FFSP routines use two common blocks (COMØ1S and COMØ2S). COMØ1S common block contains numerical data and the COMØ2S common block contains character data. The COMØ1S variables and their definitions are listed in Table VIII-1. The COMØ2S variables and their definitions are listed in Table VIII-2.

User programs never access any of these variables directly. Therefore, it is not necessary for user programs to contain the FFSP common statements.

Table VIII-1

COMØ1S Common Variables and Definitions

Variable	Definition
ICNCLS	Cancel Command Flag = 0 Command not cancelled = 1 Command cancelled
IFACOS	Pointer to the next available location in the character response output array, COUTS
IFADXS	Pointer to the next available location in the INDEXS array for storing indeces to the response output arrays.
IFAIOS	Pointer to the next available location in the integer response output array IOUTS
IFAROS	Pointer to the next available location in the real response output array ROUTS
IINITS	Initialization Flag = 0 Subroutine INITS was called = 1 Subroutine INITS has not been called
INDEXS(NDEX,4)*	Array of indeces to response output arrays
(__,1)	Response Type 1000 - 1999 = Integer 2000 - 2999 = Real 3000 - 3999 = Character
(__,2)	Pointer to the first location in the response output array (type indicated by column 1 value) where the responses are stored
(__,3)	Pointer to the last location in the response output array (type indicated by column 1 value) where the responses are stored
(__,4)	Prompt number (negative if the response entered was the default).
IOUTS(NCO)*	Integer response output array for storing integer responses

Table VIII-1 (Continued)

Variable	Definition
NPRNTS	Output file unit number
NREADS	Input file unit number
ROUTS (NRO)*	Real response output array for storing real responses

Table VIII-2

COMØ2S Common Variables and Definitions

Variable	Character Length	Definition
COUTS(NCO)*	RSPLEN *	Character response output array for storing character responses
MODES	1	Current command mode. = 'R' for regular mode = 'C' for concise mode

*NOTE-The values for NCO, NIO, NRO, NDEX, and RSPLEN are set with a PARAMETER statement. To change their values simply alter the required portion of the PARAMETER statement and recompile the source code. Current values for their PARAMETERS are NCO=50, NIV=100, NRO=100, NDEX=50, and RSPLEN=30.

IX. REFERENCES

1. Polito, J., MOPADS Free-Format Input Programs (MOPADS/FFIN2), Pritsker & Associates, Inc., MOPADS Volume 5.14 , April 1983.

2. Goodin II, J. Riley and J. Polito, MOPADS Utility Programs, Pritsker & Associates, Inc., MOPADS Volume 5.9 , January 1983.

DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

CHANGE NOTICES

COMMAND DATA SPECIFICATION

PAGE___OF___

COMMAND_____

MODULE _____

REVISION DATE_____

PURPOSE _____

PROMPT	RESPONSE TYPE	UNIVERSE TYPE	DEFAULT VALUES	ENUMERATED VALUES	RANGE	
					MIN	MAX

APPENDIX T

MOPADS FINAL REPORT:

MOPADS USER INTERFACE (MOPADS/UI)

TABLE OF CONTENTS

LIST OF FIGURES

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training arerequired. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

MOPADS Abbreviations

ACC	ADSM Character Code
DBAA	Data Base Access Address
DBAP	Data Base Application Programs
DBCS	Data Base Control System
DL	Data List
DR	Directory
DRAA	Directory Access Address
FFSP	Free Format Syntax Processor
NECC	Number Equivalent of a Character Code
UI	User Interface

I. PURPOSE

This report documents the MOPADS software modules that implement the MOPADS User Interface (UI) and the small main MOPADS module. This is a programmer's manual, not a user manual. User instructions are contained in Polito 1983(a).

The MOPADS user interface provides interactive communication between the user and the MOPADS data base. In particular, the user can perform five functions with the user interface:

1. Creating and editing simulation data sets.
2. Setting up data for a particular simulation
3. Examining statistics after a simulation.
4. Creating and editing air scenario data.
5. Creating and editing new air defense system modules.

This report documents the software that implements all five functions.

In addition, there is a small main module that directs program control to the user interface for user interface sessions or to the MSAINT programs to perform simulations. This module is also described here.

The MOPADS software suffix for the user interface is "I", and for the main module it is "M". All of the program names and COMMON variable names in the user interface end with the letter I. Similarly, all program and COMMON variable names in the main module end with "M".

II. DATA STRUCTURE DESCRIPTIONS

1-0 THE USER INTERFACE

The user interface has several variables and arrays in labeled COMMON storage which are described fully in Section VIII. It does not have large arrays that contain significant amounts of information from the data base. In general, the UI stores only data base addresses to the information it is currently operating on. Furthermore, the UI places the data base in a "write-safe" mode, see Polito 1983(b), which means that information is always written to the data base file as soon as it is changed. This means that it is unlikely that information will be lost even if the program terminates abnormally.

The user interface always has a "current directory" which is the directory whose contents will be displayed if the "DIRECTORY" command is issued, Polito 1983(a), and it may have a current data list. It is the addresses of these current data base entries which are maintained in the UI data structure. Other auxiliary data base entries are also saved as are data specific to the function being performed in the UI.

2-0 THE MAIN MODULE

The main module contains COMMON storage for relatively global data. For example, the unit numbers of all files used by MOPADS are contained in this labeled COMMON area. A complete description is given in Section VIII.

III. OVERVIEW OF THE FLOW OF CONTROL

1-0 THE USER INTERFACE

Figure III-1 shows the structure of the UI. The labels in the boxes are actual subprogram names. MOPADM is the main program for MOPADS. It is contained in the main module. MAINUI is the single entry point to the UI, and it is called whenever a user interface session is begun.

Each of the five UI functions discussed in Section I is implemented with a "subprocess" or small module within the UI. Each subprocess has a single entry point named UI1I, UI2I, UI3I, etc. In addition, all subprograms within a subprocess module ends in the appropriate combination: "1I", "2I", "3I", etc. The subprocesses and their names are shown below:

1. CREATE/EDIT SIMULATION DATA SET
2. SET UP SIMULATION RUN DATA
3. EXAMINE STATISTICS
4. CREATE/EDIT AIR SCENARIO
5. CREATE/EDIT REFERENCE SYSTEM MODULE

Each subprocess has its own set of commands, and each command is implemented in a separate subprogram as shown in the figure. Only the examine statistics (3I) subprocess has been expanded to show the structure. The others are similar. Also, the basic data base commands are available from all subprocesses. They are implemented in the same way as the major subprocess commands.

All subprocesses use the MOPADS Free-Format Syntax Processor, Goodin & Polito 1983, for command processing. Also, programs in MOPADS/FFIN2, Polito 1983(c), MOPADS/DBAP, Polito 1983(d), MOPADS/DBCS, Polito 1983(b), and MOPADS/UTIL, Polito & Goodin 1983, are used extensively by the UI programs.

2-0 THE MAIN MODULE

The primary function of the main module is to read all of the MOPADS data cards and then to call subroutine MAINUI for user interface sessions or subroutines BATCHM for MOPADS simulations. This structure is designed to permit MOPADS to be overlaid with separate UI and simulation overlays. In fact, MOPADS can be linked as two separate load modules: simulation and UI.

SUBROUTINE READM in the main module reads all of the MOPADS data cards. It uses standard FORTRAN 77 list directed input to do

this, so that MOPADS/FFIN2 programs are not required in the root overlay (if MOPADS is overlaid). This ensures that the MSAINT simulation overlay will not contain unnecessary programs.

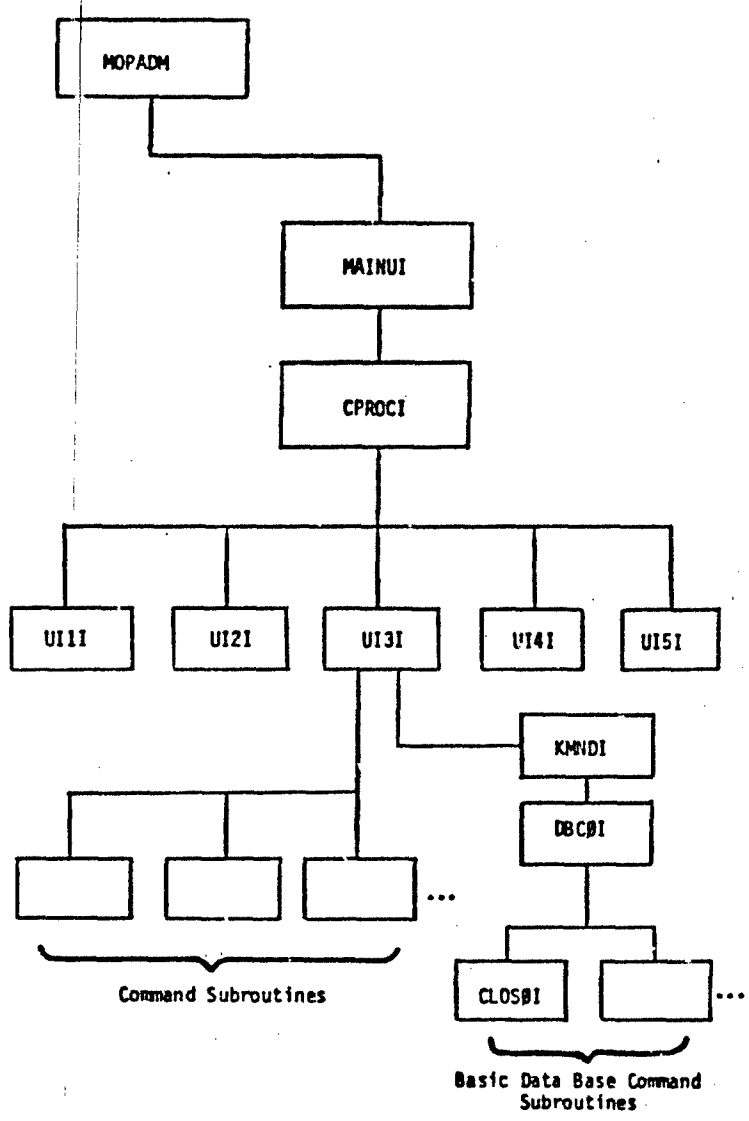


Figure III-1. Software Structure of MOPADS User Interface.

IV. EXTERNAL FILE USAGE

1-0 THE USER INTERFACE

The user interface performs interactive input and output to the MOPADS interactive files, Polito 1983(e). Also, certain options allow the user to print listings on the MOPADS line printer file. All of these files must be opened with job control language prior to executing the UI. Also, the UI reads and writes to MOPADS data base files. These files are opened and closed with calls to the appropriate MOPADS/DBCS programs.

2-0 THE MAIN MODULE

The main module reads the MOPADS data cards from the MOPADS batch input file. It opens and closes the network data files for each system module and copies their contents to the MSAINT network input file which it also opens. These are the only files used by the main module.

V. SUBPROGRAM DESCRIPTIONS

1-0 THE MAIN MODULE

Subprograms in the main module have MOPADS suffix "M".

```
      PROGRAM MOPADM
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       THIS IS THE MAIN PROGRAM FOR MOPADS. MOPADS WILL READ UNIT
C       KUNITM(5) TO DETERMINE IF THIS IS A BATCH SIMULATION OR
C       AN INTERACTIVE USER INTERFACE JOB.  A FILE MUST HAVE
C       BEEN PRE-ASSIGNED TO KUNITM(5).
```

```
      SUBROUTINE BATCHM
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       BATCHM WILL INITIATE BATCH SIMULATION PROCESSING.  IT WILL
C       PERFORM NEEDED INITIALIZATION AND CALL SUBROUTINE MAIN WHICH
C       IS THE SINGLE ENTRY POINT TO MSAINT.
```

```
      BLOCK DATA BLOCKM
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       TO INITIALIZE VARIABLES IN THE MAIN MOPADS MODULE.
C
```

```
      SUBROUTINE GTRUNM(IELL,RUND,LABEL,ACN,IERR,*,*)
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       GTRUNM WILL RETURN CONTENTS OF THE 'RUN-DATA' DATA LIST. IT
C       WILL RETURN EITHER THE 10 DATA WORDS OR A (LABEL,ACN)
C       PAIR DEPENDING UPON THE VALUES OF IELL.
```

```
C--INPUT PARAMETERS:
C       IELL-IF IELL.LE.0, GTRUNM RETURNS THE FIRST 10 WORDS OF
C            THE 'RUN-DATA' DATA LIST. RUND(10) WILL BE THE
C            NUMBER OF CRITICAL ASSETS PROTECTED BY THE AD
C            CONFIGURATION. SEE MOPADS VOL. 5.17 FOR DEFINITIONS
C            OF THE OTHER VALUES.
C
C            IF IELL.GT.0, THEN "LABEL" IS THE LABEL OF A CRITICAL
C            ASSET AND "ACN" IS THE ACN OF THE UNIT ASSIGNED TO
C            PROTECT IT.  THE IELL-TH PROTECTED ASSET IS RETURNED.
C            THE MAX NUMBER OF ASSETS IS FOUND FROM RUND(10) FOR A
C            CALL WITH IELL=0.
C--OUTPUT PARAMETERS:
C       RUND(10)-OUTPUT ARRAY FOR IELL.LE.0. NOT USED IF IELL.GT.0
C       LABEL-(CHARACTER*25) OUTPUT LABEL OF THE IELL-TH PROTECTED
C             SITE IF IELL.GT.0. NOT USED IF IELL.LE.0.
C       ACN-ACN OF THE AD UNIT PROTECTING THE SITE SPECIFIED IN
C           LABEL IF IELL.GT.0. NOT USED IF IELL.LE.0.
C       IERR-ERROR CODE
C            0-NO ERROR
C             NE.0-DB ERROR CODE
C--ALTERNATE RETURNS:
C       1-USED ONLY IF IELL.GT.0 AND IELL.GT.THE TOTAL NUMBER OF
C         PROTECTED SITES.
C       2-DB ERROR(IERR.NE.0)

------------------------------------------------------------------------

      SUBROUTINE GTSDSM(*)
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       GTSDSM WILL LOCATE THE SIMULATION DATA SET AND TACTICAL
C       SCENARIO SPECIFIED FOR A BATCH JOB.  IT WILL ALSO CREATE
C       THE STATISTICS DIRECTORY FOR THE JOB IN THE DATA BASE.
C       GTSDSM USES SOME ROUTINES FROM THE USER INTERFACE TO
C       DO THIS
C--ALTERNATE RETURNS:
C       1-PROCESSING FAILED. ABORT JOB

------------------------------------------------------------------------

      SUBROUTINE INITM
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       INITM INITIALIZES THE MOPADS MAIN MODULE.
C
```

```
      SUBROUTINE READM(ITYPE,*)
C--MODULE: MOPADS MAIN MODULE
C--REFERENCE:  MOPADS VOLUME 5.12
C--PURPOSE:
C       READM WILL READ KUNITM(5) TO DETERMINE IF THIS IS A BATCH
C       OR INTERACTIVE JOB.  KUNITM(5) MUST HAVE BEEN PREVIOUSLY
C       ASSIGNED TO THE INPUT FILE. MOPADS DOES NOT EXPLICITLY
C       OPEN KUNITM(5).
C       READING FROM KUNITM(5) IS DONE WITH LIST DIRECTED INPUT.
C       THE ENTIRE INPUT FILE IS READ. READM OPENS THE DATA
C       BASE FILE IF THIS IS A BATCH JOB, BUT IT DOES NOT
C       LOCATE THE SIMULATION DATA SET OR TACTICAL
C       SCENARIO.
C
C--OUTPUT PARAMETERS:
C       ITYPE-JOB TYPE
C             1-BATCH JOB OF SAINT
C             2-USER INTERFACE SESSION
C--ALTERNATE RETURNS:
C       1-ERROR ON INPUT FILE
```

2-0 USER INTERFACE UTILITIES AND BASIC DATA BASE PROGRAMS

Several programs in the user interface provides utility functions for other user interface modules. These programs end with the single suffix "I". The basic data base programs that implement commands common to all user interface subprocesses end in the suffix "ØI". Both of these collections of programs are included in this section.

```
------------------------------------------------------------------------

      SUBROUTINE MAINUI
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       MAINUI IS THE SINGLE ENTRY POINT TO THE USER INTERFACE MODULE.
C       BLOCK DATA PROGRAM BLOCKI MUST BE LOADED WITH THIS MODULE.
C       UNITS KUNITN(9) AND KUNITN(10) MUST HAVE BEEN
C       ASSIGNED TO THE TERMINAL FOR INTERACTIVE I/O
C       WITH JOB CONTROL LANGUAGE PRIOR TO EXECUTING
C       MOPADS.

------------------------------------------------------------------------

      SUBROUTINE ABORTI
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       TO PRINT THE "COMMAND ABORTED MESSAGE"
C

------------------------------------------------------------------------

      BLOCK DATA BLOCKI
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       BLOCKI INITIALIZES LABELED COMMON IN THE USER INTERFACE.

------------------------------------------------------------------------

      SUBROUTINE CPROCI
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CPROCI IS THE PROGRAM WHICH SELECTS WHICH SUBPROCESS THE USER WIL
C       ENTER.
C

------------------------------------------------------------------------

      SUBROUTINE FNSMI(NDB,IPRT,LABEL,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FNSMI WILL LOCATE A SPECIFIED REFERENCE ADSM IN THE
C       "REFERENCE ADSM" DR ON THE SPECIFIED DB AND MAKE IT
C       THE CURRENT DR.
```

```
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IPRT-PRINT OPTION
C            1-PRINT ERROR MESSAGE IF ADSM NOT PRESENT
C            2-DO NOT PRINT
C       LABEL-CHARACTER LABEL OF THE ADSM TO FIND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-LABEL ADSM NOT FOUND
C       2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE INITI
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       INITI INITIALIZES THE MOPADS USER INTERFACE
C

------------------------------------------------------------------------

      FUNCTION KMNDI(CMNDS,NCMD,NDEX)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       KMNDI READS THE NEXT COMMAND AND IDENTIFIES IT. IF IT IS A BASIC
C       DB COMMAND, IT PROCESSES IT.
C
C--INPUT PARAMETERS:
C       CMNDS(NCMD)-CHARACTER ARRAY OF COMMANDS
C              NCMD-THE NUMBER OF COMMANDS AVAILABLE
C       NDEX(NCMD)-INDEX ARRAY FOR CMNDS FOR USE WITH IBFCU.  ON FIRST
C                  CALL SET NDEX(1)=0
C
C--OUTPUT PARAMETERS:
C       KMNDI =0-IMPLIES COMMAND WAS NOT IDENTIFIED OR WAS PROCESSED BY
C                 KMNDI.  PROMPT FOR NEXT COMMAND.
C             =I-PROCESS COMMAND I
C             =-I PROCESS HELP COMMAND FOR COMMAND I
C
```

```
      SUBROUTINE MNCUI(NDB)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       MNCUI WILL MAKE THE MAIN DIRECTORY CURRENT IN THE
C       USER INTERFACE WITH NO CURRENT DL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
```

```
      SUBROUTINE PLINKI(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       TO MAKE THE OWNER DIRECTORY OF THE CURRENT DIRECTORY THE NEW
C       CURRENT DIRECTORY.  IT IS CALLED AS A RESULT OF THE MOPADS
C       PLINK COMMAND.
C
C--PROMPTS
C         DB-DATA BASE-WORKING/REFERENCE
C
C--INPUT PARAMETERS:
C       IOPT-COMMAND OPTION
C            0-PERFORM COMMAND
C            1-HELP ISSUED
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR CODE IF CANNOT DO COMMAND
C
```

```
      SUBROUTINE CLOSDI (IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO CLOSE A DATA BASE FILE.IT IS
C            CALLED BY ISSUING A "CLOSE" COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "CLOSE" COMMAND:
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C              STATUS=KEEP/DELETE
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THIS COMMAND
C--OUTPUT PARAMETERS:
C                      IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DBCS ERROR CODE IF CANNOT DO COMMAND
C
```

```
      SUBROUTINE CURR0I(IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DISPLAY THE IDENTITY OF THE
C            CURRENT DIRECTORY OR DATA LIST.IT IS CALLED BY ISSUING
C            A "CURRENT" COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "CURRENT" COMMAND:
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C              DISPLAY=WHAT DOES THE USER WANT DISPLAYED
C                      LABEL/ID/ALL INFO
C              TYPE=TYPE OF ENTRY TO DISPLAY
C                   DIRECTORY/DATA-LIST/ALL
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THIS COMMAND
C--OUTPUT PARAMTERS:
C                      IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DBCS ERROR CODE
C
```

```
      SUBROUTINE DBCOI(COMAND,ICMD,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       DBCOI WILL DETERMINE IF A COMMAND IS ONE OF THE LOW LEVEL DB
C       COMMANDS AVAILABLE AT ALL LEVELS. IF SO IT WILL PROCESS THE
C       COMMAND.
C
C--INPUT PARAMETERS:
C       COMAND-CHARACTER VARIABLE TO COMPARE WITH LOW LEVEL COMMANDS.
C--OUTPUT PARAMETERS:
C       ICMD- =-1 COMMAND IS AMBIGUOUS
C             = 0 COMMAND NOT FOUND
C             = I COMMAND IS THE I-TH LOW LEVEL DB COMMAND
C       IERR-ERROR CODE
C            0-NO ERROR OR ICMD.LE.0
C            DBCS ERROR CODE IF LOW LEVEL COMMAND COULD NOT BE PERFORMED
C--ALTERNATE RETURNS:
C       1-ICMD.LE.0
```

```
      SUBROUTINE DEPOOI (IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO SET VALUES IN THE ELEMENTS OF
C            THE CURRENT DATA-LIST.IT IS CALLED BY ISSUING A "DEPOSIT"
C            COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "DEPOSIT" COMMAND:
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THIS COMMAND
C--OUTPUT PARAMETERS:
C                      IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DB ERROR CODE
```

```
      SUBROUTINE DIR01(IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO LIST SELECTED CONTENTS OF THE
C            CURRENT DIRECTORY ON THE WORKING OR REFERENCE DATA BASE.
C            IT IS CALLED BY ISSUING A "DIRECTORY" COMMAND IN THE USER
C            INTERFACE.
C            PROMPTS FOR THE "DIRECTORY" COMMAND:
C              TYPE=TYPE OF CONTENTS USER WANTS LISTED
C                   DIRECTORY/DATA-LIST/ALL
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THIS COMMAND
C**OUTPUT PARAMETERS:
C                      IERR-DB ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DB ERROR CODE
C
```

```
      SUBROUTINE EDDL0I(NDB,IOF1,IOF2,IDBAA,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C      EDDL0I PROMPTS THE USER FOR EDITS OF DATA LISTS ORGANIZED LIKE
C      THE OPERATOR STATE VECTORS. THAT IS, THE COLUMNS IF01 TO IF02 ARE
C      ELIGIBLE TO BE EDITED AND THESE ELEMENTS ARE NUMBERED(FROM THE
C      USER'S STANDPOINT) FROM 1 (CORRESPONDING TO IF01) TO
C      IF02-IF01 (CORRESPONDING TO IFC2). THE USER MAY CHANGE THE
C      VALUES OR LABELS OF THESE ELEMENTS. THE DL MUST BE A REAL, 2-D
C      (WITH 2 ROWS) DL.
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IF01-FIRST COLUMN ELIGIBLE TO BE CHANGED
C      IF02-IF02 IS THE LAST COLUMN ELIGIBLE TO BE CHANGED
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

-----------------------------------------------------------------------

      SUBROUTINE EXAMOI (IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DISPLAY VALUES IN THE ELEMENTS O
C            THE CURRENT DATA-LIST.IT IS CALLED BY ISSUING A "EXAMINE"
C            COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "EXAMINE" COMMAND:
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C              HEADER=DISPLAY HEADER INFO-YES/NO
C              DEVICE=WHERE TO DISPLAY-TERMINAL/PRINTER
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THIS COMMAND
C--OUTPUT PARAMETERS:
C                      IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DB ERROR CODE

-----------------------------------------------------------------------

      SUBROUTINE OPENOI (IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO OPEN A DATA BASE.IT IS CALLED
C            BY ISSUING AN "OPEN" COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "OPEN" COMMAND:
C              FILE=FILE NAME FOR THE DATA BASE
C              STATUS=STATUS-OLD/NEW
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C              MAXRECORD=MAXIMUM NUMBER OF RECORDS IN DATA BASE
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                          =0 PERFORM COMMAND LOGIC
C                          =1 HELP COMMAND ISSUED FOR THE COMMAND
```

```
C--OUTPUT PARAMTERS:   IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DBCS ERROR CODE IF CANNOT DO COMMAND
C
```

```
      SUBROUTINE SELEOI (IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO SELECT THE CURRENT DIRECTORY OR
C            DATA-LIST FROM THOSE ON THE CURRENT DIRECTORY.IT IS CALLED
C            BY ISSUING A "SELECT" COMMAND IN THE USER INTERFACE.
C            PROMPTS FOR THE "SELECT" COMMAND:
C              DB=DATA BASE TYPE-WORKING/REFERENCE
C              TYPE=TYPE OF ENTRY BEING SELECTED
C                   DIRECTORY/DATA-LIST
C              POSITION=DIRECTORY POSITION OF THE ENTRY BEING SELECTED
C              LABEL=LABEL OF THE ENTRY BEING SELECTED
C              ID=ID OF THE ENTRY BEING SELECTED
C            NOTE:THE USER WILL RESPOND TO ONLY ONE OF THE LAST THREE
C                 PROMPTS AND WILL TAKE THE DEFAULT VALUE ON THE OTHER
C                 TWO.
C
C**INPUT PARAMETERS:   IOPT=COMMAND OPTION
C                           =0 PERFORM COMMAND LOGIC
C                           =1 HELP COMMAND ISSUED FOR THE COMMAND
C
C--OUTPUT PARAMTERS:
C                      IERR-ERROR CODE
C                           0-NO ERROR
C                           .NE.0-DB ERROR CODE IF COMMAND COULD NOT BE
C                                 DONE
```

```
      SUBROUTINE TERMOI(IOPT,IERR)
C
C**MODULE:  MOPADS/USER INTERFACE
C**REFERENCE:  MOPADS VOLUME 5.12 DF
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO TERMINATE A TERMINAL SESSION.
C            IT WILL CLOSE ALL OPEN DATA BASES WITH STATUS=KEEP.IT IS
C            CALLED BY ISSUING A "TERMINATE" COMMAND IN THE USER
C            INTERFACE.
C            PROMPTS FOR THE "TERMINATE" COMMAND:
C              NO PROMPTS
```

```
C--INPUT PARAMETERS:
C      IOPT-COMMAND OPTION
C           0-DO COMMAND
C           1-HELP COMMAND
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0-DB ERROR CODE IF CANNOT DO COMMAND
C
```

3-0 1I - CREATE/EDIT SIMULATION DATA SET

The programs in the Create/Edit Simulation Data Set subprocess which end with the suffix "1I" are in this section.

```
------------------------------------------------------------------------

      SUBROUTINE UI1I(JOPT)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       UI1I IS THE ENTRY POINT TO THE "CREATE/EDIT SIMULATION DATA
C       SET" SUBPROCESS
C
C--INPUT/OUTPUT PARAMETERS:
C       JOPT- ON INPUT IT IS 1. IF ON OUTPUT IT IS ZERO, A TERMINATE
C             COMMAND WILL BE EXECUTED.

------------------------------------------------------------------------

      SUBROUTINE ADD1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADD1I WILL PERFORM AN ADD COMMAND IN THE "CREATE/EDIT
C       SIMULATION DATA SET  SUBPROCESS
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR CODE

------------------------------------------------------------------------

      SUBROUTINE CHAN1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHAN1I WILL PERFORM A CHANGE COMMAND IN THE CREATE/EDIT
C       SIMULATION DATA SET SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-PERFORM COMMAND
C            1-PERFORM HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

```
      SUBROUTINE CHFV1I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHFV1I ALLOWS THE USER TO EDIT THE FIELD-OF-VIEW DATA LIST OF
C       CURRENT WORKING ADSM.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-COMMAND CANCELLED
C       2-IERR.NE.0
```

```
      SUBROUTINE CRFV1I(NACC,NACN,LABL12,IDRAA,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRFV1I CREATES THE FIELD OF VIEW DL FOR CODE 12 DIRECTORIES.
C--INPUT PARAMETERS:
C       NACC-NECC(ACC) OF THE OWNER DIRECTORY
C       NACN-THE ACN OF THE OWNER DIRECTORY
C       LABL12-(CHARACTER) LABEL OF THE OWNER DR
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE OWNER DIRECTORY
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE CRTR1I(NACC,LABL12,IDRAA,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRTR1I WILL CREATE THE TRACK-DATA(TRD) DATA LIST OF A
C       CODE 12 DIRECTORY
C--INPUT PARAMETERS:
C       NACC-NECC(ACC) OF THE OWNER DIRECTORY
C       LABL12-(CHARACTER) LABEL OF THE OWNER DIRECTORY
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE OWNER DR
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE DELE1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       DELE1I WILL PERFORM THE "DELETE" COMMAND IN THE "CREATE/EDIT
C       SIMULATION DATA SET" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-PERFORM COMMAND
C            1-PERFORM HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

------------------------------------------------------------------------

      SUBROUTINE FSDS1I(IOPT,IDNO,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FSDS1I WILL LOCATE A PARTICULAR SIMULATION DATA SET IN THE
C       MASTER DIRECTORY OR RETURN AN
C       INDICATION THAT IS IS NOT PRESENT.
C       IF FOUND, IT WILL LOAD ITS DRAA IN IDRF1I(1,1), ITS COMMAND
C       AND CONTROL DIRECTORY IN IDRF1I(1,2) AND ITS COPY COUNTER
C       DBAA IN IDRF(1,3).
C       IT WILL OPTIONALLY MAKE THE SIMULATION DATA SET CURRENT.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            1-DO NOT MAKE THE SIMULTION DATA SET CURRENT
C            2-DO MAKE THE SIM-DATA-SET CURRENT
C       IDNO-SIMULATION DATA SET TO FIND
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C           .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-SIMULATION DATA SET "ID" NOT FOUND
C       2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE FSM1I(IPRT,LABEL,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FSM1I WILL LOCATE A SPECIFIED WORKING ADSM IN THE
C       CURRENT WORKING DR ON THE WORKING DB.
C--INPUT PARAMETERS:
C       IPRT-PRINT OPTION
C            1-PRINT ERROR MESSAGE IF ADSM NOT PRESENT
C            2-DO NOT PRINT
C       LABEL-CHARACTER LABEL OF THE ADSM TO FIND
C--OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE WORKING ADSM. ZERO IF NOT FOUND.
C       IERR-ERROR CODE
C            0-NO ERROR
C           .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-LABEL ADSM NOT FOUND
C       2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE FUSD1I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FUSD1I WILL SEARCH UP THE DIRECTORY TREE FROM THE CURRENT
C       DR TO FIND THE SIMULATION DATA SET DR. THEN IT WILL SET
C                    IDRF1I(-,1)=DRAA OF THE SIMULATION DATA SET
C                    IDRF1I(-,2)=COMMAND AND CONTROL DR
C                    IDRF1I(-,3)=COPY COUNTER DL
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C           .NE.0-DB ERROR CODE
C--ALTERNATE RETURNS:
C       1-NO SIM. DATA SET FOUND
C       2-IERR.NE.0
```

```
      SUBROUTINE GTVW1I(IVWR,I1,I2,IDRAA,DLS,NDLS,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       GTVW1I WILL RETURN PORTIONS OF THE A VIEWER ROW IN A FIELD
C       OF VIEW DATA LIST. GTVW1I DOES NO CHECKING ON THE INPUT
C       PARAMETERS FOR VALIDITY.
C--INPUT PARAMETERS:
C       IVWR-VIEWER NUMBER(ACTUALLY, THE ROW NUMBER)
C       I1,I2-GET COLUMN I1 TO I2
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE 'FV' DL
C--OUTPUT PARAMETERS:
C
C      DLS(NDLS)-OUTPUT ARRAY. ELEMENTS I1 TO I2 WILL BE CHANGED
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR=6. DLS NOT CHANGED. ROW IVWR DOES NOT EXIST.
C       2-IERR.NE.6.AND.IERR.NE.0
```

```
      SUBROUTINE INSR1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C      INSR1I WILL PERFORM THE "INSERT" COMMAND IN THE "CREATE/EDIT
C      SIMULATION DATA SET" SUBPROCESS.
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
```

```
      SUBROUTINE MXSQ1I(NECA,IDRAA,NSEQ)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       MXSQ1I WILL SEARCH A CODE 7 OR A CODE 12 DR FOR OWNED DR'S OF
C       CODE 12 (ADSM'S) THAT HAVE A PARTICULAR ACC. IT WILL FIND
C       THE LARGEST SEQUENCE NUMBER OF THAT TYPE USED SO FAR.
C--INPUT PARAMETERS:
C       NECA-NECC(ACC)-THE NUMBER EQUIVALENT CHARACTER CODE OF THE
C            TARGET ACC.
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-THE DRAA OF THE DR TO SEARCH.  ON RETURN IT WILL BE
C                POSITIONED TO THE OWNED DR WITH THE LARGEST
C                (OR UNCHANGED IF NSEQ=0)
C--OUTPUT PARAMETERS:
C       NSEQ-LARGEST SEQUENCE NUMBER USED SO FAR(ZERO IF NONE)

------------------------------------------------------------------------

      SUBROUTINE REMV1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       REMV1I WILL PERFORM THE "REMOVE" COMMAND IN THE "CREATE/EDIT
C       SIMULATION DATA SET" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-PERFORM COMMAND
C            1-PERFORM HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

------------------------------------------------------------------------

      SUBROUTINE RFAD1I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RFAD1I WILL LOCATE THE "(MASTER-DIRECTORY)" AND PUT ITS
C       IDRAA IN MAIN1I.
C       IT WILL ALSO OPTIONALLY MAKE THE "(MASTER-DIRECTORY)"
C       CURRENT.
```

```
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            1-DO NOT MAKE THE MASTER DIRECTORY DR CURRENT
C            2-DO MAKE THE MASTER DIRECTORY DR CURRENT
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .GT.0-DB ERROR CODE
C            .LT.0-CANNOT LOCATE "(MASTER-DIRECTORY)"
```

4-0 2I - SET UP SIMULATION RUN DATA

The programs in the Set Up Simulation Run Data subprocesses which end with the suffix "2I" are in this section.

```
      SUBROUTINE UI2I(JOPT)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       UI2I IS THE ENTRY POINT TO THE "SET-UP SIMULATION RUN DATA"
C       SUBPROCESS
C
C--INPUT/OUTPUT PARAMETERS:
C       JOPT- ON INPUT IT IS 2. IF ON OUTPUT IT IS ZERO, A TERMINATE
C             COMMAND WILL BE EXECUTED.
```

```
      SUBROUTINE ADD2I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADD2I WILL PERFORM THE ADD COMMAND IN THE SET UP SIMULATION
C       RUN DATA SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

```
      SUBROUTINE CRIT2I(IDRT,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRIT2I ALLOWS EDITING OF THE FIRE-UNIT TO CRITICAL-ASSET
C       ASSIGNMENTS.
C--INPUT/OUTPUT PARAMETERS:
C       IDRT(2)-DBAA OF THE RUN-DATA DL OF A TACTICAL SCENARIO.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE DELE2I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       DELE2I WILL PERFORM THE DELETE COMMAND IN THE SET UP SIMULATION
C       RUN DATA SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

      SUBROUTINE EDIT2I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       EDIT2I WILL PERFORM THE EDIT COMMAND IN THE SET UP SIMULATION
C       RUN DATA SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

      SUBROUTINE FTS2I(IPRT,IDNO,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FTS2I WILL LOCATE A SPECIFIED TACTICAL SCENARIO DR IN THE
C       CURRENT DR ON THE WORKING DB.
C--INPUT PARAMETERS:
C       IPRT-PRINT OPTION
C            1-PRINT ERROR MESSAGE IF SCENARIO NOT PRESENT
C            2-DO NOT PRINT
C       IDNO-TACTICAL SCENARIO NUMBER
```

```
C--OUTPUT PARAMETERS:
C       IDRAA(1)-DRAM OF THE TACTICAL SCENARIO. ZERO IF NOT FOUND.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-TACTICAL SCENARIO NOT FOUND
C       2-IERR.NE.0

-----------------------------------------------------------------------

      SUBROUTINE USE2I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       USE2I WILL PERFORM THE USE COMMAND IN THE SET UP SIMULATION
C       RUN DATA SUBPROCESS
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
```

5-0 3I - EXAMINE STATISTICS

The programs in the Examine Statistics subprocesses which end with the suffix "3I" are in this section.

```
      SUBROUTINE UI3I(JOPT)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       UI3I IS THE ENTRY POINT TO THE "EXAMINE STATISTICS"
C       SUBPROCESS
C--INPUT/OUTPUT PARAMETERS:
C       JOPT-ON INPUT IT IS 3. IF ON OUTPUT IT IS ZERO, A TERMINATE
C            COMMAND WILL BE EXECUTED
```

```
      SUBROUTINE ADSM3I(NR1,NR2,ITYP)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADSM3I WILL QUERY THE USER FOR AN ADSM LABEL OR AN ADSM ACC.
C       IT WILL RETURN THE TYPE OF THE ADSM AND THE COPY ROW NUMBER(S)
C       OF THAT TYPE
C--OUTPUT PARAMETERS:
C       NR1,NR2-COPY ROW NUMBER(S) OF THE SPECIFIED ADSM.
C               FOR EXAMPLE: IF THE USER RESPONDS WITH "G1H2",
C                            THEN NR1=NR2=THE COPY ROW NUMBER OF
C                            THAT UNIT
C                            IF THE USER RESPONDS WITH "H",AN ACC, THEN
C                            NR1=1 AND NR2=NCNEXT-1. I.E. ALL THE
C                            COPY ROW NUMBERS
C               IF NR1=0 ON RETURN, THEN RETURN TO THE SECONDARY MENU
C       ITYP-THE SYSTEM MODULE TYPE OF THE SPECIFIED ADSM. E.G.
C            FOR AN IHAWK, ITYP=4
```

```
      SUBROUTINE CA103I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CA103I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF OPERATOR TASK FRACTION STATISTICS
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
```

```
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C      1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CAT33I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C      CAT33I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C      OF AIR DEFENSE UNIT.
C--INPUT PARAMETERS:
C      IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C          FOR THE VALUES.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C      1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CAT43I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C      CAT43I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C      OF OPERATOR
C--INPUT PARAMETERS:
C      IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C          FOR THE VALUES.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE CAT53I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CAT53I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF AIR SCENARIO.
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE CAT63I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CAT63I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF SAINT TASK NODE STATISTICS.
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE CAT73I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CAT73I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF SAINT RESOURCE STATISTICS.
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CAT83I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CAT83I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF SAINT USER STATISTICS.
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CAT93I(IC2,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CAT93I IMPLEMENTS THE PRINT COMMAND FOR THE MAIN CATEGORY
C       OF MOPADS COUNT STATISTICS.
C--INPUT PARAMETERS:
C       IC2-THE SECONDARY CATEGORY. SEE ISEC IN SUBROUTINE MENU3I
C           FOR THE VALUES.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE DISP3I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       DISP3I WILL PERFORM THE DISPLAY COMMAND IN THE "EXAMINE
C       STATISTICS" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR CODE
```

```
      SUBROUTINE FDOP3I(IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FDOP3I WILL FIND THE ACN'S OF ALL ADSM'S AND LABELS OF ALL
C       OPERATOR TYPES FOR USE IN THE "EXAMINE STATISTICS"
C       SUBPROCESS. FDOP3I ASSUMES THAT EACH ADSM TYPE IN THE
C       SIMULATION DATA SET IS REPRESENTED BY A REFERENCE ADSM.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0- NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE HEAD3I(ISKIP)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       HEAD3I WILL PRINT A HEADING CONSISTING OF THE
C
C                  DB NAME
C                  DB TITLE
C                  SIMULATION DATA SET NUMBER
C                  TACTICAL SCENARIO NUMBER
C                  RUN NUMBER
C
C       IT IS PRINTED ONLY ON THE MOPADS OUTPUT FILE, SO IT IF
C       THE DISPLAY OPTION (IST3I(4)) IS 1, HEAD3I HAS NO EFFECT.
C--INPUT PARAMETERS:
C       ISKIP-PAGE SKIP OPTION
C             1-SKIP PAGE BEFORE PRINTING
C             2-DO NOT SKIP PAGE
C             3-DO PAGE SKIP ONLY(DO NOT PRINT HEADER)
```

```
      SUBROUTINE LNFD3I(KLNS)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       LNFD3I WILL LINE FEED ON EACH DISPLAY DEVICE SPECIFIED BY THE
C       DISPLAY OPTION, IST3I(4), IN THE "EXAMINE STATISTICS"
C       SUBPROCESS.
C--INPUT PARAMETERS:
C       KLNS-NUMBER OF LINES TO LINE FEED(MAY BE ZERO)
```

```
      SUBROUTINE MENU3I(ISCR,ISEL)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       MENU3I WILL PRINT ONE OF THE STATISTICS MENU SCREENS AND
C       RETURN THE USER'S MENU SELECTION.
```

```
C--INPUT PARAMETERS:
C      ISCR-SCREEN NUMBER TO PRINT. ISCR IS ONE OF THE FOLLOWING:
C           IF ISCR=0, RETURN WITH NO ACTION
C           IF ISCR.LT.0, THE MENU WILL NOT BE PRINTED, ONLY A
C              RESPONSE WILL BE REQUESTED.
C
C           1-MAIN CATEGORY MENU
C           2-2NDARY CATEGORY MENU FOR MAIN CATEGORY OF AIR DEFENSE
C                                                              UNIT
C           3-"                                          " OPERATORS
C           4-"                                          " AIR SCENARIO
C           5-"                                          " SAINT TASK
C                                                          NODE STATS
C           6-"                                          " SAINT RESOURCE
C                                                          STATS
C           7-"                                          " SAINT USER STATS
C           8-"                                          " MOPADS COUNT
C                                                          STATS
C           9-"                                          " MOPADS OPERATOR
C                                                          TASK FRACTION
C                                                          STATS
C          10-"                                          " MOPADS TRACE
C--OUTPUT PARAMETERS:
C      ISEL-THE USER'S MENU SELECTION.  ONLY A SUBSET OF THE
C           FOLLOWING RESPONSES IS VALID FOR EACH SCREEN SHOWN ABOVE.
C           MENU INSURES THAT A RESPONSE VALID FOR THE CURRENT
C           SCREEN IS GIVEN.  IF THE USER GIVES THREE INVALID
C           RESPONSES IN A ROW, ISEL=1 IS RETURNED.
C
C           1-EXIT  PRINT COMMAND(-)
C           2-RETURN TO MAIN MENU(0)
C           3-AIR DEFENSE UNIT(1)
C           4-OPERATORS(2)
C           5-AIR SCENARIO(3)
C           6-SAINT TASK NODE STATISTICS(A)
C           7-SAINT RESOURCE STATISTICS(B)
C           8-SAINT USER STATISITICS(C)
C           9-MOPADS COUNT STATISTICS(D)
C          10-MOPADS OPERATOR TASK FRACTION STATISITICS
C          11-MOPADS TRACE
C          12-TOGGLE DISPLAY DEVICE(*)
C          13-ALL STATISTICS(EXCEPT TRACE)(+)
C          14-PRINT MENU AGAIN(?)
C          15-TIME INTERVAL(TI)
C          16-PRINT TRACE(PR)
C
```

```
      SUBROUTINE OPER3I(ITYP)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       OPER3I WILL QUERY THE USER FOR AN OPERATOR LABEL.
C       FOR EXAMPLE, "ASO" FOR THE IHAWK ASO.
C       ALSO, "*" CAN BE USED FOR THE
C       OPERATOR LABEL TO SIGNIFY ALL OPERATORS. E.G. "*"
C       FOR ALL OPERATORS.
C--OUTPUT PARAMETERS:
C       ITYP-OPERATOR TYPE.
C            IF ITYP.GT.0, IT IS THE OPERATOR TYPE
C            IF ITYP.EQ.0, RETURN TO MENU
C            IF ITYP.EQ.-1, AN "*" WAS ENTERED
```

```
      SUBROUTINE PCOU3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PCOU3I WILL PRINT COUNTER STATISTICS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE STATISTIC TO PRINT. IF ICLCT=0,
C            PRINT ALL STATISTICS FOR THIS UNIT. IF ICLCT.LT.0,
C            DO NOT PRINT HEADER.
C       NRN-RUN NUMBER FOR THESE STATISTICS.
```

```
      SUBROUTINE PRES3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PRES3I WILL PRINT RESOURCE STATISTICS
C       FOR ONE ADSM. PRES3I WILL NOT PRINT RESOURCES WHOSE
C       LABEL IS BLANK.
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE RESOURCE TO PRINT. IF ICLCT=0,
C            PRINT ALL RESOURCE STATISTICS FOR THIS UNIT. IF
C            ICLCT.LT.0, DO NOT PRINT HEADER.
C       NRN-NUMBER OF RUNS FOR THESE STATISTICS.
```

```
      SUBROUTINE PRIN3I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PRIN3I WILL EXECUTE THE PRINT COMMAND IN THE 'EXAMINE
C       STATISTICS" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .GT.0-DBCS ERROR CODE
```

```
      SUBROUTINE PRLN3I(LINL)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PRLN3I WILL PRINT A LINE OF STATISTICS INFORMATION IN
C       VARIABLE LINE3I. IT WILL PRINT IT ON ALL OUTPUT DEVICES
C       SPECIFIED BY THE DISPLAY OPTION, IST3I(4), OF THE "EXAMINE
C       STATISTICS" SUBPROCESS. COLUMN 1 OF LINE3I WILL NOT BE
C       PRINTED ON THE TERMINAL. IT WILL BE PRINTED ON THE LINE
C       PRINTER DEVICE FOR CARRIAGE CONTROL.
C--INPUT PARAMETERS:
C       LINL-LINE LENGTH
C            1-72 COLUMNS (ONLY THE 1ST 72 COLUMNS OF LINE3I WILL BE
C              PRINTED)
C            2-130 COLUMNS
```

```
      SUBROUTINE PTFS3I(NROW,ICLCT,NRN,IDOP)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PTFS3I WILL PRINT OPERATOR TASK STATISTICS
C       FOR ONE ADSM AND ONE OR MORE OPERATORS.
```

```
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-TASK NUMBER OF THE STATISTIC TO PRINT. IF ICLCT=0,
C             PRINT ALL STATISTICS FOR THIS UNIT. IF ICLCT.LT.0,
C             DO NOT PRINT HEADER.
C       NRN-RUN NUMBER FOR THESE STATISTICS.
C       IDOP-OPERATOR ID. IF IDOP=0,PRINT FOR ALL OPERATORS IN THE ADSM.

------------------------------------------------------------------------

      SUBROUTINE PTHS3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PTHS3I WILL PRINT TASK HISTOGRAMS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE HISTOGRAM TO PRINT. IF ICLCT=0,
C             PRINT ALL HISTOGRAMS FOR THIS UNIT.
C       NRN-RUN NUMBER FOR THESE HISTOGRAMS.

------------------------------------------------------------------------

      SUBROUTINE PTSR3I(NROW,NODE,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PTSR3I WILL PRINT TASK RUN STATISTICS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       NODE-NODE NUMBER OF THE STATISTICS TASK TO PRINT. IF NODE=0,
C             PRINT ALL NODE STATISTICS FOR THIS UNIT. IF NODE.LT.0,
C             DO NOT PRINT HEADER.
C       NRN-RUN NUMBER FOR THESE STATISTICS.
```

```
      SUBROUTINE PTSS3I(NROW,NODE,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PTSS3I WILL PRINT TASK SUMMARY STATISTICS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       NODE-NODE NUMBER OF THE STATISTICS TASK TO PRINT. IF NODE=0,
C            PRINT ALL NODE STATISTICS FOR THIS UNIT. IF NODE.LT.0,
C            DO NOT PRINT HEADER.
C       NRN-NUMBER OF RUNS FOR THESE STATISTICS.
```

```
      SUBROUTINE PUCL3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PUCL3I WILL PRINT USER COLLECTED OBSERVATION STATISTICS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE STATISTIC TO PRINT. IF ICLCT=0,
C            PRINT ALL STATISTICS FOR THIS UNIT. IF ICLCT.LT.0,
C            DO NOT PRINT HEADER.
C       NRN-RUN NUMBER FOR THESE STATISTICS.
```

```
      SUBROUTINE PUHS3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PUHS3I WILL PRINT USER HISTOGRAMS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE HISTOGRAM TO PRINT. IF ICLCT=0,
C            PRINT ALL HISTOGRAMS FOR THIS UNIT.
C       NRN-RUN NUMBER FOR THESE HISTOGRAMS.
```

```
      SUBROUTINE PUTM3I(NROW,ICLCT,NRN)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       PUTM3I WILL PRINT USER COLLECTED TIME PERSISTENT STATISTICS
C       FOR ONE ADSM
C--INPUT PARAMETERS:
C       NROW-COPY ROW NUMBER OF THE ADSM
C       ICLCT-INDEX NUMBER OF THE STATISTIC TO PRINT. IF ICLCT=0,
C             PRINT ALL STATISTICS FOR THIS UNIT. IF ICLCT.LT.0,
C             DO NOT PRINT HEADER.
C       NRN-RUN NUMBER FOR THESE STATISTICS.
```

```
      SUBROUTINE RCDB3I(N1,N2,N3,IDL,IELL,LRAY,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RCDB3I WILL READ ANY HOLLERITH ARRAY WITH UP TO THREE
C       DIMENSIONS FROM A MOPADS DATA BASE, CHARACTER, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO READ MSAINT ARRAYS FROM THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       WRITTEN TO THE DATA BASE WITH SUBROUTINE
C       WCDB3I.
C
C       ----NOTE---- HOLLERITH IS NOT A STANDARD FORTRAN 77 DATA
C                    TYPE. THIS PROGRAM IS INCUDED FOR COMPATIBILITY
C                    WITH MSAINT WHICH USES HOLLERITH FOR LABELS.
C
C--INPUT PARAMETERS:
C       N1,N2,N3-DIMENSIONS OF LRAY. SEE BELOW.
C--INPUT/OUTPUT PARAMETERS:
C       IDL(2)-DBAA OF THE DATA LIST TO READ FROM
C       IELL-ON INPUT, THE ELEMENT OF THE DATA LIST WHICH
C            CONTAINS THE FIRST ELEMENT OF LRAY. ON RETURN,
C            THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C            ELEMENT OF LRAY.
```

```
C--OUTPUT PARAMETERS:
C       LRAY(N1,N2,N3)-THE HOLLERITH ARRAY TO READ. IF THE ARRAY
C                      HAS FEWER THAN 3 DIMENSIONS, SEND
C                      N2 AND/OR N3 EQUAL TO ZERO.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0
```

```
      SUBROUTINE RIDB3I(N1,N2,N3,IDL,IELL,IRAY,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RIDB3I WILL READ ANY INTEGER ARRAY WITH UP TO THREE
C       DIMENSIONS FROM A MOPADS DATA BASE, REAL, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO READ MSAINT ARRAYS FROM THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       WRITTEN TO THE DATA BASE WITH SUBROUTINE
C       WIDB3I.
C--INPUT PARAMETERS:
C       N1,N2,N3-DIMENSIONS OF IRAY. SEE BELOW.
C--INPUT/OUTPUT PARAMETERS:
C       IDL(2)-DBAA OF THE DATA LIST TO READ FROM
C       IELL-ON INPUT, THE ELEMENT OF THE DATA LIST WHICH
C            CONTAINS THE FIRST ELEMENT OF IRAY. ON RETURN,
C            THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C            ELEMENT OF IRAY.
C--OUTPUT PARAMETERS:
C       IRAY(N1,N2,N3)-THE INTEGER ARRAY TO READ. IF THE ARRAY
C                      HAS FEWER THAN 3 DIMENSIONS, SEND
C                      N2 AND/OR N3 EQUAL TO ZERO.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0
```

```
      SUBROUTINE RRDB3I(N1,N2,N3.IDL,IELL,XRAY,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RRDB3I WILL READ ANY REAL ARRAY WITH UP TO THREE
C       DIMENSIONS FROM A MOPADS DATA BASE, REAL, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO READ MSAINT ARRAYS FROM THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       WRITTEN TO THE DATA BASE WITH SUBROUTINE
C       WRDB3I
C--INPUT PARAMETERS:
C       N1,N2,N3-DIMENSIONS OF XRAY. SEE BELOW.
C--INPUT/OUTPUT PARAMETERS:
C       IDL(2)-DBAA OF THE DATA LIST TO WRITE TO.
C       IELL-ON INPUT, THE ELEMENT OF THE DATA LIST TO
C            READ INTO THE FIRST ELEMENT OF XRAY. ON RETURN,
C            THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C            ELEMENT OF XRAY.
C       XRAY(N1,N2,N3)-THE REAL ARRAY TO READ. IF THE ARRAY
C                      HAS FEWER THAN 3 DIMENSIONS, SEND
C                      N2 AND/OR N3 EQUAL TO ZERO.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0
```

```
      SUBROUTINE RSRN3I(NRN,IDRUN,IDSTAT,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RSRN3I WILL READ THE RUN DATA STATISITICS INTO SAINT ARRAYS
C--INPUT PARAMETERS:
C       NRN-RUN NUMBER
C--INPUT/OUTPUT PARAMETERS:
C       IDRUN(4)-DRAA OF THE RUN-NRN DR
C       IDSTAT(4)-DRAA OF THE STATISTICS DR
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO DATA BASE ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0 OR CORRECT DB DATA NOT FOUND
```

```
------------------------------------------------------------------------

      SUBROUTINE RSTS3I(ITS,IDBTS,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RSTS3I WILL READ THE MSAINT-DATA AND LABELS DL'S FROM
C       A CODE 5 DIRECTORY INTO MSAINT ARRAYS
C
C--INPUT PARAMETERS:
C       ITS-TACTICAL SCENARIO NUMBER
C--INPUT/OUTPUT PARAMETERS:
C       IDBTS(4)-DRAA OF THE STATISTICS(CODE 5) DR
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0 OR CORRECT DATA BASE DATA NOT FOUND

------------------------------------------------------------------------

      SUBROUTINE SHOW3I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       SHOW3I WILL PERFORM THE SHOW COMMAND IN THE "EXAMINE
C       STATISTICS" SUBPROCESS
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO THE COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR CODE

------------------------------------------------------------------------

      SUBROUTINE TOGL3I
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       TOGL3I WILL QUERY THE USER FOR THE DISPLAY DEVICE.
C       ACCEPTABLE RESPONSES ARE T-TERMINAL, P-PRINTER, AND
C       B-BOTH. IF THREE ERRORS ARE MADE IN SUCCESSION,
C       THE DISPLAY OPTION IS SET TO T.
```

```
      SUBROUTINE USE3I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       USE3I WILL PERFORM THE USE COMMAND IN THE 'EXAMINE STATISTICS"
C       SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR CODE
```

```
      SUBROUTINE WCDB3I(LRAY,N1,N2,N3,IDL,IELL,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       WCDB3I WILL WRITE ANY HOLLERITH ARRAY WITH UP TO THREE
C       DIMENSIONS TO A MOPADS DATA BASE, CHARACTER, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO WRITE MSAINT ARRAYS TO THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       RETRIEVED FROM THE DATA BASE WITH SUBROUTINE
C       RCDB3I.
C
C       ----NOTE----  HOLLERITH IS NOT A STANDARD FORTRAN 77 DATA
C                     TYPE. THIS PROGRAM IS INCUDED FOR
C                     COMPATIBILITY WITH MSAINT, WHICH USES
C                     HOLLERITH FOR LABELS.
C--INPUT PARAMETERS:
C       LRAY(N1,N2,N3)-THE HOLLERITH ARRAY TO WRITE. IF THE ARRAY
C                      HAS FEWER THAN 3 DIMENSIONS, SEND
C                      N2 AND/OR N3 EQUAL TO ZERO.
C--INPUT/OUTPUT PARAMETERS:
C       IDL(2)-DBAA OF THE DATA LIST TO WRITE TO.
C       IELL-ON INPUT, THE ELEMENT OF THE DATA LIST TO
C            RECEIVE THE FIRST ELEMENT OF LRAY. ON RETURN,
C            THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C            ELEMENT OF LRAY.
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE WIDB3I(IRAY,N1,N2,N3,IDL,IELL,IERR,*)
C--MODULE: MOPADS/U1
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       WIDB3I WILL WRITE ANY INTEGER ARRAY WITH UP TO THREE
C       DIMENSIONS TO A MOPADS DATA BASE, REAL, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO WRITE MSAINT ARRAYS TO THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       RETRIEVED FROM THE DATA BASE WITH SUBROUTINE
C       RIDB3I.
C--INPUT PARAMETERS:
C       IRAY(N1,N2,N3)-THE INTEGER ARRAY TO WRITE. IF THE ARRAY
C                      HAS FEWER THAN 3 DIMENSIONS, SEND
C                      N2 AND/OR N3 EQUAL TO ZERO.
C--INPUT/OUTPUT PARAMETERS:
C       IDL(2)-DBAA OF THE DATA LIST TO WRITE TO.
C       IELL-ON INPUT, THE ELEMENT OF THE DATA LIST TO
C            RECEIVE THE FIRST ELEMENT OF IRAY. ON RETURN,
C            THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C            ELEMENT OF IRAY.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE WRDB3I(XRAY,N1,N2,N3,IDL,IELL,IERR,*)
C--MODULE: MOPADS/U1
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       WRDB3I WILL WRITE ANY REAL ARRAY WITH UP TO THREE
C       DIMENSIONS TO A MOPADS DATA BASE, REAL, ONE DIMENSIONAL
C       DATA LIST. IT IS USED TO WRITE MSAINT ARRAYS TO THE
C       DB FOR STATISTICS ANALYSIS. THE ARRAY MUST BE
C       RETRIEVED FROM THE DATA BASE WITH SUBROUTINE
C       RRDB3I
```

```
C--INPUT PARAMETERS:
C      XRAY(N1,N2,N3)-THE REAL ARRAY TO WRITE. IF THE ARRAY
C                     HAS FEWER THAN 3 DIMENSIONS, SEND
C                     N2 AND/OR N3 EQUAL TO ZERO.
C--INPUT/OUTPUT PARAMETERS:
C      IDL(2)-DBAA OF THE DATA LIST TO WRITE TO.
C      IELL-ON INPUT, THE ELEMENT OF THE DATA LIST TO
C           RECEIVE THE FIRST ELEMENT OF XRAY. ON RETURN,
C           THE ELEMENT OF THE DATA LIST FOLLOWING THE LAST
C           ELEMENT OF XRAY.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C     1-IERR.NE.0
```

6-0 4I - CREATE/EDIT AIR SCENARIO

The programs in the Create/Edit Air Scenario subprocesses which end with the suffix "4I" are in this section.

```
      SUBROUTINE UI4I(JOPT)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       UI4I IS THE SINGLE ENTRY POINT FOR THE "CREATE/EDIT SCENARIO
C       DATA" SUBPROCESS.
C
C--INPUT/OUTPUT PARAMETERS:
C       JOPT-ON INPUT IT IS 4. ON OUTPUT IT IS ZERO IF A TERMINATE
C            COMMAND HAS BEEN ISSUED.
```

```
      SUBROUTINE ADAR4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADAR4I WILL PERFORM THE ADD-AIR COMMAND IN THE "CREATE/EDIT
C       SCENARIO DATA" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

```
      SUBROUTINE ADAS4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADAS4I WILL PERFORM THE ADD-ASSET COMMAND IN THE "CREATE/EDIT
C       SCENARIO DATA" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0- NO ERROR
C            .NE.0-DBCS ERROR CODE
```

```
      SUBROUTINE DELE4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       DELE4I WILL PERFORM THE DELETE COMMAND IN THE "CREATE/EDIT
C       SCENARIO DATA" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
```

```
      SUBROUTINE FAR4I(NDB,LABEL,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FAR4I WILL LOCATE A PARTICULAR AIR SCENARIO IN THE CURRENT
C       CRITICAL ASSET CONFIGURATION DIRECTORY OR RETURN AN
C       INDICATION THAT IS IS NOT PRESENT.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING, 2-REFERENCE)
C       LABEL-(CHARACTER) LABEL OF THE AIR SCENARIO TO FIND
C--OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE FOUND AIR SCENARIO. ZERO IF NOT FOUND.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-AIR SCENARIO "LABEL" NOT FOUND
C       2-IERR.NE.0
```

```
      SUBROUTINE FAS4I(NDB,ID,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FAS4I WILL LOCATE A PARTICULAR ASSET CONFIG. IN THE
C       SCENARIOS DIRECTORY OR RETURN AN
C       INDICATION THAT IS IS NOT PRESENT.
```

```
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING, 2-REFERENCE)
C       ID-SECOND WORD OF THE ASSET ID TO FIND
C--OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE FOUND ASSET CONFIG. ZERO IF NOT FOUND.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-ASSET CONFIG."ID" NOT FOUND
C       2-IERR.NE.0

-----------------------------------------------------------------------

      SUBROUTINE FTR4I(NDB,ITYP,IDBAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       FTR4I WILL CHECK FOR THE PRESENCE OF A PARTICULAR TRACK TYPE
C       (HOSTILE,FRIENDLY,OR OTHER) DL IN THE CURRENT AIR SCENARIO
C       DIRECTORY.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       ITYP-TYPE OF DL TO LOOK FOR
C            1-HOSTILE
C            2-FRIENDLY
C            3-OTHER
C--OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE FOUND DL
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-TRACK DL OF TYPE ITYP NOT FOUND
C       2-IERR.NE.0

-----------------------------------------------------------------------

      SUBROUTINE GET4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       GET4I WILL PERFORM THE GET COMMAND IN THE "CREATE/EDIT
C       AIR SCENARIO" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

------------------------------------------------------------------------

      SUBROUTINE RENA4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RENA4I WILL PERFORM THE RENAME COMMAND IN THE "CREATE/EDIT
C       AIR SCENARIO" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR

------------------------------------------------------------------------

      SUBROUTINE RFSC4I(NDB,IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RFSC4I WILL LOCATE THE "SCENARIOS" DIRECTORY AND PUT ITS
C       IDRAA IN IDRF5I FOR THE SPECIFIED DB.
C       IT WILL ALSO OPTIONALLY MAKE THE "SCENARIOS"
C       DIRECTORY CURRENT.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IOPT-OPTION
C            1-DO NOT MAKE THE SCENARIOS DR CURRENT
C            2-DO MAKE THE SCENARIOS DR CURRENT
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .GT.0-DB ERROR CODE
C            .LT.0-CANNOT LOCATE "SCENARIOS" DIRECTORY
```

```
      SUBROUTINE SAVE4I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       SAVE4I WILL PERFORM THE SAVE COMMAND IN THE "CREATE/EDIT
C       AIR SCENARIO" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

7-0 5I - CREATE/EDIT REFERENCE SYSTEM MODULE

The programs in the Create/Edit Reference System Module sub-processes which end with the suffix "5I" are in this section.

```
      SUBROUTINE UI5I(JOPT)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       UI5I IS THE ENTRY POINT TO THE "CREATE/EDIT REFERENCE SYSTEM
C       MODULE" SUBPROCESS
C
C--INPUT/OUTPUT PARAMETERS:
C       JOPT- ON INPUT IT IS 5. IF ON OUTPUT IT IS ZERO, A TERMINATE
C             COMMAND WILL BE EXECUTED.
```

```
      SUBROUTINE ADD5I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       ADD5I WILL PERFORM AN 'ADD' COMMAND IN THE "CREATE/EDIT REFERENCE
C       SYSTEM MODULE" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-PERFORM COMMAND
C            1-PERFORM HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

```
      SUBROUTINE CHAN5I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHAN5I WILL PERFORM A CHANGE COMMAND IN THE CREATE/EDIT
C       REFERENCE SYSTEM MODULE SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-PERFORM COMMAND
C            1-PERFORM HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
```

```
      SUBROUTINE CHEN5I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHEN5I DOES EDITING OF THE ENVIRONMENT STATE VECTOR OF
C       THE CURRENT ADSM. ONLY THE WORKING DB MAY BE EDITED.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-COMMAND CANCELLED
C       2-IERR.NE.0
```

```
      SUBROUTINE CHOP5I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHOP5I ALLOWS THE USER TO EDIT OPERATOR STATE VECTORS FOR
C       THE CURRENT ADSM. ONLY THE WORKING DB MAY BE EDITED.
C
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C              .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-COMMAND CANCELLED
C       2-IERR.NE.0
```

```
      SUBROUTINE CHR5I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHR5I ALLOWS THE USER TO EDIT THE RESOURCE AND RESOURCE LABEL
C       DL'S FOR THE CURRENT ADSM. ONLY THE WORKING DB MAY BE EDITED.
C
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-COMMAND CANCELLED
C       2-IERR.NE.0
```

```
      SUBRCUTINE CHTD5I(IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CHTD5I ALLOWS THE USER TO EDIT THE TASK DATA LIST OF A CURRENT
C       ADSM. ONLY THE WORKING DB MAY BE EDITED.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       1-COMMAND CANCELLED
C       2-IERR .NE.0
```

```
      SUBROUTINE CRES5I(NACC,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRES5I WILL CREATE THE ENVIRONMENTAL STATE VECTOR(EN) DATA
C       LIST FOR A NEW REFERENCE ADSM.  DEFAULT VALUES ARE USED.
C--INPUT PARAMETERS:
C       NACC-THE VALUE OF NECC(ACC). ACC IS THE ADSM CHARACTER CODE
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE ADSM
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR
C--ALTERNATE RETURNS:
C       0-NO ERROR
C       1-DB ERROR, IERR.NE.0
C       2-CANCEL COMMAND
```

```
      SUBROUTINE CROP5I(NACC,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CROP5I CREATES OPERATOR STATE VECTORS(OP) AND AN OPERATOR TYPE
C       (OT) DL IN A NEW ADSM.  DEFAULT VALUES ARE LOADED FOR ALL
C       VARIABLES.
C
```

```
C--INPUT PARAMETERS:
C       NACC-THE VALUE OF NECC(ACC). ACC IS THE ADSM CODE CHARACTER.
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE ADSM
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DB ERROR
C--ALTERNATE RETURNS:
C       0-NO ERROR
C       1-DB ERROR,IERR.NE.0
C       2-CANCEL ISSUED

------------------------------------------------------------------------

      SUBROUTINE CRRS5I(NACC,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRRS5I CREATES THE RESOURCE(R) DATA LIST AND THE RESOURCE LABEL
C       (RL) DATA LIST ON NEW REFERENCE ADSM'S.
C--INPUT PARAMETERS:
C       NACC-THE VALUE OF NECC(ACC). ACC IS THE ADSM CHARACTER CODE
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-THE DRAA OF THE ADSM
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       0-NO ERROR
C       1-DB ERROR, IERR.NE.0
C       2-CANCEL ISSUED

------------------------------------------------------------------------

      SUBROUTINE CRTD5I(NACC,IDRAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       CRTD5I CREATES A TASK DATA(TD) LIST IN A NEW REFERENCE ADSM.
C--INPUT PARAMETERS:
C       NACC-THE VALUE OF NECC(ACC). ACC IS THE ADSM CHARACTER CODE
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE ADSM
```

```
      SUBROUTINE GTGL5I(IGL,NEXT,IDBAA,IERR,*,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       GTGL5I WILL QUERY FOR GOAL PARAMETERS AND LOAD THEM INTO AN
C       OPERATOR STATE VECTOR
C
C--INPUT PARAMETERS:
C       IGL-THE GOAL NUMBER TO GET PARAMETER VALUES FOR
C       NEXT-1ST ELEMENT OF THE OSV FOR GOAL IGL.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE OSV
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR
C--ALTERNATE RETURNS:
C       0-NO ERROR
C       1-DB ERROR, IERR.NE.0
C       2-CANCEL TYPED
```

```
      SUBROUTINE OPCL5I(IOPT,IPNT,NDEL,NLEN,X)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       OPCL5I WILL OPEN OR CLOSE SPACE IN AN ARRAY. FOR EXAMPLE,
C       TO OPEN NDEL ELEMENTS AT LOCATION IPNT, OPCL5I WILL SHIFT
C       (END OFF!) THE CONTENTS OF THE ARRAY X DOWN NDEL ELEMENTS
C       STARTING AT IPNT AND STORE ZERO IN THE ELEMENTS IPNT TO
C       IPNT+NDEL-1. SIMILARLY, TO CLOSE NDEL ELEMENTS STARTING AT
C       IPNT, OPCL5I WILL OVERWRITE ELEMENTS IPNT TO IPNT+NDEL-1
C       WITH THE ELEMENTS IPNT+NDEL TO IPNT+NDEL+NDEL-1. ALSO ALL
C       SUBSEQUENT WORDS WILL BE SHIFTED AND THE ARRAY WILL BE
C       ZERO FILLED FROM THE END.
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            1-OPEN
C            2-CLOSE
C       IPNT-ELEMENT OF X AT WHICH TO START THE SHIFT
C       NDEL-NUMBER OF ELEMENTS TO SHIFT
C       NLEN-LENGTH OF X
C--INPUT/OUTPUT PARAMETERS:
C       X(NLEN)-ARRAY TO SHIFT
```

```
      SUBROUTINE READ5I(NVALS,VALS)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       READ5I WILL READ A LINE OF NVALS VALUES TO THE ARRAY VALS.
C       IF A "*" IS SPECIFIED FOR ANY VALUE, THE CORRESPONDING
C       ELEMENT OF VALS WILL NOT BE CHANGED.
C
C--INPUT PARAMETERS:
C       NVALS-THE NUMBER OF VALUES TO READ. ALL MUST BE ON ONE LINE.
C             IF FEWER THAN NVALS FIELDS ARE ON THE LINE, THE REMAINING
C             ELEMENTS OF VALS WILL BE UNCHANGED. IF NVALS=0,RETURN WITH
C             NO ACTION
C--OUTPUT PARAMETERS:
C       VALS(NVALS)-OUTPUT ARRAY
```

```
      SUBROUTINE RENA5I(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RENA5I WILL PERFORM THE RENAME COMMAND IN THE "CREATE/EDIT
C       REFERENCE SYSTEM MODULE" SUBPROCESS.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
```

```
      SUBROUTINE RFAD5I(NDB,IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       RFAD5I WILL LOCATE THE "REFERENCE ADSM" DIRECTORY AND PUT ITS
C       IDRAA IN ID'F5I FOR THE SPECIFIED DB.
C       IT WILL ALSO OPTIONALLY MAKE THE "REFERENCE-ADSM"
C       DIRECTORY CURRENT.
```

```
C--INPUT PARAMETERS:
C       MDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IOPT-OPTION
C            1-DO NOT MAKE THE REF ADSM DR CURRENT
C            2-DO MAKE THE REF-ADSM DR CURRENT
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .GT.0-DB ERROR CODE
C            .LT.0-CANNOT LOCATE "REFERENCE-ADSM" DIRECTORY

-------------------------------------------------------------------------

      SUBROUTINE SAVESI(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       SAVESI WILL PERFORM THE SAVE COMMAND IN THE "CREATE/EDIT
C       REFERENCE SYSTEM MODULE" SUBPROCESS.
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE

-------------------------------------------------------------------------

      SUBROUTINE USESI(IOPT,IERR)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C       USESI WILL PERFORM THE USE COMMAND IN THE "CREATE/EDIT
C       REFERENCE SYSTEM MODULE" SUBPROCESS
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-DO COMMAND
C            1-DO HELP COMMAND
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
```

VI. USER INSTRUCTIONS

Each of the five subprocesses has a single entry point program named UI?I where the "?" is an integer from one to five. This subprogram contains the FFSP data structure for the commands of that subprocess. Each command is processed by a separate subprogram. The command programs are called from the entry point program.

It is, therefore, easy to understand and trace the structure of each subprocess by starting at the entry point program for the subprocess. Various utility programs are included in each subprocess whose purpose, and uses can be discussed from their context and internal documentation.

VII. ERROR PROCESSING

1-0 THE USER INTERFACE

The user interface is an interactive processor. It never terminates execution of the MOPADS program. Data entry errors are trapped by the program. An explanatory message is printed, and the user is given the opportunity to re-enter the data. Alternatively, the program may cancel the command being processed and return to the command mode. The user may then issue the command again.

Execution will be terminated, however, if a fatal data base error occurs. Such errors should not occur, of course, but non-recoverable errors in the DBCS will cause the program to be terminated. Little damage to the data base should result, because the DBCS operates in the "safe" mode for the user interface, see Polito 1983(b). Any such error should be diagnosed by a MOPADS modeler, however.

2-0 THE MAIN MODULE

The primary function of the main module is to read MOPADS data cards and direct control to the user interface or to the MSAINT module. Therefore, the main module operates as though it were in a batch environment. Any errors found in the data cards or in performing initial program set-up will cause execution to terminate. Error processing uses subroutine ERRORA from the DBAP. Also, some error messages are written with simple write statements from the Main module.

VIII. COMMON VARIABLE DEFINTIONS

1-0 THE USER INTERFACE

COMMON variables for the user interface are shown in the following forms.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
MMSYSI	Maximum number of system module types	4	PARAMETER
MXNAD	Maximum row dimension of NADSMI	12	PARAMETER
ADLB5I*40	Label of the directory whose DRAA is in column 1 of IDRF5I	blank	COM2I
CHRSPI(2)*1	Character response universe array for HELP command	blank	COM2I
CMDØI(11)*10	Command names of the low level data base commands	'CLOSE' 'CURRENT' 'DEPOSIT' 'DIRECTORY' 'EXAMINE' 'HELP' 'OPEN' 'SELECT' 'TERMINATE' 'TLINK' 'MENU'	
HPRMI(1)*10	Prompt for the HELP command	'COMMAND'	COM2I

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: User Interface
PROJECT: MOPADS

SAINT VARIABLES

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
ICURI(4,2,2)	Current DB elements ICURI(1:4,1,ND) is the DRAA of the current directory on data base ND ICURI(1:2,2,ND) is the DBAA of the current data list on data base ND	-	COM1I
IDRF1I(4,3)	In the 1I subprocess: IDRF1I(1:4,1) DRAA of the simulation data set DR IDRF1I(1:4,2) DRAA of the command and control DR IDRF1I(1:2,3) DBAA of the copy counter DL	-	COM1I

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: User Interface
PROJECT: MOPADS

SAINT VARIABLES Page 2 of 5

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
IDRF5I(4,3)	In the 3I subprocess: IDRF1I(1:4,1) DRAA of the simulation data set DR IDRF1I(1:4,2) DRAA of the tactical scenario DR IDRF1I(1:4,3) DRAA of the RUN-n DR In the 1I subprocess: IDRF5I(1:4,ND) DRAA of the Master-Diretory for data base ND In the 3I subprocess: IDRF5I(1:4,2) DRAA of the statistics DR In the 4I subprocess: IDRF5I(1:4,ND) DRAA of the scenarios DR for data base ND	-	COM1I

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: User Interface
PROJECT: MOPADS

SAINT VARIABLES

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
	In the 5I subprocess: IDRF5I(1:4,ND) DRAA of the REFERENCE ADSM DR for data base ND		
IST3I(4)	IST3I(1) - Simulation data set number (2) - Tactical scenario number (3) - run number (zero for summary) (4) - display option 1 - terminal 2 - printer 3 - both	(0,0,0,0)	COM1I
JINI	MOPADS interactive input file	-	COM1I
JOUTI	MOPADS interactive output file	-	COM1I
KHPMTI(1,2)	pointer array for HELP command	(1,0)	COM1I
LHTAXI(5,1)	syntax array for HELP command	3001 5 1 0 0	COM1I

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: User Interface
PROJECT: MOPADS

SAINT VARIABLES Page 4 of 5

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
LINE3I*130	buffer for one line of output	blank	COM2I
MAIN1I(4)	DRAA of Master-Directory in 1I, 2I, and 3I subprocesses	-	COM1I
MCMDØI	Length of CMDØI array	11	COM1I
NADSMI (MXNAD, MMSYSI)	Each column is for a different system module type Row 1 - system module type 2 - NECC(ACC) 3-12 - operator types for the system module	-	COM1I
NDEXØI(11)	Cross reference index for binary search of CMDØI	zero	COM1I
OPLBLI(9)*10	labels of each operator type	-	COM2I

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: User Interface
PROJECT: MOPADS

SAINT VARIABLES Page 5 of 5

2-0 THE MAIN MODULE

COMMON variables for the main module are shown in the following forms.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
MMSYSM	Maximum system module number	4	PARAMETER
EOPM(MMSYSM) *1	The echo option for network data cards of each system module type N - no echo P - partial echo F - full echo	-	COM2M
IDBCSM(4,4)	IDBCSM(1:4,1) DRAA of the simulation data set DR IDBCSM(1:4,2) DRAA of the tactical scenario DR IDBCSM(1:4,3) DRAA of the statistics DR IDBCSM(1:4,4) DRAA of the command and control		COM1M
IAIRM	Air scenario number	-	COM1M
ICRITM	Critical asset configuration number	-	COM1M

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: Main Module
PROJECT: MOPADS

SAINT VARIABLES Page 1 of 2

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
ISDSM	Simulation data set number	-	COM1M
ITACM	Tactical scenario number	-	COM1M
KUNITM(15)	Unit numbers used by MOPADS	KUNITM(I)=I	COM1M
MOPKM	Maximum number of elements in any operators task stack	50	COM1M
NETFLM(MMSYSM) *40	file names of the network data files for the system modules	-	COM2M
VERSM - 60	MOPADS version message	See MOPADM	COM2M

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X_USER VARIABLES

NAME: Joe Polito
DATE: 29 November 1983

AIR DEFENSE SYSTEM MODULE: Main Module
PROJECT: MOPADS

SAINT VARIABLES

IX. REFERENCES

Goodin, J. R. & Polito, J. MOPADS free-format syntax processor (MOPADS/FFSP) (MOPADS Vol. 5.11). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(a).

Polito, J. The MOPADS data base control system (MOPADS/DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(b).

Polito, J. MOPADS free-format input programs (MOPADS/FFIN2) (MOPADS Vol. 5.14). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(c).

Polito, J. The MOPADS data base application programs (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983(d).

Polito, J. MOPADS architecture manual (MOPADS Vol. 4.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, Tx, 1983(e).

Polito, J. & Goodin, J. R. MOPADS utility programs (MOPADS/UTIL) (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

X. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

XI. CHANGE NOTICES

APPENDIX U

MOPADS FINAL REPORT:

THE MOPADS DATA BASE CONTROL SYSTEM (MOPADS/DBCS)

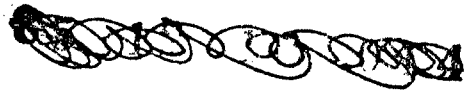


TABLE OF CONTENTS

TABLE OF CONTENTS

(continued)

TABLE OF CONTENTS

(continued)

LIST OF FIGURES

LIST OF TABLES

TERMINOLOGY

1-0 STANDARD MOPADS TERMINOLOGY.

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element time, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.
ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.

ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT(MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.
OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.

OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).
TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.

TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

2-0 OTHER TERMINOLOGY.

CDL	Character data list.
CONTROL PART	The first 10 words of each record. The control part contains information used internally by the DBCS.
DATA BASE	The collection of user information and control information that is processed by the DBCS.
DATA BASE FILE	A computer file stored on disk that contains the data base.
DATA LIST	A list of values (character or numeric) that is user specified and is stored in a data base.

DATA PART	That part of a record that stores user specified data. Records are composed of a control part and a data part.
DB	Data Base.
DBA	Data Base Address. The record number of a record on disk.
DBAA	Data Base Access Address. The DBAA is a two word address for data lists consisting of a DBA and a DBCSA.
DBCS	Data Base Control System.
DBCSA	Data Base Control System Address. An address in-core where a record is stored.
DBF	Data Base File.
DIRECTORY	A collection of data lists and other directories.
DL	Data List.
DR	Directory.
DRAA	Directory Access Address. The DRAA is a four word address for directories consisting of a DBA, DBCSA, and the current directory and data list positions of the directory.
ID	Identify. DL's and DR's have identifiers that are lists of integers and are stored in their owning directory.

IDL	Integer Data List.
INITIAL DATA BASE ADDRESS	The DBA of the first record of a data list or directory.
INTERNAL STORE	Record storage used temporarily for working storage by the DBCS.
IS	Internal Store.
LDL	Label Data List.
RDL	Real Data List.
SAF	Smoothed Access Frequency.
TEMPORARY STORE	Record storage for records that are immediately available to be "bumped" out of core by incoming records.
TS	Temporary Store.
WORKING STORE	Record storage for records that have semi-permanent in-core residence because of the value of their SAF's.
WS	Working Store.

I. PURPOSE

This document describes the data management system for the software which implements the Models of Operator Performance in Air Defense Systems (MOPADS) programs. The MOPADS software must maintain a great deal of information about a variety of subjects:

1. Data which describes the Air Scenario to be simulated,

2. Data which describes the tactical scenario which includes location and characteristics of air defense units, the command and control system, and the coordinate reference system,

3. Data which describes the characteristics of the operators of air defense systems and the environment in which they function,

4. Data which describes the dynamic relationships of operator tasks, and

5. Data which represents statistics collected during simulations.

All of this information must be maintained in an easily accessed and edited form, and it must be addressed in a way that permits multiple data sets to co-exist without confusion.

The most effective way to accomplish this is with a unified data management system or a data base. This document describes the Data Base Control System (DBCS) module of the MOPADS software. The DBCS is described in general terms as a data management system. The structure of the MOPADS specific data base used by the MOPADS software is described in detail elsewhere (Polito (1983a)).

The DBCS is a non-traditional data base manager. However, the organization of the MOPADS data base file most nearly resembles a hierarchical data base. It is non-traditional in the sense that rapid access of datum elements is needed during MOPADS simulations. The DBCS utilizes a data address that is passed into and out of the DBCS which permits rapid access of required data. The address precludes most hierarchical searches in the data base file to locate data; thus it eliminates many disk accesses. Furthermore, the DBCS dynamically retains the most frequently accessed data (not merely the most recently accessed) in core, so disk reads are minimized. These features make the DBCS a reasonably fast tool for storage and retrieval of data.

The DBCS module is a collection of FORTRAN 77 subprograms which may be called by user programs to manipulate a DBCS data base file. It has extensive error trapping features, and several protection modes from read-only to a risky, but fast, mode that exposes the data base to damage if an error occurs. This flexibility permits the user programs to tailor the DBCS performance to their requirements.

II. DATA STRUCTURE DESCRIPTIONS

1-0 DBCS DATA BASES.

1-1. Directories and Data Lists.

Directories and data lists (DR's and DL's) are the two organizational structures that comprise a DBCS data base. A data list is exactly what its name implies; it is an ordered list of data. DL's are physically where information is stored in the data base. Data lists may be designated as being of type real, integer, or character. Abbreviations for each type are RDL (real), IDL (integer), and CDL (character). RDL's and IDL's may be one or two dimensional. CDL's may have only one dimension.

Every data list may have a label which gives its name, and every element of a data list may have a label. DBCS labels are maintained in special label data lists (LDL's). LDL's are maintained internally by the DBCS, and labels may have only 25 characters. As an example, a character data list may have an element (say the seventh) whose label is "NAME" and whose value is "JOHN DOE." It is possible to access "JOHN DOE" by requesting "NAME," but this is slow. A faster method is to remember that name is the seventh element and request it directly. The DBCS has facilities for addressing data both ways.

Furthermore, each user created directory and data list has an identifier (ID) which is a list of integer numbers. The length of the list is application dependent. The ID's provide another way to locate and organize data in the data base.

Directories are collections of data lists and other directories. Directories are said to "own" DL's and other directories which belong to it. Thus a hierarchy of directories is formed in which each directory (with the exception of certain system directories which are described later) is owned by another directory and which may own other directories. This hierarchy is a tree since no directory is owned by more than one directory. (Note: it is technically possible to create directories which have no owner-i.e. disjoint trees, and disjoint cycles of directories, but this practice is not recommended since it may be difficult to "find" these disjoint structures in the data base.)

Figure II-1 is a schematic of a DBCS data base. The notation shown in the legend is used throughout to designate directories and data lists in data base schematics. This notation points out that directories may have labels, also. Every DBCS data base (DB)

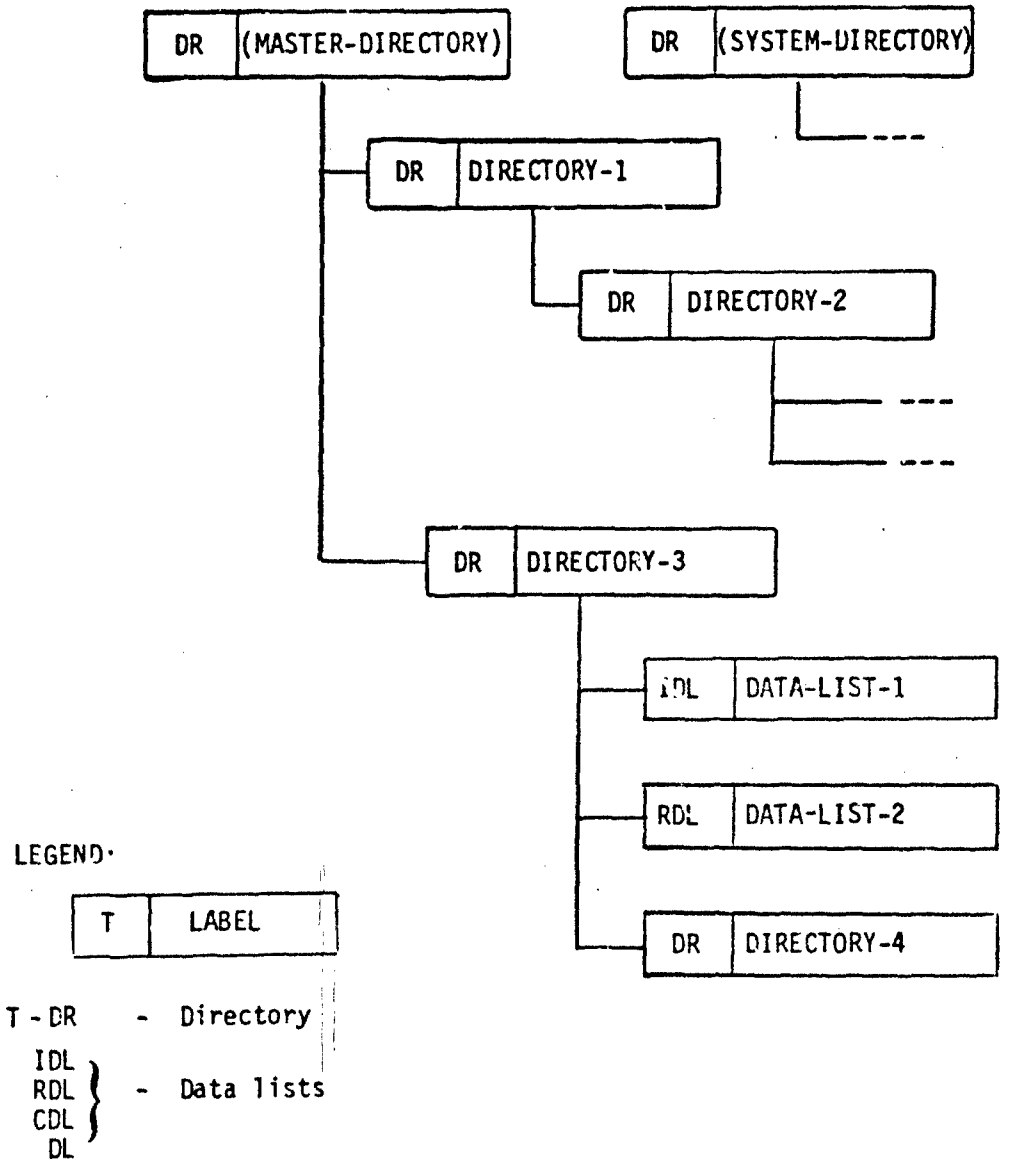


Figure II-1. Schematic of a DBCS Data Base.

is created with a master directory whose label is "(MASTER-DIRECTORY)." In addition, a disjoint directory labeled "(SYSTEM-DIRECTORY)" is also created. The system directory is used internally by the DBCS and is never directly accessed by application programs.

An application program has created four additional DR's in Figure II-1. DIRECTORY-1 and DIRECTORY-3 are owned by (MASTER-DIRECTORY). DIRECTORY-1 owns DIRECTORY-2 and DIRECTORY-3 owns DIRECTORY-4 and two data lists labeled DATA-LIST-1 and DATA-LIST-2. (NOTE: Labels may contain any character including blanks; by convention we will use "-" instead of blanks to separate words, however.)

There is no limit to how many DL's and DR's a directory may own. Moreover, DL's and DR's may be created and deleted at will, and DL's may be read, written, extended, and shortened at any time.

1-2 The Structure of a Directory.

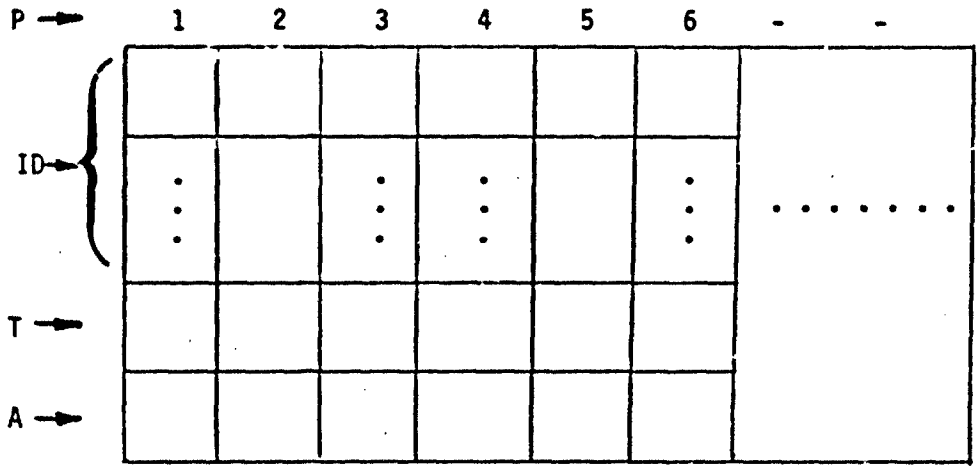
A directory may be considered as a list of DL and DR ID's as shown schematically in Figure II-2. Every DL or DR which belongs to a directory must have an ID of the same length, but this length may differ from directory to directory. In other words, it is a characteristic of the directory.

The element T identifies the types of entitity (DR, IDL, RDL, or CDL) for the column. The address element A, is used internally by the DBCS to physically locate information on the data base file. The physical column in the directory which holds an entity's ID is its "position." DL and DR columns will be intermixed in the directory.

A directory may have two ranking codes (one for DL's and one for DR's) which determine the order that DL and DR columns appear in the directory. Table II-1 shows how to assign ranking codes. For example, if the DL ranking code is 2, then the columns which represent DL's in the directory will appear in order of increasing values of ID element 2 (e.g., element 2 of column 4 would have a greater value than element 2 of column 2). Conversely, if the ranking code were -2, the columns would appear in decreasing order of ID element 2. No ordering is specified if the ranking code is zero. In case of ties, the ordering is not predictable.

A directory that is being accessed has a current "directory position" for DL's and DR's which is the most recently accessed directory and data list position. For example, if column 6 were the most recently accessed data list and column 22 were the most recently accessed directory, then the current data list position and the current directory position would be 6 and 22, respectively.

The DBCS automatically positions DL's and DR's within a directory. The only time an application program need be concerned with the directory position is when accessing data with instructions such as: "find



ID - identifiers of owned DL's and DR's

T - entity type (1-DR, 2-IDL, 3-RDL, 4-CDL)

A - internal address to the owned entity

P - the column number is the "position" of the entity

Figure II-2. Directory Structure.

the next data list in the forward (increasing directory position) direction," or "find the next directory whose ID element 4 is less than 6 in the backward direction." In these cases, it is necessary for the application program to remain aware of the current directory position.

Table II-1. Ranking Codes for Directories and Data Lists.

Ranking Codes for Owned Directories	
Code	Meaning
0	not ranked
n	increasing order of ID element n
-n	decreasing order of ID element n

Ranking Codes for Owned Data Lists	
Code	Meaning
0	not ranked
n	increasing order of ID element n
-n	decreasing order of ID element n

1-3. The Structure of a Data List.

Data lists are merely one or two dimensional lists of numbers or character strings. Character data lists (CDL's) may only be one dimensional, but the length of each element (in characters) is user selectable. In other words, one CDL may have 10 characters per element and another may have 40. Label data lists (LDL's) are special CDL's which are manipulated only by the DBCS and they always have 25 characters/label.

Real and integer data lists (RDL's and IDL's) may have one or two dimensions. Two dimensional DL's may be thought of in the usual row-column sense except in one important instance. The DBCS allows DL's to be extended (i.e., more elements added) and truncated (elements deleted). For one dimensional DL's, this produces no conceptual difficulties; elements are added and deleted from the end of the DL.

For two dimensional DL's, we must specify whether columns or rows are to be added or deleted. This gives rise to two types of two dimensional data lists: "column oriented" and "row oriented."

If a DL is column oriented, then the maximum row dimension is specified and columns may be added and deleted at will. If it is row oriented, then the maximum column dimension is specified and as many rows as desired may be added or deleted. Figure II-3 shows this distinction.

If the row and column dimensions of a two-dimensional DL will remain fixed, then the orientation should be chosen to maximize access efficiency. It is most efficient to access column oriented DL's by columns (or parts of columns) and row oriented DL's by rows (or parts thereof). For example, it is faster to read all of row 6 of a row oriented DL than row 6 of a column oriented DL.

2-0 DBCS DATA BASE FILES.

2-1. Structure of the Data Base File.

The DBCS data files are direct access, unformatted FORTRAN files. They are comprised of linked lists or records. This means that each record in such a file may have a predecessor and a successor record. The record numbers of the predecessor and successor records are stored in data fields of the record (if no predecessor or successor exists, then the appropriate field is zero). A record's Data Base Address (DBA) is its record number The DBA is the primary means of locating information on the DB file. The Initial Data Base address of a DL or DR is the DBA of the first record that makes up its record chain.

Data lists and directories are linked chains of records that contain sufficient storage to hold all of the information in the DL or DR. The DBCS performs all bookkeeping and record manipulation; thus the details of the data file implementation are transparent to the user.

2-2. Structure of Data Records.

Record one is a special record which contains various DB parameters. It has a special structure which is described subsequently. All other records have the structure described in this section.

Records are divided into two sections: the control part and the data part. The control part contains information that allows the DBCS to efficiently store and retrieve information from the DB. The data part holds information inserted by the user. There are ten words in the control part of each record (actually, there is a DBCS variable, NWCPD, that determines the control part size, but this is only to allow for possible DBCS modifications). All remaining words in a record belong to the data part. The total record size is under user control and may be selected to maximize the efficiency of disk storage hardware.

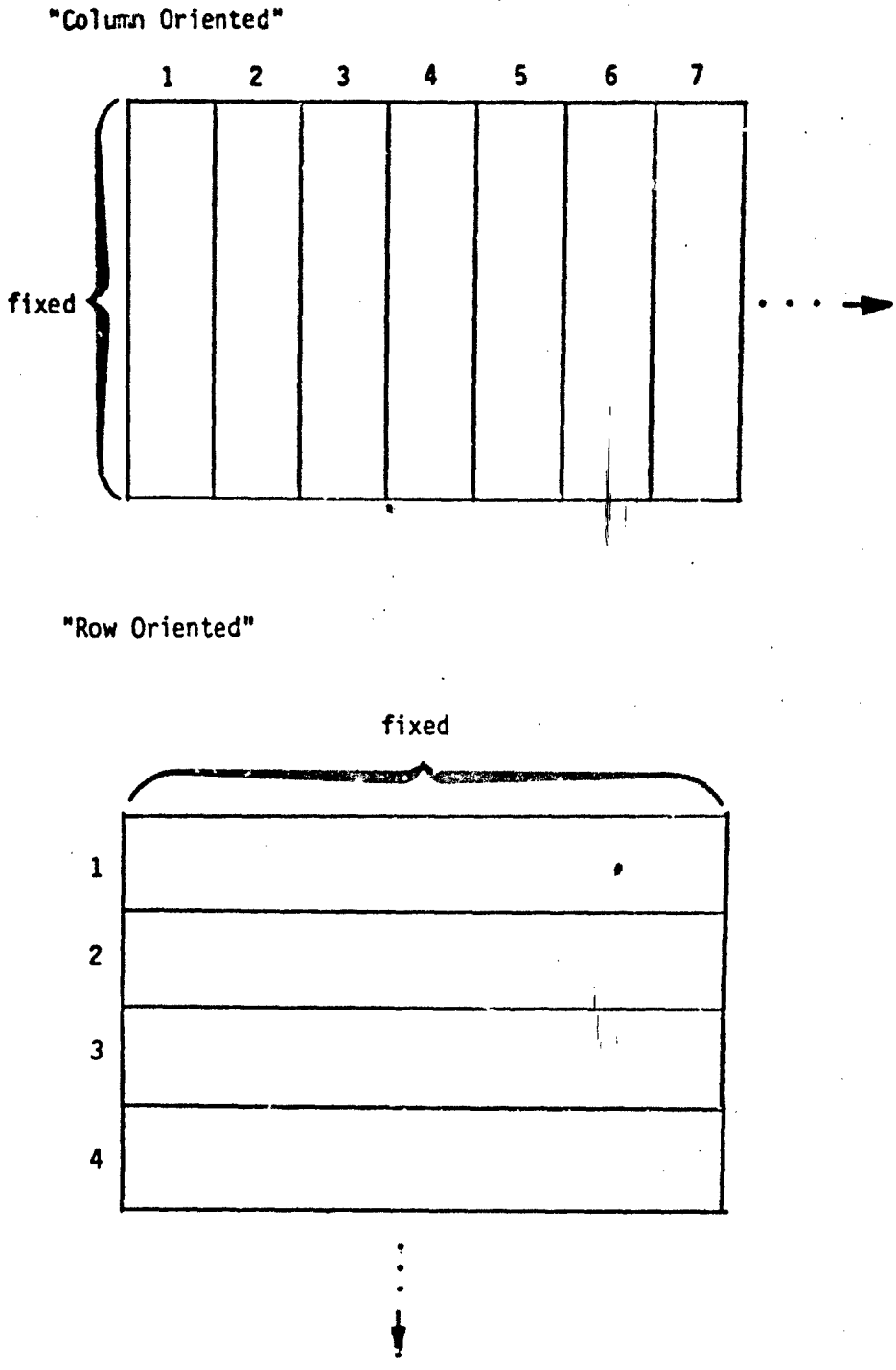


Figure II-3. Two Dimensional Data Lists.

The definitions of the various fields in the control part of records is shown in Table II-2.

Words W1 and W3 are the predecessor and successor DBA's, respectively. W2 is the record's own DBA. When a record is read into core storage, W2 is made negative if any of the data in the record is subsequently changed. This is an indication that the in-core version of the record has information that is not yet updated on the data base file. W4 is the Initial DBA of the DL or DR. Thus, it is always possible to go directly to the beginning of a record chain, and, more importantly, it is possible to determine if two records belong to the same record chain by comparing their W4 values.

If the record is a data list, then it has an LDL (which may be empty if no elements are labeled). The initial DBA of the LDL is given in W5. The label of the data list is the first element of its LDL. Then the second element of LDL is the label of the first element of the DL and so on. LDL's have W5=0 since they have no LDL.

If the record belongs to a directory, W5 contains DL and DR ranking information. The ranking codes, RDL and RDR, for data lists and directories have values as shown in Table II-1. Directories are actually two dimensional, column oriented IDL's whose maximum row dimension is the length of the ID's plus two (see Figure II-2) NWADD (See Section VIII) is a DBCS variable whose value is two (the number of extra rows in directories). The formula W5 = RDL*(W8-NW ADD) + RDR is an easily coded and decoded way to store the ranking codes in a single word. (See SUBROUTINE RKDIRD in Section V for details.)

W6 is the Initial DBA of the owner of the record. It is a directory for DR's, IDL's, RDL's, and CDL's (zero for no owner) and the owner DL for LDL's. W7 is the DB entity type.

W8 holds dimension information for this entity. For CDL's and LDL's, W8 is the number of characters per element. These DL's hold character data, and the length in characters of each element must be specified. W8 is always 25 for LDL's. This implies that DL element labels and DL labels may have a maximum of 25 characters. CDL's and LDL's are one dimensional lists.

RDL's and IDL's may be two dimensional and either row or column oriented. For one dimensional IDL's and RDL's, W8 = 1. For two dimensional column oriented DL's, W8 is the maximum row dimension. It is the negative of the maximum column dimension for row oriented DL's. For directories, W8 is the maximum row dimension since DR's are just special two dimensional IDL's.

Table II-2. The Control Part of Records.

Word	Mnemonic	Description
1	W1	Predecessor DBA (zero if none)
2	W2	DBA of this record
3	W3	Successor DBA (zero if none)
4	W4	Initial Data Base Address of the DL or DR
5	W5	For Data Lists: The Initial DBA of the LDL (zero if this is an LDL) For Directories: The combined ranking codes for DL's or DR's. RDL*(W8-NWADD)+RDR See discussion
6	W6	Initial DBA of the owner directory (zero if none). Exception: if this is an LDL, W6 is the Initial DBA of the owning DL.
7	W7	Entity Type (1-Directory, 2-IDL, 3-RDL, 4-CDL, 5-LDL)
8	W8	For Data Lists: a. For CDL's and LDL's: the number of characters per element b. For IDL's and RDL's: the maximum row or column dimensions If W8>1, column oriented If W8<0, row oriented If W8=1, one dimensional See discussion For Directories: The maximum row dimensions (directories are really column oriented IDL's). See discussion.
9	W9	Index of the first element of the DL or DR in this record
10	W10	Index of the last element of the DL or DR in this record

W9 and W10 are the indexes of the first and last elements in the data part of the record. Elements are numbered from one to infinity (even for two dimensional DL's). Thus, the first record of a DL might have elements one to fifty (W9=1, W10=50), and the second record could have elements 51 to 58 (W9=51, W10=58). A new record is added to the record chain when the data part of the current record is filled. The DBCS internally determines which elements belong to what row and column for two dimensional DL's and DR's.

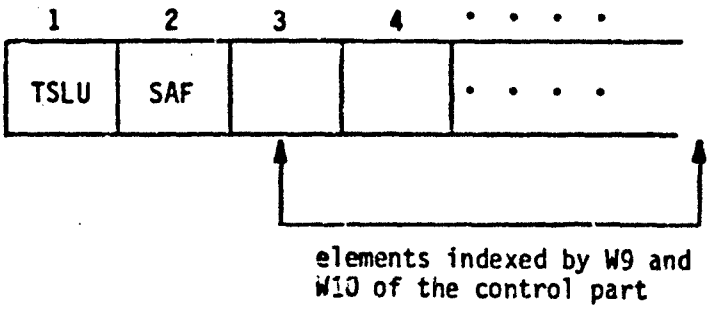
Undefined numeric elements are zero and character elements are blank. For example, if element 50 of an IDL is set to zero, but elements 1-49 were not yet set, then elements 1-49 exist and are equal to zero.

When a data list is created, an LDL is created for it which always contains at least one element (the DL label). If no label is specified for the DL, a default of "(UNLABELED)" is used. If the data list is then populated, but no element labels are specified, then the LDL still contains only one element. However, if the label of an unlabeled DL element is requested, the DBCS responds with "(UNLABELED)."

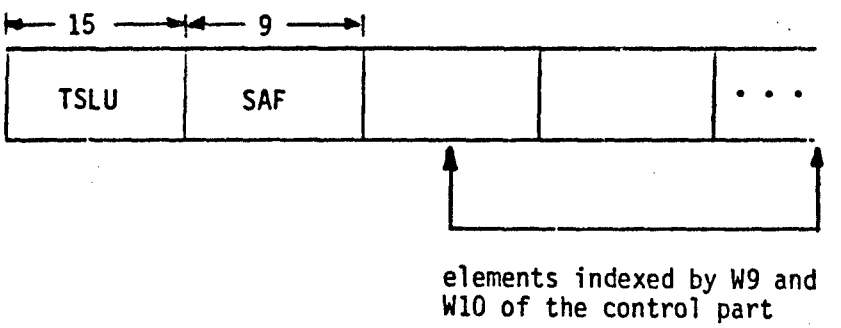
In addition to the user created information, the data parts of each record contain two numbers used internally by the DBCS. The DBCS retains frequently accessed records in fast core storage in order to reduce the number of disk accesses. In order to do this, it must keep track of the access frequency of each record. It needs two values to do this. The DBCS counts the cumulative number of accesses to the data base and computes a smoothed access frequency for each record. When a record is written to disk, the current access count and the current smoothed access frequency for the record are written with it. The mnemonics for these values are TSLU and SAF, respectively. These concepts are discussed more fully in Section 3-0 below.

Figure II-4 shows how these values are stored in the data part. There are two cases. For directories and numeric data lists, TSLU and SAF are stored in the first two words of the data part. The subsequent words in the data part are the data list elements indexed by W9 and W10 of the control part. For character data lists and label data lists, TSLU is written to the first 15 characters of the data part with a G15.7 FORTRAN format editing specification. SAF is written to the next 9 characters with an F9.6 specification. The remaining characters of the data part are partitioned into elements of length W8 and are indexed by W9 and W10. Any partial elements (of less than W8 characters) at the end of the data part are unused. In other words, character elements are not split between records.

1. Data Parts for Directories, Integer DL's and Real DL's



2. Data Parts for Character and Label Data Lists



TSLU is stored with a G15.7 specification.
SAF is stored with a F9.6 specification.

Figure II-4. Structure of the Data Parts of Records.

2-3. The Master Directory.

Every DB has a master directory whose DBA is record two. It has the following characteristics:

label	-	"(MASTER DIRECTORY)"
DL's and DR's	-	unranked
words/ID	-	user specified
W6	-	zero

The DBA of the master directory is always two. Therefore it provides a standard entry to the DB file. Every DR and DL in the DB should have an ownership chain which leads ultimately to the master directory. The user may insert DL's and DR's in the master directory just as in any other directory.

2-4. The System Directory.

The initial DBA of the system directory is three. The system directory owns three DL's that are used internally by the DBCS. The user need never be directly concerned with the system directory. The system directory has one word per ID and the ID's of the three DL's are 0, 1, and 2. The DL's are as follows:

ID's		
0	-	The Available Record List (ARL). See Section 2-6 below.
1	-	The directory label data list.
2	-	The initial DBA's of all directories.

The directory label DL is a CDL which has the labels of all directories in the DB. It has 40 characters per element, so directory labels may have up to 40 characters. If no label is specified when a directory is created, "(UNLABELED)" is used.

The initial DBA of each directory is stored in the element of the DL with ID=2 that corresponds to its label in ID=1. In other words, the DR whose label is in element 16 of the directory label data list will have its initial DBA in element 16 of the ID=2 DL. If a DR is deleted, its position in the initial DBA DL is set to zero and the label is set to "(UNUSED)" in the directory label data list.

The system directory is not owned by the master directory. It has no owner, but it can always be addressed since it always begins in record 3. The DR and DL ranking codes are both one.

2-5. Record One.

The first record of every DB file is reserved for DBCS use. The variables that are written to it are given in Table II-3 but they are not defined here. See Section VIII.

2-6. The Available Record List.

During dynamic use of a data base, DL's and DR's may be shortened or deleted. This means that some records on the DB file will be unused. The DBCS keeps track of the DBA's of these records and re-uses them when new records are needed. The Available Record List (ARL) is an IDL whose elements contain DBA's of these available records. The ARL belongs to the system directory (Section 2-4 above).

The ARL helps to keep the DB file smaller by re-using unused space which cannot be returned to the operating system. DBA's of available records are added and removed from the end of the ARL.

Table II-3. Variables on Record One.

NWRECD(1), NWCPD, NCDPD, MXRD(1,-), MXRD(2,-), ALPHD
MLDLAD, MHDLAD, TDLAD, NWADD, KARLD(1,-)

3-0 RECORD STORAGE IN CORE MEMORY.

3-1. The Smoothed Access Frequency.

In order to reduce the number of disk accesses, the DBCS computes an access frequency for each record, and then keeps the most frequently accessed records in core. Since patterns of record accessing may change over the life of a DB, an exponentially smoothed access frequency is calculated and used as the basis for core residence decisions. In order to explain the procedure we need several mnemonics:

SAF	-	smoothed access frequency for a record
TDLAD	-	the cumulative number of record accesses for all records in the data base since it was created
ALPHD	-	exponential smoothing constant

TSLU - for a particular record, the value of TDLAD at the time its SAF was last computed

ASLU - the number of accesses to a particular record since its SAF was last computed

New SAF's are computed for records when they are written to disk, read from disk, and periodically (see variables MHDLAD and MLDLAD in Section VIII) while resident in core. The expresssion ASLU/(TDLAD-TSLU) evaluated for a particular record is the frequency that the record was accessed since its last SAF update. Approximate formulas for updating SAF are given below.

Periodic Update in Core

$$SAF_{new} = (1-ALPHD)*SAF_{old} + ALPHD*(\frac{ASLU}{TDLAD-TSLU})$$

After Reading from Disk

$$SAF_{new} = (1-ALPHD)*SAF_{old} + ALPHD*(\frac{1}{TDLAD-TSLU})$$

Before Writing to Disk

$$SAF_{new} = (1-ALPHD)*SAF_{old} + ALPHD*(\frac{ASLU}{TDLAD-TSLU})$$

The formulas are approximate because TDLAD-TSLU is not the same at every update for every record. To compensate for this, the second term is weighted for the average update period in the formula for reading and writing from/to disk. SUBROUTINE SAFD contains the details.

Every record in core (whether it has been resident for some time or has just been read) has a SAF value which is approximately comparable. In other words, it represents the smoothed access frequency of the record based upon nearly the same criteria. The DBCS uses these values to determine which records will have a tendency to be retained in core.

3-2. The Internal, Temporary, and Working Stores.

The DBCS has two large core storage areas, one for numeric data and one for character data, in which data records are stored. The character storage is in a single CHARACTER variable named CSTORD. Numeric data is stored in a one dimensional array named STORD and KSTORD (the two names are assigned with the EQUIVALENCE statement). Each of these storage areas are partitioned into areas that provide three main types of storage for the DBCS.

The Internal Store (IS) (actually there are two, a character internal store and a numeric internal store) is storage used in a transient manner by the DBCS. No records remain in IS space after the

immediate operation which the DBCS is performing is completed. A stack of IS records slots is maintained and allocated to DBCS processes when needed. These processes then return the slot when done with them.

The two other areas, the Temporary Store (TS) and Working Store (WS), are where the DBCS stores records semi-permanently, and it is in these areas that the SAF is significant. When a request is made for a record that is not in core, it is read into the TS. This means that a record in the TS may have to be over-written. If so, the one with the lowest SAF (usually) is selected. The record remains in the TS and may be accessed again while there. Periodically the DBCS recomputes SAF's for all records in the TS and WS. Any record in the TS with a SAF larger than the minimum SAF in the WS is swapped into the WS. Thus records which are accessed frequently will appear in the TS frequently and will eventually migrate to the WS where they have a high degree of core residence persistence.

One exception to the above description occurs. The DBCS allows two data base files to be accessed simultaneously. They are referred to as the Working data base and the Reference data base. The DBCS assumes that the reference DB is to be used only to copy information to and from the working DB. It assumes that the application program performs its main data transactions with the working DB. Therefore, SAF's are never re-computed for reference DB records, and reference DB records are never moved into the working store. They may exist only in the temporary store. All DBCS functions may be performed on the reference DB, but because of the above restrictions, they will usually execute much slower than equivalent operations on the working DB.

3-3. Structure of CSTORD and KSTORD/STORD.

The KSTORD/STORD is partitioned into nine sections. An array, KNPD, with nine elements contains pointers to the first element of each section. In other words, KNPD(6) is the first element of the sixth section of KSTORD. The data stored in each section are shown in Table II-4.

Similarly, the data stored in the character variable, CSTORD, is partitioned by an array KCPD with three elements. Table II-5 shows the CSTORD position. In this case, KCPD(i) is the first character position in CSTORD of section i.

Before describing the structure in each section of the storage areas we need to define the Data Base Control System Address or DBCSA. When a record is read into core storage, its control part is stored somewhere in KSTORD. The element of KSTORD where the first word of the control part is stored is its DBCSA. This is in contrast to the data base address, or DBA, discussed earlier which is the record's address on the data base file (i.e., on disk).

Table II-4. The Partition of KSTORD/STORD.

Section	Contents
1	Control parts of records in the character internal store
2	Control parts of records in the character temporary store
3	Control parts of records in the character working store
4	Not used
5	Records in the numeric internal store
6	Records in the numeric temporary store
7	Records in the numeric working store
8	Records of the available record list
9	Not used

Table II-5. The CSTORD Partition.

Section	Contents
1	Data parts of records in the character internal store
2	Data parts of records in the character temporary store
3	Data parts of records in the character working store

In-core storage for the Internal Store is shown in Figure II-5. Section one of KSTORD is partitioned in segments with two more elements than the length of the control part. As specified above, the DBCSA is the first element of the control part. An "in-use" indicator specifies whether this record storage area is currently being used or if it is available to be used. Finally, a pointer to the first character of the data part that corresponds to this control part is also stored in KSTORD.

Section 5 of KSTORD/STORD is partitioned into segments to hold the control part, the data part, and one extra word (which is the in-use indicator) of the numeric internal store.

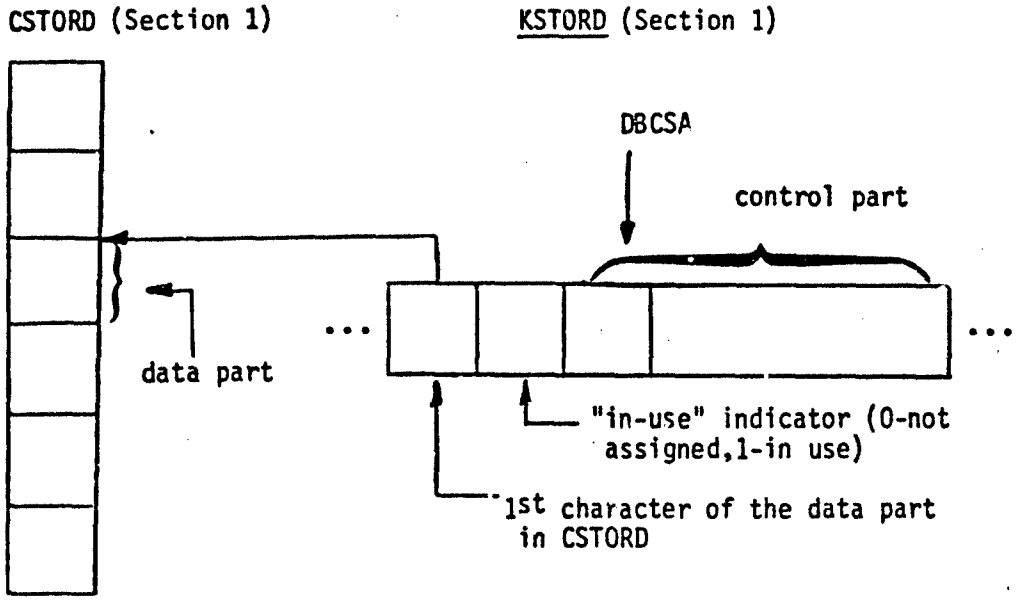
The storage scheme for the character TS and WS is shown in Figure II-6. Storage for the numeric TS and WS is shown in Figure II-7. The main difference is that the character data part is stored in CSTORD, and a pointer to the data part is maintained in KSTORD. The SAF and ASLU values for the in-core records is stored for use in updating the SAF. Also, a indicator of the DB (working or reference) is maintained.

The predecessor and successor DBCSA's need explanation. As discussed previously, records of data lists and directories are linked on the DB file by predecessor and successor DBA's. In a similar way, those records that belong to a particular record chain (a DL or DR) that are simultaneously in core are linked by their DBCSA's. For example, suppose a data list has four records. On disk, the DBA chain links the first to the second, the second to the third, and so on. Suppose at some time only the first and third records of the DL are both in core. The DBCSA chain in the TS and WS would link the first to the third. It is important to understand that the on-disk DBA chain and the in-core DBCSA chain are different. The DBCSA chain may not include some records in the DBA chain. The DBCS uses the DBCSA pointers to keep track of those records which belong to the same directory or data list.

Finally, the structure of the ARL storage is shown in Figure II-8. Only a single record of the ARL's for the working and reference DB's are stored in core. Thus, the ARL's are handled somewhat differently than other DL's. Section 8 of KSTORD/STORD is dedicated to the core storage for these records.

The size of the arrays KSTORD/STORD and CSTORD and the amount of storage allocated to their various partitions is controllable by the user. The procedures for doing so are contained in Section VI.

Character Internal Store



Numeric Internal Store

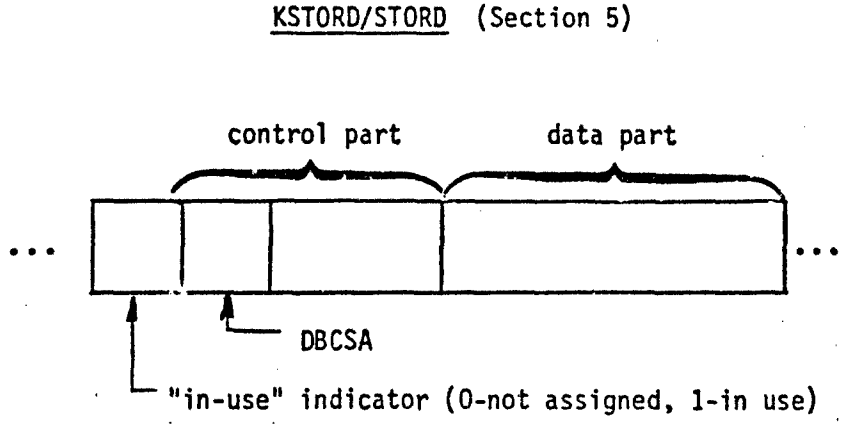


Figure II-5. Core Storage for the Internal Store.

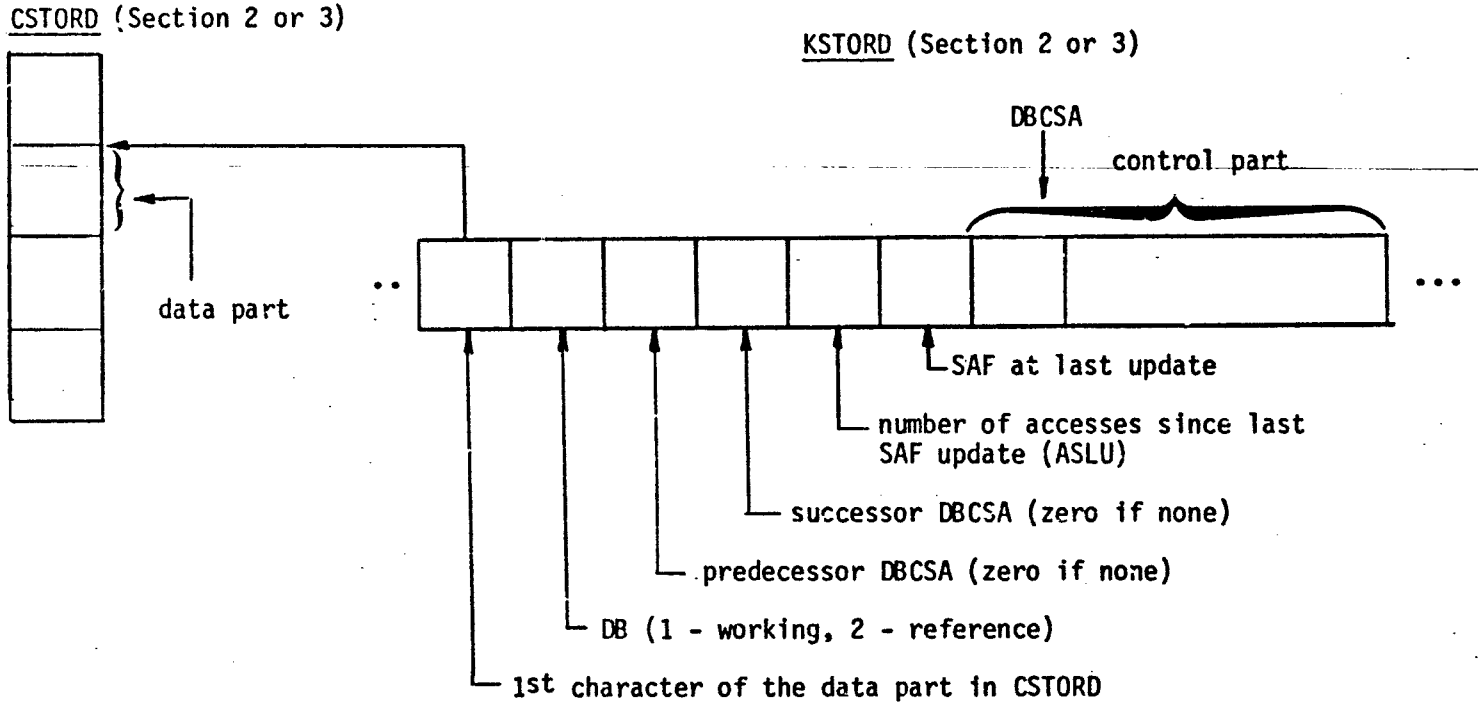


Figure II-6. Core Storage of the Character TS and WS.

KSTORD/STORD (Section 6 or 7

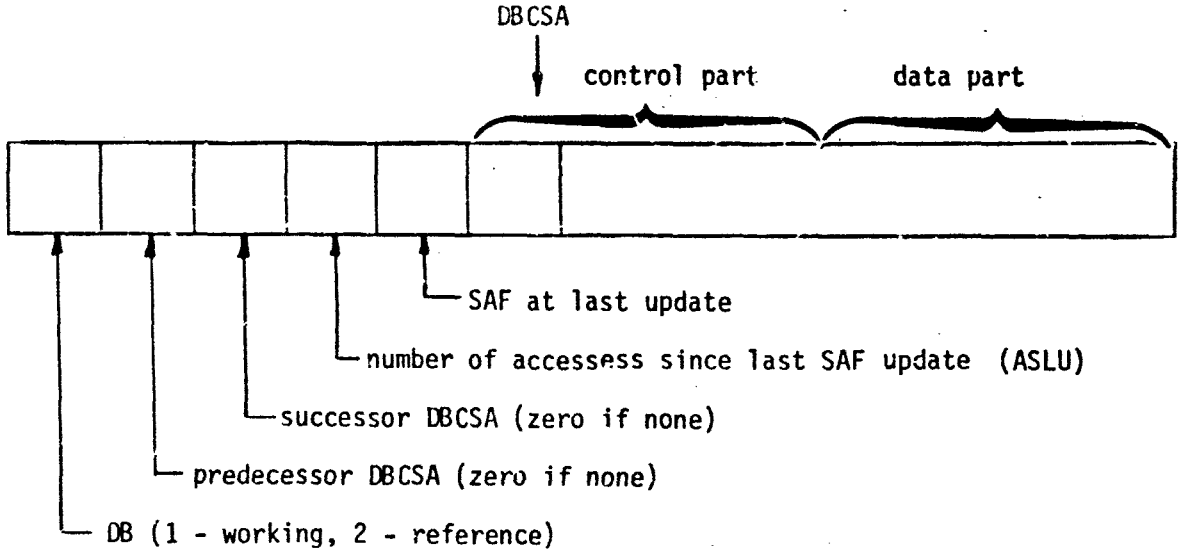


Figure II-7. Core Storage of the Numeric TS and WS.

KSTORD/STORD (Section 8)

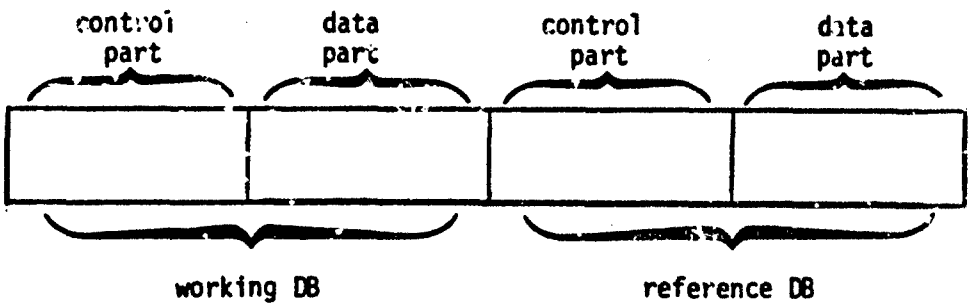


Figure II-8. Core Storage of the ARL.

4-0 DBCS PROTECTION MODES.

4-1. Read/Write Protection Modes.

Each data base (working and reference) has a user assigned read/write protection mode. The available modes are explained in Table II-6. For mode 1, no changes are permitted to the DB. The DBCS itself may change control information on the DB file, but the user may not change data in the DB.

There are three write modes that trade off increasing access speed with safety. With mode 2 every change to the data base is immediately written to disk, so the DBF is always current. This is the safest but also the slowest access method. With mode 3, changes to data lists are not written to disk until the record in which the change occurs is removed from core (i.e., bumped from the temporary store). Thus, the DB can be out of date while an application program is running. The structure and integrity of the DB is always protected, however. This mode provides substantial speed advantages over mode 2 when active read/write activity is performed on many data lists.

Mode 4 is similar to mode 3 except that even data necessary to protect the integrity of the DBF is not written to core until absolutely necessary. If an abnormal termination of the program occurs while in this mode, portions of the DBF may not be recoverable.

Table II-6. Read/Write Protection Modes.

Mode Value	Meaning
1	Read Only
2	Write-Safe. The DB may be changed, and every change is immediately written to disk.
3	Write-Loose. The DB may be changed, but only data needed to protect the integrity of the DB is written to disk immediately as it occurs. Changes to data lists may exist only in core and the disk record will be updated only when the record is removed from core storage. In case of abnormal termination, some data lists may lose information.
4	Write-Unsafe. The DB may be changed, but no data is written to disk unless absolutely necessary. If abnormal termination occurs, the structure of the data base may be flawed and irrecoverable.

When a DBF is first created by the DBSS, it has mode 1. The user may change the mode as often as desired with a subprogram call (see Section VI).

4-2. Force Read/Write Flags.

Because of the read/write protection modes described above, it is possible for a data list record in core to be different from the corresponding disk record. Each subprogram which accesses data list information has force read or force write flag as one of its parameters.

If, when writing information to a data list, the force write flag is "on" then the effected records will immediately be written to disk. This emulates mode 2 for one write access only. In this way, it is possible to treat certain data lists as though mode 2 were in effect when, in fact, the true mode is, say, 3. Note, however, that the force write flag does not override mode 1.

Similarly, when reading from a DL, a force read flag may be turned "on" which causes the record to be read from disk even if the record is resident in the TS or WS. This disk read will overwrite the in-core version if there is one. If the DB is in mode 3 or 4, then the in-core version of a DL record might contain information different than that on disk. Using the force read flag in this case will cause the in-core version to become identical to the disk version, and a DBCS error code 7 will result (see Section VII). This does not terminate execution, but it warns the user that information has been lost.

III. OVERVIEW OF THE FLOW OF CONTROL

1-0 GENERAL DESCRIPTION.

The DBCS is a collection of subprograms which an application program may use to create and manipulate a data base file. The DBCS has no main program, so it has no stand alone capability. It was designed to provide rapid access to data for the MOPADS system. The DBCS can be used in settings other than MOPADS, because it has no built in structure that is unique to MOPADS. It does not, however, have many traditional data base features because of its specialized target environment. It also imposses a somewhat greater burden on application programs than other similar data base systems. The DBCS requires the application program to remember DB address information which is used by the DBCS to implement fast data access (see Sections V and VI).

2-0 DBCS FUNCTIONS.

The DBCS provides capabilities to perform the following functions on DB files:

1. Open/Close DB Files
2. Add/Delete/Rename Directories
3. Add/Delete/Extend/Shorten Data Lists
4. Read/Write Data Lists
5. Set/Change the DB Protection Mode
6. Search Directories for Particular DL's or DR's
7. Set/Access/Change Labels of Data Lists and Data List Elements
8. Read/Write External Format Data Files for Portability of DB Files
9. Set/Access Various DBCS Options
10. Print Contents of DL's and DR's

All of these functions are described in greater detail in Section VI.

3-0 DATA BASE ACCESSES.

The most frequent operation performed on the data base is to store or retrieve data from a data list. A schematic of this process is shown in Figure III-1. A variety of subprograms are

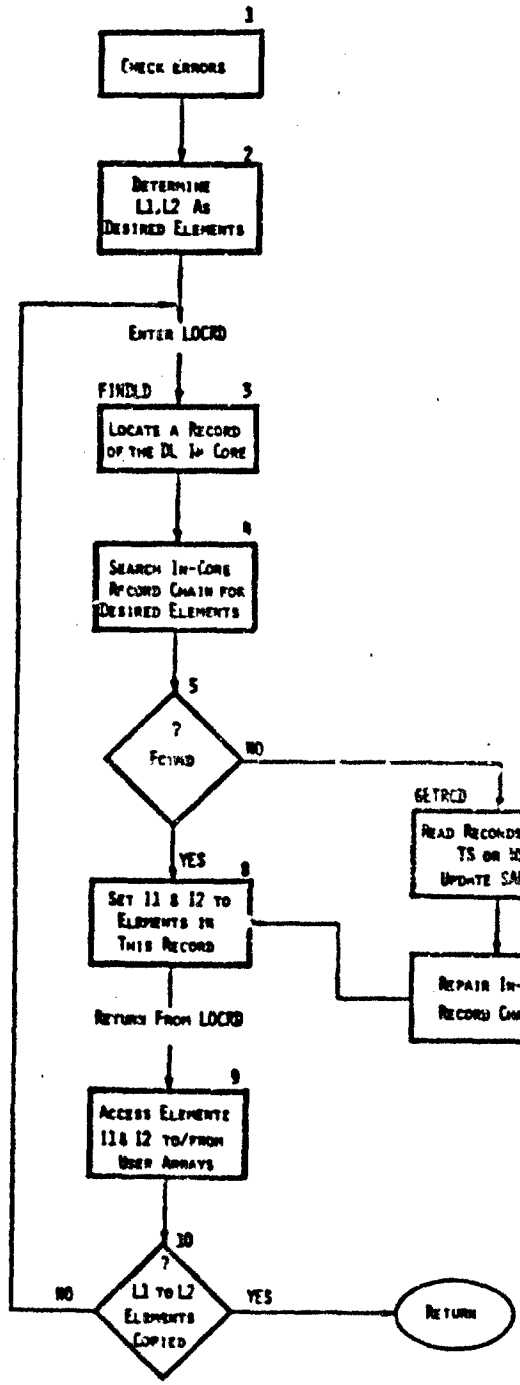


Figure III-1. Program Flow for Data List Access.

represented by this diagram since separate subprograms are used depending upon whether the operation is a read or write, the DL is one or two dimensional, and the data type of the DL. Figure III-1 is typical for all cases, however.

Virtually all user callable DBCS subprograms have error trapping features. These are represented generically in box 1 of Figure III-1. The next step is to determine the elements, L1 to L2, of the data list that are desired. Note that these elements may cross record boundries. In other words, some of the elements may be in one record and some on subsequent records. In the case of two dimensional DL's, this step involves decoding the structure of the DL so it can be treated as a one dimensional DL.

The workhorse of DB accessing is SUBROUTINE LOCRD. It's primary activity is to locate records in core which contain elements in the interval L1 to L2. The user will have passed a data base address (to be described in Section VI) which will point directly to a record of the DL in-core if the normal dynamic core memory management has not moved it. If this is the case, block 3 (SUBROUTINE FINDLD) is trivial. If the DL record has been moved, FINDLD will find it again (or read it from disk if needed).

At block 4, LOCRD will search the in-core record chain for elements in [L1,L2]. If the required elements are not found (block 5), SUBROUTINE GETRCD will be called. When GETRCD is called, LOCRD has identified the exact record which will contain the desired elements. Only a single block represents GETRCD, but in fact it performs an important function. GETRCD and the subprograms it calls are responsible for all of the dynamic memory management of the TS and WS. It is in these subprograms that SAF's are checked and updated and that records migrate from the TS to the WS. On return from GETRCD, the record is in-core. LOCRD repairs the in-core record chain (block 7).

At block 8, LOCRD sets I1 and I2 to the element numbers (a subset of [L1,L2] that are contained in the record it has found. The calling program accesses these elements (i.e., write to or copies from them)(block 9), then it determines if all of the elements L1 to L2 have been accessed (block 10). If not, LOCRD is called again to find another record with more of the desired elements.

When frequently accessed records that are in the working store are being requested, the step at block 3 is trivial and GETRCD is not called. All other operations in LOCRD are simply stepping through a linked list of records in-core which is quite fast. The result is that data base accesses from records in the WS or TS are relatively fast.

Access of data in directories is handled in the same way since directories are two dimensional DL's and are stored in the MS and TS in the same way that data lists are.

4-0 THE DBCS AND MOPADS.

The MOPADS data base is the main repository of information for MOPADS simulations. It maintains the simulation data set prior to simulation runs, the dynamic simulation state during simulations, and result statistics after simulations are completed. Therefore, the DBCS is a key module during all phases of MOPADS operation. Figure III-2 is a schematic of the MOPADS modules which call the DBCS. The DBAP (Data Base Application Programs (MOPADS/DBAP)) is a collection of program utilities designed especially for manipulation of the MOPADS DB (Polito (1983b)). The DBCS is required for both batch job simulations of an defense system and for interactive user interface sessions.

5-0 UTILITIES.

The DBCS makes extensive use of programs from the MOPADS utilities module (Polito, Goodin).

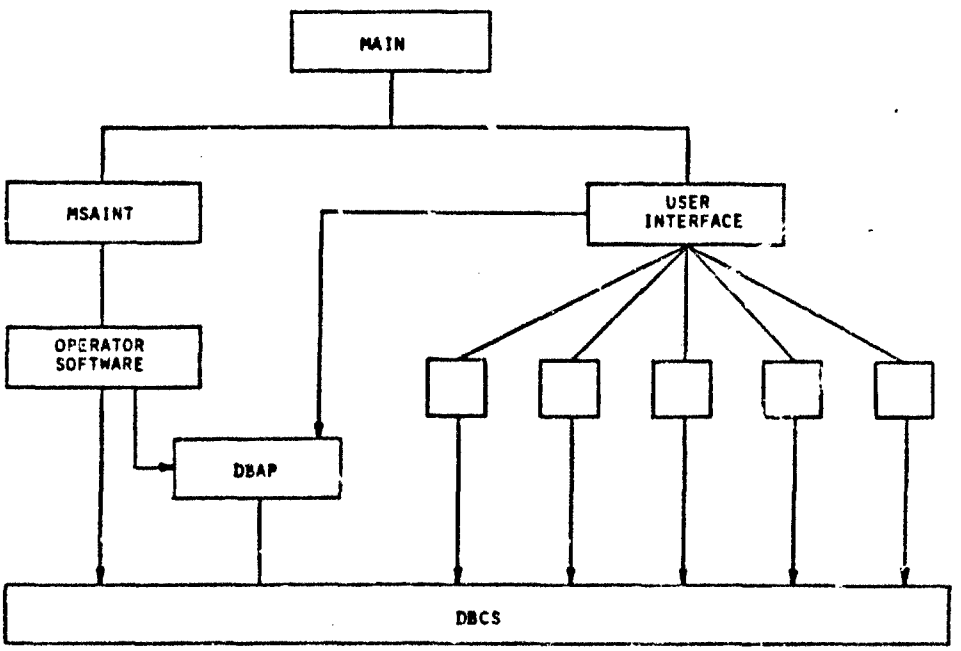


Figure III-2. DBCS and MOPADS.

IV. EXTERNAL FILE USAGE

1-0 DATA BASE FILES.

The DBCS may simultaneously access two DBF's. These files are opened and closed explicitly with FORTRAN OPEN and CLOSE statements. By default, the DBCS uses unit number one for the working DB and unit number two for the reference DB. These unit numbers may be changed by the user, however.

Data base files have the following characteristics:

1. Direct Access
2. Unformatted
3. Fixed Record Length

The record length is selectable by the user. The user gives the DB file names to the DBCS and they are opened with the specified unit number. No job control language is needed to assign the unit number to the file unless required by the computer system.

2-0 OUTPUT AND TRACE FILES.

The DBCS will produce certain output and trace information on request. Two files, an output and a trace file are used for this purpose. By default, unit number ten is used for both of these files. The unit number is user selectable, however, and need not be the same for both files. These files have the following characteristics:

1. Formatted
2. Sequential

They are not rewound by DBCS, so both files may refer to an interactive terminal. Carriage control characters are included on these files so they may be listed at a line printer.

The DBCS does not open or close these files. They must be opened and closed by the application program or assigned to the correct unit numbers by appropriate job control language statements.

3-0 BACK-UP FILES.

The DBCS provides two mechanisms for backing up data base files on magnetic tape and for transferring data base files from one computer to another. DBF's are unformatted, direct access files. Some

systems may not permit these files to be copied directly onto a magnetic tape in a way which preserves their integrity. The DBCS has a utility which will allow the application program to write a DBF to a "sequential, internal format" file. This file is a sequential, unformatted, fixed record length file which may be copied to tape. These files may also be read from tape and a normal DBF file re-created.

Since unformatted files cannot usually be transferred from one type of computer to another, the DBCS also has the ability to write the DBF to a "sequential, external format" file which is a sequential formatted file with 80 characters or less per record. Thus, DBCS data base files are portable from one computer to another.

The user has total responsibility to open, close, and assign unit numbers to these files. No defaults are provided. The user passes the unit number to the DBCS. The DBCS will generate an error if the required operations cannot be performed on the file.

V. SUBPROGRAM DESCRIPTIONS

The following pages contain description of all subprograms that are part of the DBCS. Each program description contains an explanation of its purpose, parameters, and alternate returns. No examples of subprogram use are contained in this section, since user instructions are given in Section VI.

The DBCS makes extensive use of the MOPADS utility subprograms described in Polito & Goodin (1983). Also, the MOPADS module suffix for the DBCS is "D", so all DBCS subprogram names end with the letter "D".

```
      SUBROUTINE ARISD(IOPT,IDBCSA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       ARISD ALLOCATES AND DE-ALLOCATES ELEMENTS OF THE INTERNAL STORE
C       FOR INTERNAL DBCS USE.
C--INPUT PARAMETERS:
C      IOPT-OPTION
C           1-ALLOCATE NUMERIC IS
C           2-DE-ALLOCATE NUMERIC IS
C           3-ALLOCATE CHARACTER IS
C           4-DE-ALLOCATE CHARACTER IS
C--INPUT/OUTPUT PARAMETERS:
C       IDBCSA(2)-DBCSA'S.
C                 FOR:
C               INPUT-IOPT=2, DE-ALLOCATE THE RECORD OF THE IS WHOSE
C                             CONTROL PART STARTS AT LOCATION IDBCSA(1).
C                             IDBCSA SET TO ZERO ON RETURN.
C                     IOPT=4, DE-ALLOCATE THE RECORD OF THE CHARACTER IS
C                             THAT HAS ITS CONTROL PART STORED AT
C                             IDBCSA(1) AND ITS DATA PART AT IDBCSA(2)
C                             IDBCSA SET TO ZERO ON RETURN.
C              OUTPUT-IOPT=1, IDBCSA(1)=POSITION IN THE DATA STORE OF
C                             THE 1ST WORD OF THE CONTROL PART.
C                             IDBCSA(2) IS THE POSITION OF THE FIRST
C                             WORD OF THE DATA PART.
C                     IOPT=3, IDBCSA(1)=POSITION IN THE DATA STORE OF
C                             1ST WORD OF THE CONTROL PART. IDBCSA(2)
C                             IS THE CHARACTER POSITION IN CSTOR OF THE
C                             DATA PART.
C--OUTPUT PARAMETERS:
C             IERR-ERROR CODE
C                  0-NO ERROR
C                  4003-INTERNAL STORE EXHAUSTED
C--ALTERNATE RETURNS:
C             1-IERR.NE.0
C
```

```
      SUBROUTINE ASKD(NDB,IOPR)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      ASKD WILL DETERMINE IF A WORKING OR REFERENCE DB IS OPEN.
C
```

```
C--INPUT PARAMETERS:
C       NDB-DB(1-WORKING,2-REFERENCE)
C--OUTPUT PARAMETERS:
C       IOPN-DB STATUS
C            1-OPEN
C            2-CLOSED
C
```

```
      BLOCK DATA BLOCKD
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO INITIALIZE LABELED COMMON
```

```
      SUBROUTINE BRACKD(NDB,NRANK,IVAL,IDRDL,IDBO,ISL1,ISL2,MTY,IERR,
     -                  *)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       BRACKD IS A UTILITY PROGRAM CALLED ONLY BY INSDRD.
C       IT WILL FIND COLUMNS ISL1 AND ISL2 IN A DR(IDBO) THAT
C       BRACKET A SPECIFIED VALUE(IVAL) OF THE NRANK-TH
C       ELEMENT OF DR'S OR DL'S OWNED BY IDBO.
C       THE POSITION OF IDBO IS NOT CHANGED.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       NRANK-THE RANKING CODE FOR THE DR(IDBO) FOR THE CORRECT TYPE.
C            IF NRANK=0, MTY IS RETURNED AS AN EMPTY COLUMN.
C       IVAL-THE VALUE OF THE ABS(NRANK)-TH ID ELEMENT TO BRACKET.
C       IDRDL-TYPE
C            1-DR'S
C            2-DL'S
C--INPUT/OUTPUT PARAMETERS:
C       IDBO(4)-DRAA OF THE DR. IT MUST BE CURRENT.
C--OUTPUT PARAMETERS:
C       ISL1,ISL2-COLUMNS ISL1 AND ISL2 WILL BRACKET IVAL.
C                 LET ID1 BE THE IDENTIFIER AT COLUMN ISL1.
C                 SIMILAR FOR ID2.   LET IRANK=ABS(NRANK). THEN
C                 ON RETURN,
C
C                  ISL1.LE.ISL2
C                  ID1(IRANK).LE.IVAL.LT.ID2(IRANK) IF NRANK.GT.0
C                  ID1(IRANK).GE.IVAL.GT.ID2(IRANK) IF NRANK.LT.0
C                  IF ISL1=0 AND ISL2.NE.0, ISL2 IS THE FIRST COLUMN
```

```
C                                          IN THE DR OF TYPE IDRDL.
C                IF ISL1.NE.0 AND ISL2=0, ISL1 IS THE LAST COLUMN
C                                          IN THE DR OF TYPE IDRDL
C                IF ISL1=ISL2=0,           THEN NRANK=0 OR NO ENTITIES
C                                          IOF TYPE IDRDL EXIST.
C      MTY-THE COLUMN NUMBER OF AN EMPTY COLUMN "NEAR" ISL1 AND ISL2.
C          IF ISL1=ISL2=0, MTY IS STILL AN EMPTY COLUMN.
C      IERR-ERROR CODE
C           0-NO ERROR
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE CHAIND(NDB,IOPT,IDBO,IDBN,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          CHAIND LINKS AND UNLINKS RECORD CHAINS ON THE DISK.
C          IT CHANGES ONLY THE PREDECESSOR AND SUCCESSOR RECORD
C          NUMBERS IN THE CONTROL PARTS. IT CHANGES NO LINKING
C          INFORMATION IN THE DATA STORE.
C--INPUT PARAMETERS:
C          NDB-DBF(1-WORKING,2-REF)
C          IOPT-OPTION
C               1-LINK IDBN BEFORE IDBO
C               2-LINK IDBN AFTER IDBO
C               3-UNLINK IDBO. IDBN NOT USED.
C          IDBO(2)-DBCSA'A OF THE RECORD TO LINK TO(OR UNLINK).
C                  IDBO(1)-CONTROL PART,IDBO(2)-DATA PART
C          IDBN(2)-DBCSA'A OF THE RECORD TO BE LINKED. CHAIND
C                  ASSUMES IDBN IS OF THE SAME TYPE AS IDBO.
C                  IDBN(1)-CONTROL PART,IDBN(2)-DATA PART.
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               1,2,3004,4002,4004,4005
C--ALTERNATE RETURNS:
C          1-IERR.NE.0
C
```

```
      SUBROUTINE CHECKD(ICODE,KPAR,NK,PAR,NP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO CHECK STANDARD ERROR CONDITIONS AND RETURN
```

```
C          THE APPROPRIATE ERROR CODE. CHECKD DOES NOT CALL
C          ERRORD.
C--INPUT PARAMETERS:
C          ICODE-ERROR CONDITION TO CHECK FOR. IF ICODE
C               IS INVALID, RETURN WITH IERR=0.
C       ICODE   CONDITION                  ERROR CODE
C       ----    ---------                  ----------
C         1     WORKINKG DATA BASE NOT OPEN     1
C         2     REFERENCE DATA BASE NOT
C                 OPEN                      2
C         3     DB# (KPAR(1)) =0 OR
C                 GREATER THAN USED SO FAR  4
C         4     SAME AS 3 BUT IN SERIOUS
C                 SITUATION                 4005
C         5     CHECK VALID DL OR DR
C               ELEMENT NUMBER. KPAR(1).
C               LT.KPAR(0).OR.KPAR(0)
C               .LE.0                     5
C         6     CHECK SUFFICIENT SPACE
C               IN KWORKD ARRAY.          4009
C         7     DB IS READ ONLY. KPAR(0)=DB
C               NUMBER (1 OR 2)           8
C
C          KPAR(NK)-LIST OF INTEGER PARAMETERS TO BE USED TO
C                   CHECK THE ICODE CONDITION. IF NK=0, NONE.
C          PAR(NP)-LIST OF REAL PARAMETERS TO BE USED TO CHECK
C                  THE ICODE CONDITION. IF NP=0, NONE.
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE FOR THE ICODE CONDITION IF IT IS TRUE.
C              ZERO IF NOT.
C--ALTERNATE RETURNS:
C          1-IF IERR=0 (I.E. NO ERROR)
C
```

```
      SUBROUTINE CLEARD(NDB,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO CLEAR THE US AND TS OF ENTRIES OF THE INDICATED DB.
C       THIS MAY BE DONE WHEN A DIFFERENT SET OF DATA LISTS
C       IS TO BE ACCESSED OR WHEN A NEW DB IS TO BE OPENED.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(0-BOTH,1-WORKING,2-REFERENCE)
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           .NE.0-ERROR WRITING TO DBF
```

```
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE CLRDLD(NDB,IDRAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       CLRDLD WILL DELETE ALL DL'S FROM A DIRECTORY.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY TO CLEAR
C
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE CLRDRD(NDB,IDRAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       CLRDRD WILL EMPTY A DR OF ALL CONTENTS(THE DR IS NOT
C       DELETED,HOWEVER). ALL OWNED DL'S ARE DELETED AND ALL
C       OWNED DR'S (WITH THEIR CONTENTS) ARE DELETED.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C           15-IDRAA NOT A DR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE CLRDSD(NDB,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
```

```
C       TO CLEAR A RECORD CHAIN FROM THE DATA STORE.
C       THIS PROGRAM IS USED MAINLY TO REMOVE LDL'S
C       FROM THE DATA STORE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF ONE RECORD OF THE CHAIN.
C                ON RETURN IDBAA(2)=0
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------ ----------------------------------------------------------

      SUBROUTINE CLSDBB(NDB,ISTAT,IERR,IOERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           TO CLOSE A DBF
C--INPUT PARAMETERS:
C           NDB-DF(1-WORKING,2-REF)
C               NO ACTION IF THE DB IS NOT OPEN
C           ISTAT-STATUS(1-KEEP,2-DELETE)
C--OUTPUT PARAMETERS:
C           IERR-ERROR CODE
C                0-NO ERROR
C                3004 SYSTEM I/O ERROR
C           IOERR-SYSTEM I/O ERROR CODE
C                 0-NO ERROR
C                 SYSTEM I/O ERROR NUMBER IF IERR=3004
C--ALTERNATE RETURNS:
C           1-IERR.NE.0

----------------------------------------------------------------------------

      SUBROUTINE CRDLD(NDB,ITYPE,NDIM,LABEL,IDBO,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO CREATE A DATA LIST
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       ITYPE-TYPE OF DATA LIST
C            1-IDL
C            2-RDL
```

```
C            3-CDL
C      NDIM-FOR IDL AND RDL, THE ROW OR COLUMN DIMENSION.
C             (+) VALUES-MAX ROW DIMENSION(I.E. COLUMN ORIENTED)
C             (-) VALUES-MAX COLUMN DIMENSION(I.E. ROW ORIENTED)
C              FOR 1-D DATA LISTS, USE 1
C           FOR CDL, THE NUMBER OF CHARACTERS/ELEMENT.
C           IF NDIM.EQ.0, A VALUE OF 1 WILL BE USED.
C      LABEL-(CHARACTER) LABEL OF THE DATA LIST(LIMITED TO
C            25 CHARACTERS). IF BLANK, THE LABEL
C            '(UNLABELED)' WILL BE USED.
C--INPUT/OUTPUT PARAMETERS:
C      IDBO(4)-DBAA OF THE OWNING DIRECTORY(CRDLD DOES NOT INSERT
C           THE NEW DATA LIST IN THE OWNING DIRECTORY)
C--OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA FOR THE NEW DATA LIST
C      IERR-ERROR CODE
C           0-NO ERROR
C        3005-TRY TO CREATE CDL WITH TOO MANY CHARACTERS/ELEMENT
C--ALTERNATE RETURNS:
C      1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CRDRD(NDB,NRDR,NRDL,NWID,LABEL,IDBO,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO CREATE A DIRECTORY
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      NRDR-RANKING CODE FOR OWNED DIRECTORIES
C           0-NOT RANKED
C           N-INCREASING ORDER OF WORD N OF THE ID
C         (-N)-DECREASING ORDER OF WORD N OF THE ID
C      NRDL-RANKING CODE FOR OWNED DL'S
C           (THE MEANINGS ARE THE SAME AS FOR NRDR)
C      NWID-NUMBER OF WORDS PER ID FOR OWNED DR'S AND DL'S
C      LABEL-(CHARACTER) LABEL OF THE DIRECTORY(LIMITED TO
C            40 CHARACTERS). IF BLANK, THE LABEL
C            '(UNLABELED)' WILL BE USED.
C--INPUT/OUTPUT PARAMETERS:
C      IDBO(4)-DBAA OF THE OWNING DIRECTORY(CRDRD DOES NOT INSERT
C           THE NEW DIRECTORY IN THE OWNING DIRECTORY)
```

```
C--OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA FOR THE NEW DIRECTORY
C       IERR-ERROR CODE
C            0-NO ERROR
C            14-NRDR/NRDL INCORRECT OR INCOMPATIBLE WITH NUID
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CURD(NDB,IDRDL,NID,IDRAA,IDAA1,ID,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       CURD WILL RETURN THE DBA AND ID OF THE CURRENT DL OR DR
C       POSITION OF A DIRECTORY.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IDRDL-POSITION TYPE
C            1-DR
C            2-DL
C       NID-LENGTH OF ID. IF NID.LE.0, ID WILL NOT BE USED.
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C       IDAA1-DBA OF THE CURRENT POSITION OF TYPE IDRDL.
C            IF THE DIRECTORY IS NOT POSITIONED, IDAA1=0
C            AND ALTERNATE RETURN 1 IS TAKEN.
C       ID(NID)-THE FIRST NID WORDS OF THE IDENTIFIER. IF
C              NID.LE.0, ID IS NOT USED. IF ALTERNATE RETURN
C              1 IS TAKEN, ID=0.
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-DIRECTORY POSITION OF TYPE IDRDL IS NOT DEFINED.
C         IDAA1=0.
C       2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE DBOPTD(IOPT,IPG,NIV,NRV,IVL,RVL,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       DBOPTD WILL SET OR RETURN THE VALUES OF DBCS OPTIONS AND
C       PARAMETERS.
C--INPUT PARAMETERS:
C       IOPT-OPTION NUMBER(SEE BELOW)
C       IPG-PUT/GET FLAG
```

```
C          1-PUT NEW OPTION VALUE
C          2-RETRIEVE OPTION VALUE
C--INPUT/OUTPUT PARAMETERS:
C      IVL(NIV)-ARRAY TO HOLD INTEGER VALUES
C      RVL(NRV)-ARRAY TO HOLD REAL VALUES
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C          17-IOPT INCORRECT
C          18-INVALID UNIT NUMBER
C          19-INSUFFICIENT ROW/COLUMN SPECIFICATION ON OUTPUT FILE
C        2004-UNIT NUMBER OF AN OPEN DB MAY NOT BE CHANGED
C        2005-BOTH DB'S MUST BE CLOSED TO RECONFIGURE THE DATA STORE
C        2006-RECORD LENGTH TOO SHORT
C
C--OPTIONS AVAILABLE
C
C     VALUES OF IOPT    OPTION                                        TYPE
C     --------------    ------                                        ----
C              1        UNIT NUMBER FOR WORKING DATA BASE              I
C              2        UNIT NUMBER FOR REFERENCE DATA                 I
C              3        UNIT NUMBER FOR DBCS OUTPUT FILE               I
C              4        UNIT NUMBER FOR DBCS TRACE FILE                I
C              5        TRACE CODE                                     I
C              6        ERROR PRINT FLAG                               I
C              7        DB PROTECTION MODE FOR WORKING DB              I
C              8        DB PROTECTION MODE FOR REFERENCE DB            I
C              9        SAF SMOOTHING CONSTANT                         R
C             10        LOWER BOUND ON SAF UPDATE PERIOD               I
C             11        UPPER BOUND ON SAF UPDATE PERIOD               I
C             12        (1)-NUMBER OF PRINT COLUMNS ON THE
C                           DBCS OUTPUT FILE                         I
C                       (2)-NUMBER OF LINES/PAGE ON DBCS
C                           OUTPUT FILE                              I
C             13        (1)-NUMBER OF PRINT COLUMNS ON THE
C                           DBCS TRACE FILE                          I
C                       (2)-NUMBER OF LINES/PAGE ON DBCS
C                           TRACE FILE                               I
C             14        NUMBER OF WORDS PER RECORD ON DB FILES         I
C             15        NUMBER OF RECORDS IN THE INTERNAL STORE        I
C             16        NUMBER OF RECORDS IN THE NUMERIC
C                       TEMPORARY STORE                              I
C             17        NUMBER OF RECORDS IN THE CHARACTER
C                       TEMPORARY STORE                              I
C
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      FUNCTION DBRND()
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO GENERATE RANDOM NUMBERS FOR THE DBCS.  THIS SUBROUTINE IS
C       A SPECIALIZED VERSION OF FUNCTION DRAND IN MSAINT.
C
C
C***** ALGORITHM FOR SEED GENERATION IS ADAPTED FROM FUNCTION RAND
C***** DEVELOPED BY L. SCHRAGE AND USED WITH THE AUTHOR'S PERMISSION.
C
```

```
      SUBROUTINE DELDLD(NDB,IDBAA,IDRAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       DELDLB WILL DELETE A DATA LIST. IF WILL ALSO REMOVE
C       ITS ENTRY FROM THE DR THAT OWNS IT.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DL. IT WILL BE ZERO ON RETURN
C                IF DELETE IS SUCCESSFUL.
C       IDRAA(4)-DRAA OF THE OWNING DIRECTORY,IF KNOWN. IF NOT,
C                SEND IDRAA(1)=0.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C           12-IDBAA NOT A DL
C         4008-DL IDBAA NOT IN CORRECT DIRECTORY. DB FLAWED
C         4009-INSUFFICIENT SPACE IN KWORK
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE DELDRD(NDB,IDRAA,IDRDAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       DELDRD WILL DELETE A DIRECTORY.  THE DR MUST BE
C       EMPTY.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
```

```
C       IDRAA(4)-THE DRAA OF THE DR TO BE DELETED. ON RETURN
C                IT WILL BE ZERO.
C       IDRDAA(4)-DRAA OF THE OWNING DIRECTORY IF KNOWN. IF NOT KNOWN,
C                 SEND IDRDAA(1)=0.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C         2001-IDRAA NOT EMPTY
C           15-IDRAA NOT A DR
C         4010-IDRAA NOT IN OWNER DIRECTORY
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE ERRORD(PROGM,MSG,IERR,KPAR,NK,PAR,NP)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           ERROR PROCESSING FOR THE DBCS
C--INPUT PARAMETERS:
C           PROGM-CALLING PROGRAM NAME(CHARACTER)
C           MSG-ERROR MESSAGE(CHARACTER)
C           IERR-ERROR CODE
C                1-1999          INFORMATIVE
C                2000-2999       WARNING
C                3000-3999       SEVERE
C                4000-           FATAL
C
C               THE DBCS USES POSITIVE CODES ONLY.
C               THE USER MAY CALL ERRORD WITH NEGATIVE
C               CODES. THE ABSOLUTE VALUE OF THE CODE WILL
C               DETERMINE ITS SEVERITY AS ABOVE.
C           KPAR(NK)-INTEGER ARRAY TO BE PRINTED WITH
C                    ERROR MESSAGE IF NK.GT.0.
C           PAR(NP)-REAL ARRAY TO BE PRINTED WITH
C                   ERROR MESSAGE IF NP.GT.0.
C
```

```
      SUBROUTINE FDBID(NDB,ID,NID,IDRDL,IDRAA,IDAA,IWRK,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       SEARCH FOR AN ID OF A DL OR DR IN A DIRECTORY AND RETURN THE
C       DBAA.  THE DL OR DR POSITION OF THE DIRECTORY WILL BE MOVED
C       TO THE FOUND ID. IF IT IS NOT FOUND, THE POSITION IS NOT
C       CHANGED.
```

```
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       ID(NID)-ID TO SEARCH FOR
C       IDRDL-TYPE TO SEARCH
C             1-DR'S
C             2-DL'S
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY TO SEARCH
C--OUTPUT PARAMETERS:
C       IDAA(4)-DBAA OF THE DR OR DL WITH ID="DL".
C               IF IDRDL=2,ONLY 2 WORDS OF IDAA ARE USED. IDAA=0
C               IF "ID" IS NOT FOUND.
C       IWRK(3,NID)-INTEGER WORKING SPACE FOR FDBID
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-"ID" NOT FOUND
C       2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE FIDLD(NDB,IDBA,IDLDR,NID,IDRAA,ID,LABEL,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       FIDLD WILL RETURN THE ID AND LABEL OF A DR OR DL
C       THAT HAS A SPECIFIED DBA.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IDBA(4)-DBAA OF THE DL OR DR TO FIND
C       IDLDR-TYPE FOR IDBA
C             1-DR
C             2-DL
C       NID-LENGTH OF ID ARRAY. NID WORDS OF THE IDENTIFIER
C           WILL BE RETURNED. IF NID IS GREATER THAT NEEDED,
C           THE REMAINDER WILL NOT BE CHANGED. IF NID.LE.0,
C           ONLY THE LABEL WILL BE RETURNED(ID NOT USED)
C
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY TO SEARCH FOR IDBA. IT WILL
C                BE POSITIONED TO THE FOUND DL OR DR.  IF NOT
C                FOUND, THE POSITION WILL NOT BE CHANGED.
C--OUTPUT PARAMETERS:
C       ID(NID)-ARRAY TO HOLD IDENTIFIER OF THE FOUND DL OR DR.
C               ZERO IF NONE FOUND. SEE NID.
C       LABEL-CHARACTER VARIABLE TO HOLD THE DL OR DR LABEL.
C             DL LABELS MAY HAVE 25 CHARACTERS. DR LABELS MAY HAVE
C             40.
```

```
C       IERR-ERROR CODE
C             0-NO ERROR
C
C--ALTERNATE RETURNS:
C       1-IDBA NOT FOUND
C       2-IERR.NE.0
C
```

```
      SUBROUTINE FINDLD(NDB,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO LOCATE A RECORD OF THE DL OR DR IN THE DATA
C       STORE AND RETURN ITS CURRENT DBAA IN IDBAA.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERNCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-A DBAA OF A DL OR DR. IDBAA WILL RETURN
C                UNCHANGED IF IDBAA(2) IS A MEMBER OF THE
C                DL SPECIFIED IN IDBAA(1). IF NOT, A
C                 RECORD OF THE CORRECT DL WILL BE LOCATED(OR READ)
C                 AND ITS LOCATION STORD IN IDBAA(2)
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0 ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE FMNSFD(ITYP)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          FMNSFD FINDS THE MINIMUM SAF IN THE US OF
C          TYPE ITYP AND SETS THE VALUES OF SAFMND AND MNSFD.
C--INPUT PARAMETERS:
C          ITYP-US TYPE
C               1-NUMERIC
C               2-CHARACTER
```

```
      SUBROUTINE GCD(NDB,IP1,IP2,IR,NDL,IDBAA,CDL,IERR,LAST,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A CHARACTER DL TO THE ARRAY CDL
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IP1,IP2-ELEMENTS IP1 TO IP2 WILL BE COPIED TO THE
C               ARRAY CDL. IF NDL.GT.IP2-IP1+1, THE
C               REMAINDER OF CDL IS NOT DISTURBED. IF NDL.LT.
C               IP2-IP1+1, ONLY NDL ELEMENTS WILL BE COPIED.
C               SEE IERR. IF IP2.LT.IP1 OR IP1.LE.0, RETURN
C               WITH NO ACTION.
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA
C            TO BE OVERWRITTEN, IERR=7 ON RETURN.
C       NDL-LENGTH OF CDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRETORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOUSLY. GCD MAY CHANGE THE
C                VALUES OF IDBAA TO REFLECT CURRENT DBCS
C                ADDRESSING. (1)-DBA, (2)-DBCSA.
C--OUTPUT PARAMETERS:
C       CDL(NDL)-CHARACTER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IP1 AND/OR IP2 INVALID
C            6-IP2 EXTENDS PAST THE END OF THE DL OR DR
C            7-FORCE READ CAUSES DATA TO BE OVERWRITTEN.
C       LAST-LAST HAS TWO MEANINGS
C            IF IERR.NE.6, LAST IS THE LAST DL ELEMENT COPIED
C                          TO THE OUTPUT ARRAY.
C            IF IERR=6,    LAST IS THE LAST ELEMENT OF THE DL.
C                          EXISTING ELEMENTS BETWEEN IP1 AND IP2
C                          FOR WHICH THERE WAS ROOM IN THE OUTPUT
C                          ARRAY HAVE BEEN COPIED.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE GCOLCD(NDB,NCOL,IR1,IR2,IR,NDL,IDBAA,CDL,IERR,LASTC,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A COLUMN OF A 2-D DL TO THE CHAR ARRAY CDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               COPIED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF CDL
C               WILL NOT BE DISTURBED. IF NDL.LT.IR2-IR1+1, ONLY
C               NDL ROW ELEMENTS WILL BE COPIED. SEE IERR. IF
C               IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C            BE OVERWRITTEN, IERR=7 ON RETURN
C       NDL-LENGTH OF CDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       CDL(NDL)-CHARACTER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IR1 AND/OR IR2 INVALID
C            6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C              SEE LASTC.
C            7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT CHARACTER
C       LASTC-IF IERR.NE.6, LASTC=0
C             IF IERR=6, LASTC=THE LAST DEFINED COLUMN OR ROW
C                        IN THE DL.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C

-------------------------------------------------------------------------

      SUBROUTINE GCOLID(NDB,NCOL,IR1,IR2,IR,NDL,IDBAA,IDL,IERR,LASTC,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
```

```
C       TO COPY DATA FROM A COLUMN OF A 2-D DL TO THE INTEGER ARRAY IDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFFRENCE)
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               COPIED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF IDL
C               WILL NOT BE DISTURBED. IF NDL.LT.IR2-IR1+1, ONLY
C               NDL ROW ELEMENTS WILL BE COPIED. SEE IERR. IF
C               IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C            BE OVERWRITTEN, IERR=7 ON RETURN
C       NDL-LENGTH OF IDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IDL(NDL)-INTEGER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IR1 AND/OR IR2 INVALID
C            6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C              SEE LASTC.
C            7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT INTEGER
C       LASTC-IF IERR.NE.6, LASTC=0
C             IF IERR=6, LASTC=THE LAST DEFINED COLUMN OR ROW
C                        IN THE DL.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
-------------------------------------------------------------------

      SUBROUTINE GLOLRD(NDB,NCOL,IR1,IR2,IR,NDL,IDBAA,RDL,IERR,LASTC,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A COLUMN OF A 2-D DL TO THE REAL ARRAY RDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               COPIED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF RDL
```

```
C              WILL NOT BE DISTURBED. IF NDL.LT.IR2-IR1+1, ONLY
C              NDL ROW ELEMENTS WILL BE COPIED. SEE IERR. IF
C              IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C      IR-FORCE READ FLAG
C         1-DO NOT FORCE READ
C         2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C           BE OVERWRITTEN, IERR=7 ON RETURN
C      NDL-LENGTH OF RDL
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C               BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C               SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C               (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C      RDL(NDL)-REAL ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C      IERR-ERROR CODE
C           0-NO ERROR
C           4-IDBAA INVALID
C           5-IR1 AND/OR IR2 INVALID
C           6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C           7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
C           9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C          10-DL NOT REAL
C      LASTC-IF IERR.NE.6, LASTC=0
C            IF IERR=6, LASTC=THE LAST DEFINED COLUMN OR ROW
C                       IN THE DL.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE GDLLBD(NDB,LABEL,IDRAA,IDBAA,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      GDLLBD WILL FIND THE DBAA OF A DL THAT HAS A SPECIFIED LABEL IN
C      A DIRECTORY.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      LABEL-(CHARACTER) LABEL OF THE DL. IT MUST EXACTLY MATCH AN
C           EXISTING DL LABEL IN THE DIRECTORY.
C--INPUT/OUTPUT PARAMETERS:
C      IDRAA(4)-THE DIRECTORY TO SEARCH FOR A DL WITH LABEL='LABEL'.
C               IF FOUND, THE DL WILL BECOME THE CURRENT POSITION
C               IN THE DIRECTORY.
C--OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE FOUND DL. ZERO IF NOT FOUND.
C      IERR-ERROR CODE
```

```
C           0-NO ERROR
C
C--ALTERNATE RETURNS:
C       1-NO DL WITH THE SPECIFIED LABEL WAS FOUND. CURRENT POSITION
C         OF IDRAA IS NOT CHANGED.
C       2-IERR.NE.0
```

```
      SUBROUTINE GDRLBD(NDB,LABEL,IDRAA,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       GDRLBD WILL FIND THE DRAA OF A DR GIVEN ITS LABEL.
C       IF NON-UNIQUE DIRECTORY LABELS EXIST, GDRLBD WILL
C       RETURN THE FIRST DIRECTORY IT ENCOUNTERS WITH THE
C       SPECIFIED LABEL.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       LABEL-(CHARACTER) LABEL OF THE DR TO SEARCH FOR. IT
C            MUST EXACTLY MATCH AN EXISTING DR LABEL.
C--OUTPUT PARAMETERS:
C       IDRAA(4)-A DRAA FOR THE FOUND DR. ZERO IF NONE FOUND.
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-'LABEL' NOT FOUND
C       2-IERR.NE.0
```

```
      SUBROUTINE GETRCD(NDB,IDBAA,IPY,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           GETRCD READS RECORDS INTO THE TS AND/OR THE WS.
C           IT MANAGES THE STORAGE SPACE AND UPDATES SAF'S
C           IF NEEDED. ON RETURN, THE RECORD IS NOT LINKED
C           INTO THE DATA STORE.
C--INPUT PARAMETERS:
C           NDB-DATA BASE (1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C           IDBAA(2)- (1)-(INPUT) DBA OF THE RECORD(.LT.0 IF
C                        CHARACTER)
C                     (2)-(OUTPUT) THE DBCSA OF THE RECORD
C--OUTPUT PARAMETERS:
C           IPY-LOCATION FROM LSTORD.  IPY WILL =2,3,6,OR 7
C           IERR-ERROR CODE
```

```
C               0-NO ERROR
C               3004,4002
C
C--ALTERNATE RETURNS:
C          1-IERR.NE.0
C
```

```
      SUBROUTINE GID(NDB,IP1,IP2,IR,NDL,IDBAA,IDL,IERR,LAST,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO COPY DATA FROM AN INTEGER DL OR DR TO THE ARRAY IDL
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IP1,IP2-ELEMENTS IP1 TO IP2 WILL BE COPIED TO THE
C              ARRAY IDL. IF NDL.GT.IP2-IP1+1, THE
C              REMAINDER OF IDL IS NOT DISTURBED. IF NDL.LT.
C              IP2-IP1+1, ONLY NDL ELEMENTS WILL BE COPIED.
C              SEE IERR. IF IP2.LT.IP1 OR IP1.LE.0, RETURN
C              WITH NO ACTION.
C      IR-FORCE READ FLAG
C         1-DO NOT FORCE READ
C         2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA
C           TO BE OVERWRITTEN, IERR=7 ON RETURN.
C      NDL-LENGTH OF IDL
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DATA LIST OR DIRETORY. IT MUST HAVE
C               BEEN OBTAINED PREVIOUSLY. GID MAY CHANGE THE
C               VALUES OF IDBAA TO REFLECT CURRENT DBCS
C               ADDRESSING. (1)-DBA, (2)-DBCSA.
C--OUTPUT PARAMETERS:
C      IDL(NDL)-INTEGER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C      IERR-ERROR CODE
C           0-NO ERROR
C           4-IDBAA INVALID
C           5-IP1 AND/OR IP2 INVALID
C           6-IP2 EXTENDS PAST THE END OF THE DL OR DR
C           7-FORCE READ CAUSES DATA TO BE OVERWRITTEN.
C      LAST-LAST HAS TWO MEANINGS
C           IF IERR.NE.6, LAST IS THE LAST DL ELEMENT COPIED
C                         TO THE OUTPUT ARRAY.
C           IF IERR=6,    LAST IS THE LAST ELEMENT OF THE DL.
C                         EXISTING ELEMENTS BETWEEN IP1 AND IP2
C                         FOR WHICH THERE WAS ROOM IN THE OUTPUT
C                         ARRAY HAVE BEEN COPIED.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE GLBDLB(NDB,IDBAA,LABEL,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO RETRIEVE A LABEL TO A DL.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IDBAA(2)-DBAA OF A RECORD OF THE DL.
C--OUTPUT PARAMETERS:
C       LABEL-DL LABEL(CHARACTER).
C       IERR-ERROR CODE
C            0-NO ERROR
C            12-IDBAA NOT A DL.
C            13-DL HAS NO LABEL
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE GLBDRD(NDB,IDBAA,LABEL,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       GLBDRD WILL FIND THE LABEL OF A DR GIVEN ITS DBA
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(4)-DBAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C       LABEL-CHARACTER VARIABLE FOR THE DR LABEL(UP TO 40
C             CHARACTERS)
C       IERR-ERROR CODE
C            0-NO ERROR
C
C--ALTERNATE RETURNS:
C       1-IDBAA NOT FOUND
C       2-IERR.NE.0
```

```
      SUBROUTINE GLBELD(NDB,IRC,IL1,IL2,NLBL,IDBAA,LABEL,IG1,IG2,
     -    IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
```

```
C      GLBELD RETRIEVES LABELS OF DL ELEMENTS.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IRC-ROW/COLUMN INDICATOR
C          IRC.GT.0-ROW IRC
C          IRC.LT.0-COLUMN (-IRC)
C          IRC=0     % 1-D DATA LIST
C      IL1,IL2-ELEMENT NUMBERS
C              IRC.GT.0-THESE ARE THE FIRST AND LAST COLUMNS
C                       OF ROW IRC WHOSE LABELS ARE TO BE GOTTEN.
C              IRC.LT.0-THESE ARE THE FIRST AND LAST ROWS OF COLUMN
C                       -IRC WHOSE LABELS ARE TO BE GOTTEN.
C              IRC.EQ.0-THESE ARE THE FIRST AND LAST ELEMENTS WHOSE
C                       LABELS ARE TO BE GOTTEN.
C      NLBL-LENGTH OF OUTPUT ARRAY (LABEL)
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL WHOSE ELEMENTS LABELS ARE TO BE SET.
C--OUTPUT PARAMETERS:
C      LABEL(NLBL)-CHARACTER ARRAY FOR LABELS.
C                  IF IL2-IL1+1.GT.NLBL, ONLY NLBL LABELS WILL
C                     BE CHARGED.
C                  IF IL2-IL1+1.LT.NLBL, ONLY IL2-IL1+1 ELEMENTS OF
C                     LABEL WILL BE REFERENCED.
C      IG1,IG2-ACTUAL ELEMENTS COPIED TO LABEL.
C              IG1 AND IG2 CORRESPOND IN MEANING TO IL1 AND IL2.
C              IF IERR.EQ.0, IG1=IL1, IG2=MIN(IG1+NLBL-1,IL2)
C              IF IERR.EQ.6, IG1 AND IG2 EQUAL ACTUAL ELEMENT
C               LABELS COPIED. IF NONE, IG1=IG2=0.
C      IERR-ERROR CODE
C           0-NO ERROR
C           5,6,9,12
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE GRD(NDB,IP1,IP2,IR,NDL,IDBAA,RDL,IERR,LAST,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO COPY DATA FROM A REAL DL TO THE ARRAY RDL
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IP1,IP2-ELEMENTS IP1 TO IP2 WILL BE COPIED TO THE
C              ARRAY RDL. IF NDL.GT.IP2-IP1+1, THE
C              REMAINDER OF RDL IS NOT DISTURBED. IF NDL.LT.
C              IP2-IP1+1, ONLY NDL ELEMENTS WILL BE COPIED.
C              SEE IERR. IF IP2.LT.IP1 OR IP1.LE.0, RETURN
```

```
C              WITH NO ACTION.
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA
C            TO BE OVERWRITTEN, IERR=7 ON RETURN.
C       NDL-LENGTH OF RDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRETORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOUSLY. GRD MAY CHANGE THE
C                VALUES OF IDBAA TO REFLECT CURRENT DBCS
C                ADDRESSING. (1)-DBA, (2)-DBCSA.
C--OUTPUT PARAMETERS:
C       RDL(NDL)-REAL ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IP1 AND/OR IP2 INVALID
C            6-IP2 EXTENDS PAST THE END OF THE DL OR DR
C            7-FORCE READ CAUSES DATA TO BE OVERWRITTEN.
C       LAST-LAST HAS TWO MEANINGS
C            IF IERR.NE.6, LAST IS THE LAST DL ELEMENT COPIED
C                          TO THE OUTPUT ARRAY.
C            IF IERR=6,    LAST IS THE LAST ELEMENT OF THE DL,
C                          EXISTING ELEMENTS BETWEEN IP1 AND IP2
C                          FOR WHICH THERE WAS ROOM IN THE OUTPUT
C                          ARRAY HAVE BEEN COPIED.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE GROWCD(NDB,NROW,IC1,IC2,IR,NDL,IDBAA,CDL,IERR,LASTR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A ROW OF A 2-D DL TO THE CHAR ARRAY CDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NROW-THE ROW OF THE DL TO BE COPIED
C       IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
C               COPIED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF CDL
C               WILL NOT BE DISTURBED. IF NDL.LT.IC2-IC1+1, ONLY
C               NDL COLUMN ELEMENTS WILL BE COPIED. SEE IERR. IF
C               IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C            BE OVERWRITTEN, IERR=7 ON RETURN
```

```
C       NDL-LENGTH OF CDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       CDL(NDL)-CHARACTER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C            7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT CHARACTER
C       LASTR-IF IERR.NE.6, LASTR=0
C             IF IERR=6, LASTR=THE LAST DEFINED ROW OR COLUMN
C                        IN THE DL.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE GROWID(NDB,NROW,IC1,IC2,IR,NDL,IDBAA,IDL,IERR,LASTR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A ROW OF A 2-D DL TO THE INTEGER ARRAY IDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NROW-THE ROW OF THE DL TO BE COPIED
C       IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
C                COPIED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF IDL
C                WILL NOT BE DISTURBED. IF NDL.LT.IC2-IC1+1, ONLY
C                NDL COLUMN ELEMENTS WILL BE COPIED. SEE IERR. IF
C                IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C            BE OVERWRITTEN, IERR=7 ON RETURN
C       NDL-LENGTH OF IDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
```

```
C       IDL(NDL)-INTEGER ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C            7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT INTEGER
C       LASTR-IF IERR.NE.6, LASTR=0
C             IF IERR=6, LASTR=THE LAST DEFINED ROW OR COLUMN
C                        IN THE DL.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE GROWRD(NDB,NROW,IC1,IC2,IR,NDL,IDBAA,RDL,IERR,LASTR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM A ROW OF A 2-D DL TO THE REAL ARRAY RDL.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NROW-THE ROW OF THE DL TO BE COPIED
C       IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
C               COPIED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF RDL
C               WILL NOT BE DISTURBED. IF NDL.LT.IC2-IC1+1, ONLY
C               NDL COLUMN ELEMENTS WILL BE COPIED. SEE IERR. IF
C               IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IR-FORCE READ FLAG
C          1-DO NOT FORCE READ
C          2-FORCE READ FROM DISK. IF THIS CAUSES IN CORE DATA TO
C            BE OVERWRITTEN, IERR=7 ON RETURN
C       NDL-LENGTH OF RDL
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       RDL(NDL)-REAL ARRAY TO HOLD OUTPUT VALUES FROM THE DB.
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            6-ATTEMPT TO READ PAST END OF THE DATA LIST.
C            7-FORCE READ FLAG CAUSES DATA TO BE OVERWRITTEN.
```

```
C           9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C          10-DL NOT REAL
C      LASTR-IF IERR.NE.6, LASTR=0
C            IF IERR=6, LASTR=THE LAST DEFINED ROW OR COLUMN
C                       IN THE DL.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE GSAFD(IDBAA,SAF,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO RETRIEVE THE SAF OF A DL OR DR RECORD.
C--INPUT PARAMETERS:
C      IDBAA(2)-CURRENT DBAA OF THE DL OR DR RECORD. AN
C               ERROR RESULTS IF IDBAA IS NOT CURRENT.
C--OUTPUT PARAMETERS:
C      SAF-VALUE OF THE SAF FOR THE RECORD AT IDBAA.
C      IERR-ERROR CODE
C           0=NO ERROR
C           11-IDBAA NOT CURRENT(VALUE OF SAF NOT CHANGED)
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE INCPD(NP,NR,NS,IDBA,IENTY,KCP)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO INITIALIZE A CONTROL PART. SOME OF THE
C          ELEMENTS OF THE CONTROL PART ARE PASSED
C          TO INCPD.  OTHER ELEMENTS ARE SET TO ZERO.
C--INPUT PARAMETERS:
C          NP-PREDECESSOR RECORD
C          NR-THE RECORD NUMBER
C          NS-THE SUCCESSOR RECORD
C          IDBA-INITIAL DATA BASE ADDRESS OF THIS DL OR DIRECTORY
C          IENTY-ENTITY TYPE(1-DIRECTORY,2-IDL,3-RDL,4-CDL,5-LDL)
C--OUTPUT PARAMETERS:
C          KCP(NWCPD)-ARRAY TO HOLD THE CONTROL PART.
```

```
      SUBROUTINE INITD(IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       INITD WILL SET UP THE DBCS DEFAULT CONDITION. IT MUST BE
C       CALLED ONCE PRIOR TO CALLING ANY OTHER DBCS PROGRAMS.
C       IT MAY BE CALLED AT ANY TIME TO RESTORE THE DEFAULT SET-UP.
C       INITD WILL CLOSE ANY DB'S THAT ARE OPEN.
C
C       INITD DOES NOT RESTORE THE DEFAULTS FOR THE FOLLOWING
C       VARIABLES:
C           LUD      ITRCD
C           MISD     KEPFD
C           JOUTD    KFRTD
C           JTRCD
C
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C          0-NO ERROR
C       4001-INSUFFICIENT DATA STORE SPACE
C
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE INPD(PROGM)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           INPD INCREMENTS THE DBCS SUBROGRAM LEVEL AND
C           PRINTS THE ENTERING TRACE MESSAGE WHEN A
C           PROGRAM IS CALLED AND TRACING IS ON.
C--INPUT PARAMETERS:
C           PROGM-NAME OF THE CALLING SUBPROGRAM(CHARACTER)
C

------------------------------------------------------------------------

      ENTRY OUTPD
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           OUTPD DECREMENTS THE DBCS SUBPROGRAM LEVEL FOR
C           DBCS TRACING. OUTPD IS IN SUBROUTINE INPD.
C

------------------------------------------------------------------------
```

```
      SUBROUTINE INSDRD(NDB,IDRDL,IDBRA,ID,NID,IDBO,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       INSDRD WILL INSERT A DL OR DR INTO ITS CORRECT POSITION
C       IN A DR.  THE INSERTED ENTITY BECOMES THE CURRENT POSITION.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IDRDL-TYPE TO BE INSERTED
C             1-DR
C             2-DL
C       IDBRA(4)-DRAA(IDRDL=1) OF DBAA(IDRDL=2) OF THE ENTITY TO BE
C                INSERTED INTO THE DR.
C       ID(NID)-THE IDENTIFIER OF IDBRA
C--INPUT/OUTPUT PARAMETERS:
C       IDBO(4)-DRAA OF THE DR THAT WILL OWN IDBRA.
C               (I.E. THE DR TO INSERT IDBRA INTO.)
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C           15-IDBO NOT A DIRECTORY
C         2002-IDBRA NOT OF TYPE IDRDL
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE KPNPD(KPN,LENN,IOFF)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       GIVEN A PARTITION OF THE DATA STORE(KPN),KPNPD
C       RETURNS THE LENGTH (LENN) IN WORDS OF THE
C       RECORDS AND THE OFFSET(IN WORDS) TO THE START
C       OF THE CONTROL PART.
C--INPUT PARAMETERS:
C       KPN-A PARTITION OF THE DATA STORE. KPN IS AN
C           INDEX OF THE KNPD ARRAY.
C--OUTPUT PARAMETERS:
C       LENN-LENGTH(IN WORDS) OF THE RECORDS STORED IN
C            THE KPN SECTION OF THE DATA STORE.
C       IOFF-NUMBER OF WORDS TO THE FIRST CONTROL WORD
C            (E.G. KSTORD(KNPD(KPN)+IOFF) IS THE 1ST
C            WORD OF THE 1ST RECORD IN SECTION KPN)
C
```

```
      SUBROUTINE LNKDSD(NDB,IDBCS,IDBO)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO LINK A RECORD INTO THE DATA STORE.(NUMERIC
C          OR CHARACTER).
C--INPUT PARAMETERS:
C          NDB-DATA BASE(1-WORKING,2-REFERENCE)
C          IDBCS-DBCSA OF THE RECORD TO BE LINKED(1ST WORD OF CONTROL
C               PART)
C          IDBO-DBCSA OF THE RECORD IN THE DATA STORE TO LINK TO
C               (    ")
C          IF IDBO=0, LNKDSD SEARCHES THE DATA STORE FOR
C          AN ENTRY IN THE SAME RECORD CHAIN AS IDBCS.
C
```

```
      SUBROUTINE LOCRD(NDB,IRW,L1,L2,IR,IDBDL,I1,I2,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          LOCRD RETURNS THE DBCSA OF DL OR DR RECORDS TO BE ACCESSED.
C          IT IS CALLED SEQUENTIALLY WHEN THE ELEMENTS OF THE
C          DL OR DR SPAN MORE THAN ONE RECORD.
C--INPUT PARAMETERS:
C          NDB-DATA BASE(1-WORKING,2-REFERENCE)
C          IRW-READ/WRITE FLAG (1-LOCRD IS BEING CALLED FOR A
C              READ ACCESS, 2-LOCRD IS BEING CALLED FOR A WRITE ACCESS)
C         L1,L2-ELEMENTS REQUESTED BY THE USER. IF L2 IS BEYOND
C               THE END OF THE DL OR DR AND IRW=2, THE DL OR DR WILL BE
C               EXTENDED.
C          IR-FORCE READ FLAG(1-DO NOT FORCE READ, 2-FORCE READ
C             FROM DISK)(NOT APPLICABLE IF IRW=2)
C--INPUT/OUTPUT PARAMETERS:
C          IDBDL(2)-DBAA OF THE DL OR DR. IT WILL BE SET BY LOCRD
C                   TO THE DBAA OF THE RECORD WITH ELEMENTS I1 TO I2.
C                   IT MUST NOT BE CHANGED BETWEEN CALLS.
C                   IDBDL(1)-RECORD NUMBER(.LT.0 IF CHARACTER)
C                        (2)-DBCSA
C          I1,I2-RETURNED AS THE ELEMENTS CONTAINED IN THE RECORD AT
C                IDBDL. NO FURTHER CALLS SHOULD BE MADE WHEN I2.GE.L2
C                OR IERR=6.
C                ON THE FIRST CALL TO LOCRD, SEND I2.LT.I1. DO NOT
C                CHANGE I1 OR I2 BETWEEN CALLS.
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               6-ATTEMPT TO READ PAST END OF DL OR DR.
```

```
C                 IDBDL IS THE DBAA OF THE LAST RECORD AND I1 AND I2
C                 ARE RETURNED AS THE ELEMENTS ON THIS RECORD.
C               7-FORCE READ OF RECORDS THAT ARE OUT OF DATE ON DISK.
C                 IN CORE VERSIONS ARE OVERWRITTEN.
C--ALTERNATE RETURNS:
C          1-IERR.NE.0.AND.IERR.NE.6
C          2-IERR=6
C            IF BOTH IERR=6 AND 7 ARE APPLICABLE, IERR WILL
C            EQUAL 7 AND ALTERNATE RETURN 2 WILL BE TAKEN.

------------------------------------------------------------------------

      FUNCTION LSTORD(IDBCS)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO DETERMINE WHAT PART OF THE DATA STORE A DBCSA
C          LIES IN.  LSTORD RETURNS AN INDEX TO KNPD SUCH
C          THAT KNPD(LSTORD).LE.IDBCS.LT.KNPD(LSTORD+1)
C
C--INPUT PARAMETERS:
C          IDBCS-A DBCSA

------------------------------------------------------------------------

      FUNCTION MRAND(I1,I2)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C
C      TO GENERATE AN INTEGER UNIFORMLY DISTRIBUTED BETWEEN
C      I1 AND I2.
C
C--INPUT PARAMETERS:
C      I1,I2-LOWER AND UPPER BOUND OF THE RANDOM INTEGER
C
C--OUTPUT PARAMETERS:
C      MRAND-INTEGER BETWEEN I1 AND I2.
```

```
      FUNCTION NCPARD(IPAR,IDBCSA)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO RETURN A CONTROL PARAMETER FOR A DB RECORD IN THE
C      DATA STORE.
C      NCPARD IS INTENDED FOR INTERNAL DBCS USE ONLY. IT DOES
C     NO ERROR CHECKING FOR CORRECT INPUT PARAMETERS.
C--INPUT PARAMETERS:
C      IPAR-THE NUMBER OF THE CONTROL PARAMETER
C           1-PREDECESSOR DBA ON DISK
C           2-DBA OF THIS RECORD(MAY BE NEGATIVE)
C           3-SUCCESSOR DBA ON DISK
C           4-INITIAL DBA ON DISK
C           5-DBA OF THE LDL
C           6-INITIAL DBA OF THE OWNER DIRECTORY
C           7-ENTITY TYPE, 1-DR,2-IDL,3-RDL,4-CDL,5-LDL
C           8-FOR DL'S-THE ROW(+) OR COLUMN(-) DIMENSION.
C             +1 FOR 1-D DL'S.
C             FOR RR'S-THE ROW DIMENSION
C           9-INDEX OF THE FIRST ELEMENT ON THE RECORD.
C          10-INDEX OF THE LAST ELEMENT ON THE RECORD.
C         ---------
C          -1 ERROR
C          -2 ERROR
C          -3 SUCCESSOR DBCSA IN CORE
C          -4 PREDECESSOR DBCSA IN CORE
C          -5 DATA BASE (1 OR 2)
C          -6 LOCATION IN CSTORD OF THE DATA PART FOR
C             CHARACTER DL'S AND LDL'S.  ERROR FOR NUMERIC DL'S.
C      IDBCSA-CURRENT DBCSA OF THE DB RECORD.

-------------------------------------------------------------------------

      SUBROUTINE NEGNRD(IDBA)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO SET THE RECORD NUMBER IN CORE NEGATIVE.
C          THIS INDICATES THAT THE IN CORE RECORD
C          CONTAINS INFORMATION NOT YET WRITTEN TO DISK.
C--INPUT PARAMETERS:
C          IDBA-DBCSA OF THE RECORD

-------------------------------------------------------------------------
```

```
      SUBROUTINE NEWRCD(NDB,IDBA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           TO OBTAIN A NEW RECORD NUMBER. THE RECORD IS NOT READ
C           NOR LINKED IN ANY WAY. IF A RECORD IS CURRENTLY NOT
C           BEING USED, ITS NUMBER WILL BE RETRIEVED FROM THE
C           ARL AND USED. IF NO RECORD IS AVAILABLE, A NEW ONE WILL BE
C           CREATED. ONE RECORD ALWAYS EXISTS IN THE ARL, EVEN IF
C           IT IS EMPTY.
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING,2-REF)
C--OUTPUT PARAMETERS:
C           IDBA-DBA OF THE NEW RECORD(RECORD NUMBER)
C           IERR-ERROR CODE
C           2000-NO MORE RECORDS AVAILABLE
C--ALTERNATE RETURNS:
C           1-IERR.NE.0
```

```
      FUNCTION NRD(IOPT,NR,ITY)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           DBCSA'S HAVE NEGATIVE RECORD NUMBERS FOR DATA LISTS
C           OF TYPE CHARACTER. NRD WILL RETURN THE APPROPRIATE
C           (+ OR -) VALUE OF THE RECORD NUMBER.
C
C--INPUT PARAMETERS:
C           IOPT-OPTION
C                1-RETURN THE CORRECT SIGN OF THE RECORD NUMBER(SEE
C                  NRD)
C                2-RETURN IABS(NR) (ITY NOT USED)
C           NR-DBCSA (IE RECORD NUMBER) NR MAY BE + OR -.
C           ITY-TYPE
C               1-NUMERIC,  2-CHARACTER
C--OUTPUT PARAMETERS:
C           NRD-FOR IOPT=1, NRD=IABS(NR) IF ITY=1
C                           =-IABS(NR) IF ITY=2
C               FOR IOPT=2, NRD=IABS(NR)
```

```
      SUBROUTINE NXD(NDB,IOPT,IBRDL,IDIR,ICOND,NCOND,NIB,IDRAA,
     -ID,IFOUND,NTY,IERR,*,*)
C--MODULE: MOPADS/BBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       NXD MOVES THE CURRENT DL OR DR POSITION OF A DIRECTORY
C       TO A DL OR DR THAT SATISFIES SPECIFIED CONDITIONS.
C       IF NO DL OR DR SATISFYING THE CONDITIONS IS FOUND OR IF
C       AN ERROR OCCURS, THE CURRENT POSITION IS NOT CHANGED.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IOPT-OPTION
C            1-RETURN THE CURRENT POSITION IF IT SATISFIES THE
C              CONDITION.
C            2-THE CURRENT POSITION IS NOT ELIGIBLE TO BE FOUND.
C       IBRDL-TYPE TO SEARCH OVER
C             1-DR'S
C             2-DL'S
C       IDIR-DIRECTION OF SEARCH
C            1-FORWARD(AWAY FROM ONE)
C            2-BACKWARD(TOWARD ONE, I.E. TOWARD THE BEGINNING OF
C              THE LIST)
C      ICOND(3,NCOND)-CONDITIONS TO SATISFY. 'NCOND' CONDITIONS EXIST.
C                     ICOND(1,J) IS THE ELEMENT OF THE ID TO WHICH
C                     CONDITION ICOND(2,J) APPLIES.  ICOND(1,J)=0
C                     MEANS CONDITION J IS NULL.
C
C          IF ICOND(2,J) IS      THEN THE CONDITION IS
C          ----------------      -------------------------------------
C             .LE.0              NULL(ANY VALUE IS OK)
C                1               ID ELEMENT ICOND(1,J) .LT. ICOND(3,J)
C                2                 "                "  .LE.  "   "
C                3                 "                "  .EQ.  "   "
C                4                 "                "  .GE.  "   "
C                5                 "                "  .GT.  "   "
C                6                 "                "  .NE.  "   "
C
C                       E.G. ICOND(-,2)=(4,3,12) MEANS ID ELEMENT 4
C                            MUST EQUAL 12.
C                            ICOND(-,3)=(5,0,0) MEANS ID ELEMENT 5
C                            MAY HAVE ANY VALUE.
C
C                       IF NCOND=0, ICOND WILL NOT BE USED. NXD WILL
C                       SEARCH IN THE DIRECTION "IDIR" FOR THE NEXT
C                       ENTRY OF TYPE "IBRDL" AND RETURN IT. IF IOPT=1
C                       NXD WILL RETURN THE CURRENT POSITION(IF THERE IS
C                       NO CURRENT POSITION, ALTERNATE RETURN 1 IS
C                       TAKEN.)
C
```

```
C                     SEARCHING MAY BE PERFORMED ON THE ENTITY TYPE
C                     AND THE DBA AS WELL.  LET 'LID' BE THE LENGTH
C                     OF THE ID'S.  THEN ID 'ELEMENT' LID+1
C                     IS THE TYPE (1-DIRECTORY,2-IDL,3-RDL,4-CDL).
C                     ELEMENT LID+2 IS THE DBA OF THE DL OR DR.
C                     TO SEARCH ON THESE USE ICOND(J,1)=LID+1 OR
C                     LID+2.
C
C     NID-AT LEAST THE LENGTH OF THE ID'S OF THE DIRECTORY AT IDRAA.
C--INPUT/OUTPUT PARAMETERS:
C     IDRAA(4)-DRAA OF THE DIRECTORY TO SEARCH
C--OUTPUT PARAMETERS:
C     ID(NID)-IDENTIFIER FOUND BY THE SEARCH. ZERO IF ALTERNATE
C             RETURN 1 IS TAKEN. IF NID IS GREATER THAN THE
C             LENGTH OF THE ID'S, THE REMAINDER IS NOT CHANGED.
C     IFOUND(4)-DBAA OR DRAA OF THE FOUND ENTRY. IF IDRDL=2,
C               ONLY THE FIRST 2 WORDS OF IFOUND ARE USED.
C               IFOUND=0 IF ALTERNATE RETURN 1 IS TAKEN.
C     MTY-COLUMN POSITION OF THE LAST EMPTY COLUMN SEEN
C         BEFORE RETURN. ZERO IF NONE.
C     IERR
C          0-NO ERROR
C          9-NID TOO SHORT
C         15-IDRAA NOT A DIRECTORY
C         16-ICOND NOT ACCEPTABLE
C--ALTERNATE RETURNS:
C     1-NO DL OR DR SATISFYING THE CONDITIONS WAS FOUND
C     2-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE OPNDBD(NDB,ION,FILN,MAXR,NID,IERR,IDERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C     OPNDBD WILL OPEN A DB FILE. THE DB IS OPENED IN READ-ONLY
C     MODE
C--INPUT PARAMETERS:
C     NDB-DATA BASE(1-WORKING,2-REFERENCE)
C         OPNDBD WILL CLOSE AN EXISTING DB OF TYPE NDB BEFORE
C         OPENING THE NEW FILE.
C     ION-OLD/NEW OPTION
C         1-CREATE A NEW DB ON THE SPECIFIED FILE.
C         2-OPEN A PREVIOUSLY EXISTING DB
C     FILN-FILE NAME(CHARACTER)
C     MAXR-MAX RECORD NUMBER FOR THE FILE
C          .LE.0-USE NO LIMIT IF ION=1. IF ION=2, USE PREVIOUS
C                VALUE.
C          .GT.0-USE MAXR IF ION=1. IF ION=2, USE THE MAXIMUM OF
```

```
C                MAXR AND THE PREVIOUS VALUE.
C      NID-THE NUMBER OF WORDS/ID FOR THE MASTER DIRECTORY.
C          USED ONLY IF ION=1. (NID.GE.1)
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C        2003-NO PRIOR CALL TO INITD
C        3000-BOTH DB'S(WORKING AND REFERENCE) MUST HAVE SAME
C             RECORD SIZE. THIS ERROR OCCURS IF A DB IS OPENED
C             WITH A RECORD SIZE THAT DOES NOT MATCH THE OTHER
C             OPEN DB.
C        3001-NEW DB DOES NOT HAVE THE CORRECT CONTROL PART SIZE.
C        3003-NEW DB DOES NOT HAVE CORRECT DIRECTORY ADDRESS SIZE.
C        3004-SYSTEM I/O ERROR OPENING DB OR ACCESSING PARAMETERS.
C             INTEGER PARAMETER 1 IS THE I/O ERROR CODE. SEE IOERR.
C
C             NOTE: ERRORS 3001 AND 3003 OCCUR ONLY IF MODIFICATIONS
C                   ARE MADE TO DBCS AND A DB CREATED WITH THE EARLIER
C                   VERSION ARE RUN WITH THE NEW VERSION.
C      IOERR-SYSTEM I/O ERROR CODE IF IERR=3004. ZERO OTHERWISE.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0

-------------------------------------------------------------------------

      SUBROUTINE PCD(NDB,IP1,IW,CDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO COPY THE CONTENTS OF THE CHARACTER ARRAY CDL TO A DATA LIST
C      OR DIRECTORY.
C--INPUT PARAMETERS:
C      NDB-DATA BASE (1-WORKING,2-REFERENCE)
C      IP1-THE FIRST ELEMENT OF THE DL OR DR TO BE
C          OVERWRITTEN WITH DATA FROM CDL.  IF IP1 IS
C          BEYOND THE CURRENT END OF THE DL, THE DL WILL
C          BE EXTENDED (WITH ZERO VALUES FOR UNSPECIFIED
C          ELEMENTS).  IF IP1.LE.0, RETURN WITH NO ACTION.
C      IW-FORCE WRITE FLAG
C         1-DO NOT FORCE WRITE.  CHANGED DL OR DR MAY EXIST
C           ONLY IN CORE.
C         2-FORCE WRITE THE CHANGES TO THE DBF.
C         THIS OPTION IS IN ADDITION TO THE DB PROTECTION MODE.
C      CDL(NDL)-THE CHARACTER ARRAY WITH VALUES TO BE WRITTEN TO THE
C               DATA BASE. NDL ELEMENTS WILL BE COPIED FROM CDL
C               TO THE DATA BASE.
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL OR DR. IDBAA MAY BE CHANGED BY
C               PCD TO REFLECT CURRENT DBCS ADDRESSING.
```

```
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE. IF IERR.NE.0, THE CONTENTS OF THE
C            IDBAA DL OR DR ARE UNPREDICTABLE.
C            0-NO ERROR
C           1,2-DB NOT OPEN
C           4-INVALID IDBAA
C           8-INVALID WRITE TO READ ONLY DB
C           2000-NO MORE RECORDS AVAILABE IN DBF.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PCOLCD(NDB,NCOL,IR1,IR2,IW,CDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 3.13
C--PURPOSE:
C       TO COPY DATA FROM THE CHARACTER ARRAY CDL TO A 2-D DL IN
C       THE DATA BASE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               CHANGED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF CDL
C               WILL NOT BE USED. IF NDL.LT.IR2-IR1+1, ONLY
C               NDL ROW ELEMENTS WILL BE CHANGED. SEE IERR. IF
C               IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       CDL(NDL)-CHARACTER ARRAY WITH VALUES TO BE WRITTEN TO THE DL.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOUSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IR1 AND/OR IR2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT CHARACTER
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PCOLID(NDB,NCOL,IR1,IR2,IW,IDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM THE INTEGER ARRAY IDL TO A 2-D DL IN
C       THE DATA BASE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               CHANGED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF IDL
C               WILL NOT BE USED. IF NDL.LT.IR2-IR1+1, ONLY
C               NDL ROW ELEMENTS WILL BE CHANGED. SEE IERR. IF
C               IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       IDL(NDL)-INTEGER ARRAY WITH VALUES TO BE WRITTEN TO THE DL.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IR1 AND/OR IR2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT INTEGER
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PCOLRD(NDB,NCOL,IR1,IR2,IW,RDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM THE REAL ARRAY RDL TO A 2-D DL IN
C       THE DATA BASE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
```

```
C       NCOL-THE COLUMN OF THE DL TO BE COPIED
C       IR1,IR2-ROW ELEMENTS IR1 TO IR2 OF COLUMN NCOL WILL BE
C               CHANGED. IF NDL.GT.IR2-IR1+1, THE REMAINDER OF RDL
C               WILL NOT BE USED. IF NDL.LT.IR2-IR1+1, ONLY
C               NDL ROW ELEMENTS WILL BE CHANGED. SEE IERR. IF
C               IR2.LT.IR1 OR IR1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       RDL(NDL)-REAL ARRAY WITH VALUES TO BE WRITTEN TO THE DL.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IR1 AND/OR IR2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT REAL
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PID(NDB,IP1,IW,IDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY THE CONTENTS OF THE ARRAY IDL TO AN INTEGER DATA LIST
C       OR DIRECTORY.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       IP1-THE FIRST ELEMENT OF THE DL OR DR TO BE
C           OVERWRITTEN WITH DATA FROM IDL.  IF IP1 IS
C           BEYOND THE CURRENT END OF THE DL, THE DL WILL
C           BE EXTENDED (WITH ZERO VALUES FOR UNSPECIFIED
C           ELEMENTS).  IF IP1.LE.0, RETURN WITH NO ACTION.
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE.  CHANGED DL OR DR MAY EXIST
C            ONLY IN CORE.
C          2-FORCE WRITE THE CHANGES TO THE DBF.
C          THIS OPTION IS IN ADDITION TO THE DB PROTECTION MODE.
C       IDL(NDL)-THE INTEGER ARRAY WITH VALUES TO BE WRITTEN TO THE
```

```
C               DATA BASE. NDL ELEMENTS WILL BE COPIED FROM IDL
C               TO THE DATA BASE.
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL OR DR. IDBAA MAY BE CHANGED BY
C               PID TO REFLECT CURRENT DBCS ADDRESSING.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE. IF IERR.NE.0, THE CONTENTS OF THE
C           IDBAA DL OR DR ARE UNPREDICTABLE.
C           0-NO ERROR
C          1,2-DB NOT OPEN
C          4-INVALID IDBAA
C          8-INVALID WRITE TO READ ONLY DB
C          2000-NO MORE RECORDS AVAILABE IN DBF.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE PLBDLD(NDB,IDBAA,LABEL,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO ASSIGN A LABEL TO A DL. PLBDLD WILL OVERWRITE A
C      PREVIOUS DL LABEL.
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IDBAA(2)-DBAA OF A RECORD OF THE DL.
C      LABEL-NEW DL LABEL(CHARACTER). DL LABELS WILL BE
C            TRUNCATED TO 25 CHARACTERS.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           12-IDBAA NOT A DL.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE PLBELD(NDB,IRC,IL1,IL2,LABEL,NLBL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      PLBELD SETS LABELS OF DL ELEMENTS.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
```

```
C       IRC-ROW/COLUMN INDILATOR
C           IRC.GT.0-ROW IRC
C           IRC.LT.0-COLUMN (-IRC)
C           IRC=0   -A 1-R DATA LIST
C       IL1,IL2-ELEMENT NUMBERS
C                IRC.GT.0-THESE ARE THE FIRST AND LAST COLUMNS
C                         OF ROW IRC WHOSE LABELS ARE TO BE CHANGED.
C                IRC.LT 0-THESE ARE THE FIRST AND LAST ROWS OF COLUMN
C                         -IRC WHOSE LABELS ARE TO BE CHANGED.
C                IRC.EQ.0-THESE ARE THE FIRST AND LAST ELEMENTS WHOSE
C                         LABELS ARE TO BE CHANGED.
C       LABEL(NLBL)-CHARACTER ARRAY CONTAINING LABELS.
C                    IF IL2-IL1+1.GT.NLBL, ONLY NLBL LABELS WILL
C                       BE CHABGED.
C                    IF IL2-IL1+1.LT.NLBL, ONLY IL2-IL1+1 ELEMENTS OF
C                       LABEL WILL BE REFERENCED.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DL WHOSE ELEMENTS LABELS ARE TO BE SET.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            5,9.12
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PLINKD(NDB,IDRAA,IDOAA,NID,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       PLINKD WILL RETURN THE DRAA OF THE OWNER OF A DL OR DR.
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      IDRAA(4)-DRAA(DIRECTORY) OR DBAA(DL) OF THE DR OR DL WHOSE
C               OWNER IS TO BE FOUND. IF IT IS A DL, IDRAA NEED BE
C               DIMENSIONED ONLY TO 2.
C--OUTPUT PARAMETERS:
C       IDOAA(4)-THE DRAA OF THE OWNER DIRECTORY. IF IDRAA HAS NO
C                OWNER, IDOAA(1)=0.
C       NID-NUMBER OF WORDS IN THE ID OF IDRAA
C       IERR-ERROR CODE
C            0-NO ERROR
C
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE POSDRD(NDB,IBE,IDRDL,IDRAA,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       POSDRD WILL POSITION A DR AT ITS BEGINNING OR END(FOR
C       OWNED DL'S OR DR'S)
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IBE-DIRECTION
C           1-POSITION AT BEGINNING
C           2-POSITION AT END
C       IDRDL-TYPE OF SEARCH
C             1-DR'S
C             2-DL'S
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-IDRAA HAS NO ENTRIES OF TYPE IDRDL
C       2-IERR.NE.0
```

```
      SUBROUTINE PRD(NDB,IP1,IW,RDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY THE CONTENTS OF THE ARRAY RDL TO A REAL DATA LIST
C       OR DIRECTORY.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       IP1-THE FIRST ELEMENT OF THE DL OR DR TO BE
C           OVERWRITTEN WITH DATA FROM RDL.  IF IP1 IS
C           BEYOND THE CURRENT END OF THE DL, THE DL WILL
C           BE EXTENDED (WITH ZERO VALUES FOR UNSPECIFIED
C           ELEMENTS).  IF IP1.LE.0, RETURN WITH NO ACTION.
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE.  CHANGED DL OR DR MAY EXIST
C            ONLY IN CORE.
C          2-FORCE WRITE THE CHANGES TO THE DBF.
C          THIS OPTION IS IN ADDITION TO THE DB PROTECTION MODE.
C       RDL(NDL)-THE REAL ARRAY WITH VALUES TO BE WRITTEN TO THE
C                DATA BASE. NDL ELEMENTS WILL BE COPIED FROM RDL
C                TO THE DATA BASE.
```

```
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL OR DR. IDBAA MAY BE CHANGED BY
C               PRD TO REFLECT CURRENT DBCS ADDRESSING.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE. IF IERR.NE.0, THE CONTENTS OF THE
C           IDBAA DL OR DR ARE UNPREDICTABLE.
C           0-NO ERROR
C          1,2-DB NOT OPEN
C          4-INVALID IDBAA
C          8-INVALID WRITE TO READ ONLY DB
C          2000-NO MORE RECORDS AVAILABE IN DBF.
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
------------------------------------------------------------------------

      SUBROUTINE PROTD(NDB,KPROT,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO SET THE PROTECTION CODE FOR THE DB
C
C--INPUT PARAMETERS:
C          NDB-DB(1-WORKING, 2-REF)
C          KPROT-PROTECTION CODE
C                1-READ ONLY
C                2-WRITE(SAFE)
C                3-WRITE(LOOSE)
C                4-WRITE(UNSAFE)
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               1,2
C--ALTERNATE RETURNS:
C          1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE PROWCD(NDB,NROW,IC1,IC2,IW,CDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO COPY DATA FROM THE CHARACTER ARRAY CDL TO A 2-D DL IN
C      THE DATA BASE.
C--INPUT PARAMETERS:
C      NDB-DATA BASE (1-WORKING,2-REFERENCE)
C      NROW-THE ROW OF THE DL TO BE COPIED
C      IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
```

```
C               CHANGED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF CDL
C               WILL NOT BE USED. IF NDL.LT.IC2-IC1+1, ONLY
C               NDL COLUMN ELEMENTS WILL BE CHANGED. SEE IERR. IF
C               IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       CDL(NDL)-CHARACTER ARRAY WITH VALUES TO BE WRITTEN TO THE DB.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT CHARACTER
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE PROWID(NDB,NROW,IC1,IC2,IW,IDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM THE INTEGER ARRAY IDL TO A 2-D DL IN
C       THE DATA BASE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NROW-THE ROW OF THE DL TO BE COPIED
C       IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
C               CHANGED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF IDL
C               WILL NOT BE USED. IF NDL.LT.IC2-IC1+1, ONLY
C               NDL COLUMN ELEMENTS WILL BE CHANGED. SEE IERR. IF
C               IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       IDL(NDL)-INTEGER ARRAY WITH VALUES TO BE WRITTEN TO THE DB.
C--INPUT/OUTPUT PARAMETERS:
```

```
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
C           10-DL NOT INTEGER
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C

-----------------------------------------------------------------------

      SUBROUTINE PROWRD(NDB,NROW,IC1,IC2,IW,RDL,NDL,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO COPY DATA FROM THE REAL ARRAY RDL TO A 2-D DL IN
C       THE DATA BASE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       NROW-THE ROW OF THE DL TO BE COPIED
C       IC1,IC2-COLUMN ELEMENTS IC1 TO IC2 OF ROW NROW WILL BE
C                CHANGED. IF NDL.GT.IC2-IC1+1, THE REMAINDER OF RDL
C                WILL NOT BE USED. IF NDL.LT.IC2-IC1+1, ONLY
C                NDL COLUMN ELEMENTS WILL BE CHANGED. SEE IERR. IF
C                IC2.LT.IC1 OR IC1.LE.0, RETURN WITH NO ACTION
C       IW-FORCE WRITE FLAG
C          1-DO NOT FORCE WRITE. DL CHANGED IN CORE ONLY.
C          2-FORCE WRITE THE CHANGES TO DISK. THIS IS IN ADDITION TO THE
C            DB PROTECTION MODE.
C       RDL(NDL)-REAL ARRAY WITH VALUES TO BE WRITTEN TO THE DB.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DATA LIST OR DIRECTORY. IT MUST HAVE
C                BEEN OBTAINED PREVIOSLY. IT MAY BE CHANGED IN THIS
C                SUBROUTINE TO REFLECT CURRENT DBCS ADDRESSING.
C                (1)-DBA,(2)-DBCSA
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            4-IDBAA INVALID
C            5-IC1 AND/OR IC2 INVALID
C            8-ATTEMPT TO WRITE TO READ ONLY DB
C            9-ATTEMPT TO USE INCORRECT DIMENSIONS TO ACCESS DB
```

```
C          10-DL NOT REAL
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C
```

```
      SUBROUTINE PRTDLD(NDB,MSG,IL1,IL2,IHEAD,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      PRTDLD PRINTS THE CONTENTS OF A DATA LIST ON THE DBCS OUTPUT
C      FILE.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      MSG-(CHARACTER) MESSAGE TO BE PRINTED WITH THE DL.(NOT PRINTED
C          IF BLANK)
C      IL1,IL2-FIRST AND LAST ELEMENTS TO PRINT.
C              IL1=0 STARTS AT FIRST ELEMENT.
C              IL2=0 MEANS PRINT TO END
C              FOR 1-D DL'S, IL1 AND IL2 ARE ELEMENTS
C              FOR 2-D COLUMN ORIENTED DL'S, IL1 AND IL2 ARE COLUMN
C                 NUMBERS.
C              FOR 2-D ROW ORIENTED DL'S, IL1 AND IL2 ARE ROW NUMBERS
C      IHEAD-HEADER OPTION
C            1-YES
C            2-NO
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(2)-DBAA OF THE DL
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C          12-IDBAA NOT A DL
C        4010-DL NOT IN SPECIFIED DR
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE PRTDRD(NDB,IOPT,MSG,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      PRTDRD WILL PRINT THE CONTENTS OF A DIRECTORY ON THE DBCS OUTPUT
C      FILE.
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
```

```
C      IOPT-OPTION
C           1-PRINT OWNED DIRECTORIES
C           2-PRINT OWNED DATA LISTS
C           0=PRINT BOTH
C      MSG-(CHARACTER) MESSAGE TO BE PRINTED(NONE IF BLANK)
C--INPUT/OUTPUT PARAMETERS:
C      IDBAA(4)-DBAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C          15-IDBAA NOT A DIRECTORY
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE PSAFD(IDBAA,SAF,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO SET THE SAF OF A DL OR DR RECORD TO A SPECIFIED VALUE.
C--INPUT PARAMETERS:
C      IDBAA(2)-CURRENT DBAA OF THE DL OR DR RECORD. AN
C               ERROR RESULTS IF IDBAA IS NOT CURRENT.
C      SAF-VALUE OF THE SAF FOR THE RECORD AT IDBAA.
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0=NO ERROR
C           11-IDBAA NOT CURRENT(VALUE OF SAF NOT CHANGED)
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE PSTORD
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      PSTORD PRINTS THE CONTENTS OF THE DBCS DATA STORE TO THE
C      DBCS OUTPUT FILE.
C
```

```
      SUBROUTINE RCD(NDB,IDBA,ICP,DP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           TO READ A CHARACTER RECORD
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING, 2-REF)
C           IDBA-DBA(RECORD NUMBER)
C--OUTPUT PARAMETERS:
C           ICP(NWCPD)-ARRAY FOR CONTROL PART OF THE RECORD
C           DP(NCDPD)-CHARACTER VARIABLE FOR THE DATA PART OF THE RECORD
C           IERR-ERROR CODE
C                0-NO ERROR
C              1,2,3004,4002,4004,4005
C--ALTERNATE RETURNS:
C           1-IERR.NE.0
```

```
      SUBROUTINE RD(NDB,ITYP,NR,IDBAR,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           RD READS A RECORD FROM THE DB
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING,2-REF)
C           ITYP-RECORD TYPE(1-NUMERIC,2-CHARACTER)
C           NR-RECORD NUMBER
C           IDBAR(2)-DBCSA'S OF CONTROL AND DATA PARTS,RESPECTIVELY
C--OUTPUT PARAMETERS:
C           IERR-ERROR CODE
C                0-NO ERROR
C--ALTERNATE RETURNS:
C           1-IERR.NE.0
```

```
      SUBROUTINE RETCHD(NDB,IOPT,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO RETURN A CHAIN OF DISK RECORDS TO THE ARL
C       STARTING AT IDBAA.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
```

```
C       IOPT-OPTION
C            0-RETURN ENTIRE CHAIN
C            1-RETURN ALL RECORDS PRIOR TO IDBAA
C            2-RETURN ALL RECORDS AFTER IDBAA
C       IDBAA(2)-DBAA OF A RECORD IN THE CHAIN. THE RECORD MUST
C                EXIST IN THE DATA STORE AND IDBAA MUST BE CURRENT
C                (IDBAA(2) CORRECT). IDBAA IS UNCHANGED ON
C                RETURN.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            1,2,PLUS OTHERS
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE RETRCD(NDB,IOPT,IDBA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO RETURN A RECORD TO THE ARL.  RETRCD PUTS THE
C          RECORD IN THE ARL. IT WILL OPTIONALLY UNLINK
C          THE DATA STORE SPACE AND MAKE IT AVAILABLE, AND UNLINK
C          THE RECORD ON DISK.  FOR UNLINKING,THE RECORD MUST
C          EXIST IN THE DATA STORE.
C--INPUT PARAMETERS:
C          NDB-DB(1-WORKING,2-REF)
C          IOPT-OPTION
C               1-RETURN RECORD IDBA(1) TO THE ARL ONLY
C               2-RETURN THE RECORD TO THE ARL AND UNLINK ON
C                 DISK AND IN DATA STORE.(IF IT EXIST IN THE STORE)
C          IDBA(2)-THE CURRENT DBAA OF THE RECORD ON INPUT. IDBA(2)
C                       IS NOT USED IF IOPT=1.
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               8
C            3006
C--ALTERNATE RETURNS:
C          1-IERR.NE.0
```

```
      SUBROUTINE RID(NDB,IDBA,ICP,IDP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
```

```
C--PURPOSE:
C          TO READ AN INTEGER RECORD FROM A DB
C--INPUT PARAMETERS:
C          NDB-DB(1-WORKING,  2-REF)
C          IDBA-DBA(RECORD NUMBER)
C--OUTPUT PARAMETERS:
C          ICP(NWCPD)-ARRAY FOR CONTROL PART OF RECORD
C          IDP(NWDPD)-ARRAY FOR THE DATA PART OF THE RECORD
C          IERR-ERROR CODE
C               0-NO ERROR
C               1,2,3004,4002,4004,4005
C--ALTERNATE RETURNS:
C          1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE RKDIRD(IOPT,KODE8,KRKDL,KRKDR,KODE5)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      RKDIRD COMPUTES THE RANKING CODE FOR CONTROL WORD
C      5 OF DIRECTORIES FROM THE DIRECTORY AND DATA LIST RANKING
C      CODES OR VICE VERSA.
C--INPUT PARAMETERS:
C      IOPT-OPTION
C           1-COMPUTE CONTROL WORD 5 FROM THE RANKING
C             CODES(INPUT-KRKDL,KRKDR,OUTPUT-KODE5)
C           2-COMPUTE INDIVIDUAL RANKING CODES FROM
C             CONTROL WORD 5(INPUT-KODE5,OUTPUT-KRKDL AND KRKDR)
C      KODE8-VALUE OF CONTROL WORD 8 OF THE DIRECTORY
C--INPUT/OUTPUT PARAMETERS:
C      KRKDL-THE INDIVIDUAL RANKING CODE FOR OWNED DATA LISTS
C      KRKDR-THE INDIVIDUAL RANKING CODE FOR OWNED DIRECTORIES.
C      KODE5-CONTROL WORD FIVE FOR DIRECTORIES
C

------------------------------------------------------------------------

      SUBROUTINE RNDRD(NDB,LABEL,IDRAA,IERR,*,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      RNDRD WILL RENAME A DIRECTORY
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      LABEL-THE NEW LABEL FOR THE DIRECTORY(CHARACTER).
C            DIRECTORY LABELS MAY HAVE 40 CHARACTERS.
C            IF LABEL='  ', '(UNLABELED)' WILL BE USED.
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IDRAA(4)-DRAA OF THE DIRECTORY
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-IDRAA NOT FOUND
C       2-IERR.NE.0
```

```
      SUBROUTINE RRD(NDB,IDBA,ICP,DP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           TO READ A REAL RECORD FROM A DB
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING,  2-REF)
C           IDBA-DBA(RECORD NUMBER)
C--OUTPUT PARAMETERS:
C           ICP(NWCPD)-ARRAY FOR CONTROL PART OF RECORD
C           DP(NWDPD)-ARRAY FOR THE DATA PART OF THE RECORD
C           IERR-ERROR CODE
C                0-NO ERROR
C                1,2,3004,4002,4004,4005
C--ALTERNATE RETURNS:
C           1-IERR.NE.0
```

```
      SUBROUTINE RWRCD(IRW,IFMT,IFILE,INC,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       RWRCD WILL READ/WRITE A RECORD FROM/TO A SEQUENTIAL INTERNAL
C       FORMAT OR A SEQUENTIAL, EXTERNAL FORMAT FILE.
C
C--INPUT PARAMETERS:
C       IRW-READ/WRITE FLAG
C           1-READ
C           2-WRITE
C       IFMT-FILE FORMAT
C            1-SEQUENTIAL INTERNAL
C            2-SEQUENTIAL EXTERNAL
C       IFILE-LOGICAL UNIT NUMBER OF THE FILE
C--INPUT/OUTPUT PARAMETERS:
C       INC-TYPE(1-NUMERIC,2-CHARACTER)
C                   IRW=1 -ON OUTPUT, INC IS THE COLUMN NUMBER OF IDBAA
C                          THAT CONTAINS THE DBAA OF THE RECORD.
C                   IRW=2 -ON INPUT, INC IS THE COLUMN NUMBER OF IDBAA
C                          THAT CONTAINS THE RECORD.
```

```
C       IDBAA(2,2)-CURRENT DBAA FOR THE RECORD. COLUMN 1 CONTAINS
C                  A NUMERIC DBCSA AND COLUMN 2 CONTAINS A CHARACTER
C                  DBCSA.
C                IRW=1-ON INPUT IDBAA(2,-) IS THE DBCSA FOR THE RECORD.
C                     ON OUTPUT, IDBAA(1,-) IS THE CORRECT RECORD
C                     NUMBER. IN OTHER WORDS, RWRCD READS THE NEXT
C                     RECORD ON THE FILE INTO CORE SPACE AT IDBAA(2,-)
C                     AND RETURNS THE RECORD NUMBER OF THE RECORD IN
C                     IDBAA(1,-).
C                IRW=2-IDBAA IS NOT CHANGED BY RWRCD
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            11-IDBAA NOT CURRENT FOR IRW=2
C            3004-SYSTEM ERROR ON IFILE.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE R1D(NDB,NCREC,NWCP,NCDP,MXR,MAXR,ALPH,MLDLA,
     -               MHDLA,TDLA,NWAD,KARL1,IERR,IOERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPUSE:
C           TO READ RECORD 1 OF A DBF
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING,  2-REF)
C--OUTPUT PARAMETERS:
C           NCREC-NWRECD(1)
C           NWCP-NWCPD
C           NCDP-NCDPD
C           MXR-MXRD(1,NDB)
C           MAXRD-MXRD(2,NDB)
C           ALPH-ALPHD
C           MLDLA-MLDLAD
C           MHDLA-MHDLAD
C           TDLA-TDLAD
C           NWAD-NWADD
C           KARL1-KARLD(1,ND)
C           IERR-ERROR CODE
C                0-NO ERROR
C                3004
C           IOERR-SYSTEM ERROR CODE
C--ALTERNATE RETURNS:
C           1-IERR.NE.0
```

```
      SUBROUTINE SAFD(SAFOLD,TSLU,ACCN,SAFNEW)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          SAFD COMPUTES THE NEW ESTIMATE OF THE SAF FROM THE OLD.
C--INPUT PARAMETERS:
C          SAFOLD-THE LAST ESTIMATE OF SAF
C          TSLU-THE VALUE OF TDLAD WHEN SAFOLD WAS COMPUTED.
C          ACCN-NUMBER OF ACCESSES OF THIS RECORD SINCE SAFOLD
C               WAS COMPUTED.
C--OUTPUT PARAMETERS:
C          SAFNEW-THE NEW ESTIMATE OF SAF
```

```
      SUBROUTINE SAF1D(IOPT)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          SAF1D PERFORMS AN IN-CORE UPDATE ON THE SAF DATA
C          IN THE WS AND TS. SAF1D DOES NOT SCHEDULE THE
C          NEXT PERIODIC UPDATE.
C--INPUT PARAMETERS:
C          IOPT-OPTION
C               1-DO NOT SWAP RECORDS FROM THE TS TO THE WS TO GET
C                 HIGHEST SAF'S IN WS.
C               2-AFTER UPDATE, SWAP RECORDS TO WS IF NEEDED TO GET
C                 HIGHEST SAF'S IN WS(WORKING DATA BASE ONLY).
```

```
      SUBROUTINE SAF2D(NDB,IDBCS,ITYP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          SAF2D READS A RECORD AND UPDATES THE SAF DATA.
C          SAF2D DOES NOT CHANGE THE IN-CORE LINKING DATA.
C--INPUT PARAMETERS:
C          NDB-DATA BASE(1-WORKING,2-REFERENCE)
C          IDBCS(2)-DBAA OF THE RECORD.
C          ITYP-DL TYPE
C               1-NUMERIC
C               2-CHARACTER
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               3004,4002
C--ALTERNATE RETURNS:
C          1-IERR.NE.0
```

```
      SUBROUTINE SAF3D(NDB,IOPT,IDBC,ITYP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          TO UPDATE THE SAF OF A DL OR DR RECORD PRIOR TO WRITING
C          IT TO DISK. SAF3D WRITES THE RECORD, AND OPTIONALLY DELINKS
C          IT FROM THE IN-CORE DATA STORE AND MAKES THE SPACE AVAILABLE
C          IN THE TS OR US.
C--INPUT PARAMETERS:
C          NDB-DATA BASE(1-WORKING,2-REFERENCE)
C          IOPT-OPTION
C               1-UPDATE AND WRITE TO DISK
C               2-UPDATE,WRITE TO DISK, AND DELINK RECORD
C          IDBC-DBCSA OF THE RECORD TO BE WRITTEN.
C          ITYP-DL TYPE
C               1-NUMERIC
C               2-CHARACTER
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C               3004
C--ALTERNATE RETURNS:
C          1-IERR.NE.0
```

```
      SUBROUTINE SETDSD(NNTSD,NNCTSD,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      SETDSD SETS THE BREAK POINTS OF THE DATA STORE
C
C--INPUT PARAMETERS:
C      NNTSD-NUMBER OF RECORDS FOR THE NUMERIC TS(IF .LE. 0, USE
C            THE CURRENT VALUE)
C      NNCTSD-NUMBER OF RECORDS FOR THE CHARACTER TS(IF .LE.0, USE
C             THE CURRENT VALUE)
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C        4001-INSUFFICIENT DATA STORE SPACE
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
```

```
      SUBROUTINE SHFDRD(NDB,IC1,IC2,NC,IDIR,IDBO,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       SHFDRD IS A UTILITY PROGRAM WHICH WILL SHIFT COLUMNS OF
C       A DIRECTORY TO OPEN UP SPACE FOR INSERTION. IT WILL
C       SHIFT COLUMNS IC1 THRU IC2 NC COLUMNS IN THE DIRECTION
C       IDIR. NOTE: NO CHECK IS MADE TO INSURE THAT DATA IS NOT
C       DESTROYED BY THE OPERATION. THE CURRENT POSITIONS OF THE
C       DIRECTORY ARE UPDATED IF NECESSARY. SHFDRD IS CALLED ONLY
C       BY INSDRD.
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       IC1,IC2-COLUMNS TO SHIFT. IF THE COLUMNS ARE INVALID, RETURN
C              WITH NO ACTION
C       NC-THE NUMBER COLUMN POSITIONS TO SHIFT.
C       IDIR-DIRECTION
C            1-FORWARD(AWAY FORM ONE)
C            2-BACKWARD(TOWARD ONE)
C--INPUT/OUTPUT PARAMETERS:
C       IDBO(4)-DRAA OF THE DIRECTORY. IT MUST BE CURRENT
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE SHRTND(NDB,NELCR,IDBAA,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO SHORTEN A DL OR DR
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKIN ,2-REFERENCE)
C       NELCR-THE LAST DESIRE  ELEMENT OF THE DL OR DR.
C             ALL ELEMENTS AFTER NELCR WILL BE DELETED(IRREVOCABLY).
C             IF NELCR .E.0, THE ENTIRE DL OR DR CHAIN WILL BE
C             DELETED. IF NELCR IS BEYOND THE END OF THE DL OR DR,
C             RETURN WITH NO ACTION. NELCR HAS DIFFERENT MEANINGS
C             FOR EACH TYPE OF DL.
C               1. A 1-DIMENSIONAL DL --NELCR IS AN ELEMENT NUMBER.
C               2. A COLUMN ORIENTED DL OR
C                    A DIRECTORY  --NELCR IS THE LAST DESIRED COLUMN
C                                   TO RETAIN.
C               3. A ROW ORIENTED DL --NELCR IS THE LAST ROW TO
C                                     RETAIN.
```

```
C--INPUT/OUTPUT PARAMETERS:
C      IOBAA(2)-A DBAA FOR THE DL OR DR. ON RETURN IT WILL BE THE
C               CURRENT DBAA OF THE RECORD CONTAINING NELCR.
C
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           1,2,4,8,OTHERS
C--ALTERNATE RETURNS:
C      1-IERR.NE.0

-----------------------------------------------------------------------

      SUBROUTINE SRDSD(NDB,ITYP,IOFF,MATCH,NOTI,LOCN)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          SRDSD SEARCHES THE DATA STORE FOR AN ENTRY
C          EQUAL TO MATCH THAT IS IN THE SAME DATA BASE. IT LOOKS FOR
C          IABS(KSTORD(I+IOFF))=MATCH, WHERE I IS
C          THE DBCSA OF THE RECORD
C--INPUT PARAMETERS:
C          NDB-DATA BASE(1-WORKING,2-REFERENCE)
C          ITYP-TYPE(1-NUMERIC,2-CHARACTER)
C          IOFF=OFFSET FROM THE 1ST WORD OF THE
C               CONTROL PART
C          MATCH-ELEMENT TO FIND
C          NOTI-SRDSD WILL IGNORE A RECORD WITH DBCSA=NOTI EVEN
C               IF IT MATCHES.  THIS PERMITS A RECORD TO SEARCH
C               FOR ANOTHER RECORD OF THE SAME CHAIN WITHOUT FINDING
C               ITSELF.
C--OUTPUT PARAMETERS:
C          LOCN-THE DBCSA OF THE RECORD WHOSE IOFF-TH
C               WORD=MATCH.
C               ZERO IF NONE.

-----------------------------------------------------------------------

      SUBROUTINE SUP6D(IONOFF)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      SUP6D WILL SUPPRESS DBCS ERROR 6.  SOME INTERNAL DBCS PROGRAMS
C      DELIBERATELY READ PAST THE END OF A DL OR DR. THESE OCCURRANCES
C      ARE NOT REPORTED AS ERRORS BY USING SUP6D TO ELIMINATE THE
C      ERROR MESSAGE.
C
C--INPUT PARAMETERS:
```

```
C       IONOFF-SUPPRESSION SWITCH
C             1-TURN ERROR & REPORTING ON
C             2-TURN ERROR & REPORTING OFF

------------------------------------------------------------------------

      SUBROUTINE SWAPD(ITYP,ICWRK,IDTS,IDWS)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          SWAPD SWAPS A RECORD FROM THE TS TO THE WS. IT DOES
C          NOT EXAMINE THE WS TO UPDATE SAFMND OR MNDSFD.
C          BOTH RECORDS REMAIN LINKED PROPERLY IN THE DATA STORE.
C--INPUT PARAMETERS:
C          ITYP-TYPE(1-NUMERIC,2-NUMERIC)
C          ICWRK(2)-DBCSA'S OF ONE RECORD OF SPACE IN THE IS.
C                   (1)-CONTROL PART,(2)-DATA PART
C--INPUT/OUTPUT PARAMETERS:
C          IDTS-(INPUT)-DBCSA OF THE TS RECORD
C               (OUTPUT)-DBCSA OF THE SAME RECORD NOW IN THE WS.
C          IDWS-(INPUT)-DBCSA OF THE WS RECORD
C               (OUTPUT)-DBCSA OF THE SAME RECORD IN THE TS

------------------------------------------------------------------------

      SUBROUTINE ULKDSD(IDBCS,LSTR)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C          ULKDSD WILL UNLINK A RECORD FROM THE DATA STORE
C          AND, IF NECESSARY, LINK THE
C          FREED UP SPACE TO THE AVAILABLE SPACE IN THE WS.
C          THE RECORD NUMBER IS SET TO ZERO ALSO.  ULKDSD
C          ASSUMES ANY INFORMATION IN THE RECORD IS NOT
C          USEFUL, SINCE IT MAY NOT BE ACCESSIBLE AFTER
C          RETURN.
C--INPUT PARAMETERS:
C          IDBCS-DBCSA OF THE RECORD TO BE UNLINKED.
C                IF IDBCS.LE.0, RETURN WITH NO ACTION
C          LSTR-STORAGE POSITION(IE LSTR=LSTORD(IDBCS))
C               IF LSTR .LE. 0, ULKDSD WILL CALL LSTORD.

------------------------------------------------------------------------

      SUBROUTINE UPDATD(NDB,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
```

```
C       TO UPDATE THE DBF ON DISK SO IT IS THE SAME AS IN-CORE.
C       IF THE DB IS NOT OPEN OR IS READ-ONLY, RETURN WITH NO
C       ACTION.
C--INPUT PARAMETERS:
C       NDB-DATA BASE
C            0-BOTH
C            1-WORKING
C            2-REFERENCE
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C             0-NO ERROR
C            .NE.0-I/O ERRORS ON DB
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE WCD(NDB,ICP,CDP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO WRITE A CHARACTER RECORD TO THE DBF
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       ICP(NWCPD) THE CONTROL PART OF THE RECORD
C       CDP*NCDPD-THE DATA PART OF THE RECORD(CHARACTER)
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C             0-NO ERROR
C             1,2,3004,4004,4005
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C

------------------------------------------------------------------------

      SUBROUTINE WD(NDB,IDBAR,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C           WD WRITES A RECORD TO THE DB
C--INPUT PARAMETERS:
C           NDB-DB(1-WORKING,2-REF)
C           IDBAR(2)-DBCSA'S OF CONTROL AND DATA PARTS,RESPECTIVELY
```

```
C--OUTPUT PARAMETERS:
C          IERR-ERROR CODE
C               0-NO ERROR
C--ALTERNATE RETURNS:
C          1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE WID(NDB,ICP,IDP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO WRITE A INTEGER RECORD TO THE DBF
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      ICP(NWCPD) THE CONTROL PART OF THE RECORD
C      IDP(NWDPD)-THE DATA PART OF THE RECORD
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           1,2,3004,4004,4005
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C

------------------------------------------------------------------------

      SUBROUTINE WRD(NDB,ICP,RDP,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C      TO WRITE A REAL RECORD TO THE DBF
C
C--INPUT PARAMETERS:
C      NDB-DATA BASE(1-WORKING,2-REFERENCE)
C      ICP(NWCPD) THE CONTROL PART OF THE RECORD
C      RDP(NWDPD)-THE DATA PART OF THE RECORD
C--OUTPUT PARAMETERS:
C      IERR-ERROR CODE
C           0-NO ERROR
C           1,2,3004,4004,4005
C--ALTERNATE RETURNS:
C      1-IERR.NE.0
C

------------------------------------------------------------------------
```

```
      SUBROUTINE W1D(NDB,IERR,IOERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       TO WRITE THE FIRST RECORD TO A DBF
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERR
C            1,2,3004
C       IOERR-SYSTEM I/O ERROR CODE
C             0-NO ERROR
C             .NE.0-SYSTEM ASSIGNED ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
C
```

```
      SUBROUTINE W2RWD(IRW,IFMT,IFILE,LDBF,IERR,*)
C--MODULE: MOPADS/DBCS
C--REFERENCE: MOPADS VOLUME 5.13
C--PURPOSE:
C       W2RWD WILL :
C          1. READ SEQUENTIAL INTERNAL/EXTERNAL FORMATTED DBF'S TO
C             AN EMPTY DIRECT ACCESS DBF.
C          2. WRITE A DIRECT ACCESS DBF TO A SEQUENTIAL INTERNAL
C             OR EXTERNAL FORMAT FILE.
C       NO REFERENCE DATA BASE MAY BE OPEN DURING THIS OPERATION.
C--INPUT PARAMETERS:
C       IRW-READ/WRITE FLAG
C           1-READ SEQUENTIAL FILE AND WRITE TO A DIRECT ACCESS DBF
C           2-READ DIRECT ACCESS DBF AND WRITE TO A SEQUENTIAL FILE
C       IFMT-SEQUENTIAL FILE FORMAT
C            1-SEQUENTIAL INTERNAL (UNFORMATTED) FILE
C            2-SEQUENTIAL EXTERNAL (FORMATTED) FILE
C       IFILE-LOGICAL UNIT NUMBER OF THE SEQUENTIAL FILE. THE DBCS DOES N
C             OPEN, CLOSE, OR REWIND THIS FILE. IF IRW=2, AN ENDFILE WILL
C             BE EXECUTED ON THIS FILE AFTER THE DB IS WRITTEN TO IT.
C       LDBF-(CHARACTER) THE NAME OF THE DIRECT ACCESS DB FILE. W2RWD WIL
C             OPEN THIS FILE AND CLOSE IT WHEN DONE. LDBF SHOULD NOT BE
C             CONCURRENTLY OPEN AS THE WORKING OR REFERENCE DB.
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR.  POSITION OF FILE IFILE IS UNPREDICTABLE.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

VI. USER INSTRUCTIONS

1-0 ACCESS ADDRESSES.

The primary means of accessing information with the DBCS is through access addresses. There are two types: Data Base Access Addresses (DBAA's) for data lists and Directory Access Addresses (DRAA's) for directories. DBAA's consist of two words, and DRAA's consist of four words. See Table VI-1. The user's application program need never be concerned about the contents of the DBAA's and DRAA's. It must simply provide storage to hold the addressing information.

The normal accessing method is as follows: The application program requests the DBCS to locate a data list or directory (the ways this can be done are explained below). The DBCS locates it and returns its DBAA (or DRAA) in an array the user has provided. What the DBCS has done is bring at least one record of the DL (or DR) into core and store its DBA (or its negative if it is a character record) in word one of the DBAA (DRAA). Word two contains its current DBCSA. For directories, words 3 and 4 are also set to zero (this essentially sets the current directory positions to the beginning).

Now, whenever the application program desires to access that DL (or DR), it passes the DBAA (DRAA) to the DBCS which is able to quickly locate the information in core or to determine that additional records must be read from disk. As discussed previously, all record management is performed internally by the DBCS. The DBCS also updates elements of the DBAA's (and DRAA's) as needed.

The DBAA and DRAA are the primary means by which the DBCS avoids frequent DB searches for information. Conventional hierarchical searches for a DL or DR can be performed once (these are slow), and then the DBAA or DRAA can be used on all subsequent accesses of that data (these are fast). The user program must "remember" DBAA's and/or DRAA's for each DL or DR which is frequently accessed in order to take advantage of this efficient addressing.

2-0 USING THE DBCS.

2-1. Initializing the DBCS.

Subroutine INITD must be called prior to calling any other DBCS programs. Note that BLOCK DATA program BLOCKD must be loaded with the DBCS also.

Table VI-1. Contents of Data Base and Directory Access Addresses (DBAA and DRAA).

Word	DBAA	DRAA
1	The DBA of the last record of the DL accessed. Word 1 will be negative for character DL's.	Same as for DBAA but DR's.
2	DBCSA of the record specified in Word 1.	Same as for DBAA.
3	Not applicable.	Current directory position for directories of the DR.
4	Not applicable.	Current directory position for data lists of the DR.

2-2. Opening/Closing DB's.

Use Subroutine OPNDBD to open DB files. MAXR is a parameter needed on some computer systems which require that the maximum size of the file be specified at creation time.

Use Subroutine CLSDBD to close a DB. If ISTAT is specified as 2, the DBF will be deleted(i.e., lost forever) after the file is closed.

DBF's may be opened and/or closed as often as needed during a DBCS job.

2-3. Creating Directories.

Two subprogram calls are required to create a directory. The first is a call to Subroutine CRDRD. CRDRD creates an empty directory that is not owned by another directory. It also returns the DRAA of the new directory. Subroutine INSDRD is used to "insert" the new directory into another directory. In other words, after calling INSDRD, the new directory is owned by another directory.

INSDRD requires knowledge of the DRAA of the owning directory. This is no problem except for the master directory. When inserting directories (or data lists) in the master directory, it is necessary to know the master directory's DRAA. This case is the only instance in which an application program must be aware of internal DBCS variables. The array ISYDBD (see Section VIII) always contains the DRAA of the master directory. The user application program must have COMMON/DBCS3D/ to gain access to this array. The recommended procedure is to minimize the number of application program subprograms which must access this COMMON area.

2-4. Creating Data Lists.

Creating data lists is similar to creating directories. Two subprogram calls are required. The first is to Subroutine CRDLD to create the DL, and the second is to Subroutine INSDRD to insert the new DL in its proper place in the owning direcotry. Creating DL's to be owned by the master directory require access to the array ISYDBD as explained in Section 2-3 above.

The data lists created by this procedure are empty. In other words, they contain no elements. Populating a data list is discussed next.

2-5. Storing and Retrieving Information From a Data List.

There are 18 subprograms for storing and retrieving elements of data lists which will be described shortly. When an application program nstructs that data be stored in an element that is beyond the current length of the data list, the DL is extended and the data is stored. All intervening elements are also created and are assigned values of zero for numeric DL's or blank for character DL's. For example, if a user assigns a value of 15 to element 12 of a new DL (which currently has no elements), then element 12 will equal 15 and elements 1 through 11 will exist and equal zero.

If a user program requests the value of an element that is beyond the current length of a DL, an error results (see DBCS error 6 in Section VII).

The subprograms for storing and retrieving DL elements are shown below.

P{C,I,R}D	-	These subroutines put information into DL's of type character (C), integer (I), and real (R).
G{C,I,R}D	-	These subroutines retrieve (get) information from DL's analogous to PCD, PID, and PRD above.
PROW{C,I,R}D	-	These subroutines put information into a row of a two-dimensional DL. See the note below.
GROW{C,I,R}D	-	These subroutines retrieve information from a row of a two-dimensional DL. See the note below.
PCOL{C,I,R}D	-	These subroutines put information into a column of a two-dimensional DL. See the note below.
GCOL{C,I,R}D	-	These subroutines retrieve information from a row of a two-dimensional DL. See the note below.

NOTE

Recall that character DL's can be one-dimensional only. Therefore, Subroutines PROWCD, GROWCD, PCOLCD, and GCOLCD are non-functional. They are included only to allow for future modifications to the DBCS.

With each of these subprograms, the user program passes in an array to hold the values of the DL. If the operation is to retrieve data, the DBCS copies the specified values to this array. If the operation is to store data to the DL, the DBCS copies the new values from the user array into the DL. This effectively overwrites the previous values of the elements that were stored in the DL.

When storing and retrieving data from two-dimensional DL's, the user need not be concerned as to whether the DL is row or column oriented. The DBCS accounts for this internally. It is, however, more efficient to access row oriented DL's by rows and column oriented DL's by columns. In other words, it is more efficient to use PROW-D and GROW-D with a row oriented DL. This implies that during design of the data base, DL's that will be accessed most frequently by rows, for example, should be row oriented.

2-6. Assigning and Accessing Labels of Data Lists and Data List Elements.

To give a data list a label, use Subroutine PLBDLD. PLBDLD will overwrite a previous label. If no label has been assigned to a DL, it has a default label of "(UNLABLED)".

Subroutine PLBELD is used to assign labels to elements of DL's. In this case an array of labels is passed to PLBELD for assignment to specified elements. Every element of a DL may be assigned a label, even every element of a two-dimensional DL. DL elements which have no assigned labels are given default labels of "(UNLABELED)". Recall that all DL and DL element labels have 25 or less characters.

Subroutine GLBDLD will return the label of a DL given its DBAA. Similarly, Subroutine GLBELD will return labels of specified elements of a DL given the DL's DBAA. If the user program requests labels beyond the end of the LDL (in other words, the LDL is shorter than its parent DL so some DL elements have no assigned labels), GLBELD will return labels of "(UNLABELED)" for the missing elements, and it will also generate an error number 6 (see Section VII).

2-7. Assigning and Accessing Directory Labels.

Directory labels are normally assigned when the DR is created. A directory may be renamed by using Subroutine RNDRD. A DR's label can be accessed with Subroutine GLBDRD. Both RNDRD and GLBDRD require the DR's DRAA.

2-8. Searching a Directory.

It is sometimes necessary to search a directory for a particular DL or DR that is owned by the directory. There are several facilities in the LBCS for doing this.

2.8.a. Subroutine FDBID will search a directory for an owned DL or DR with a specified ID. FDBID returns the DBAA or DRAA of the found DL or DR and sets the current position of the owing DR to the found entry.

2.8.b. Subroutine CDLLBD will search a directory for an owned DL that has a specified label. GDLLBD returns the DL's DBAA and sets the owning DR's current DL position to the found data list.

2.8.c. Subroutine NXD provides a flexible search capability to locate owned DL's and DR's. The important feature of NXD is that the search criteria are specified by the user. NXD allows searches of the following type:

- Find the next DL in the forward direction (away from the start of the DR) with
  - the first ID element, ID(1), equal to 4, and
  - the third ID element, ID(3), greater than 6, and
  - the fifth ID element, ID(5), not equal to 1.

NXD, in the case above, would begin at the current DL position of the DR and search forward for a DL whose ID satisfies the specified condition. Similar searches can be performed for owned Directories.

NXD returns the found entity's ID, and DBAA or DRAA; it also sets the owning DR's current DL or DR position to the found entity.

2.8.d. Subroutine POSDRD can be used with NXD. POSDRD sets the current DL or DR position of a directory to its beginning or end.

2-9. Locating a Directory or Data List.

2.9.a. Subroutine GDRLBD will return the DRAA of a directory that has a specified label. It will search the entire data base. If directory labels are not unique, GDRLBD will return the DRAA of the first directory whose label matches the specified label.

2.9.b. Subroutine FIDLD will return the label and ID of a DL or DR that has a specified DRAA.

2-10. Clearing and Deleting Directories.

2.10.a. Subroutine CLRDLD will delete all data lists that are owned by a specified directory. The DL's are irrevocably lost.

2.10.b. Subroutine CLRDRD will empty a directory of all of its contents. All owned DL's are deleted and all owned directories with all of their owned entities, etc., are deleted. In other words, CLRDRD deletes every thing which descends from a specified directory.

2.10.c. Subroutine DELDRD will delete a directory. It removes the ID of the DR from its owning directory and deletes the directory itself. A directory must be empty before it can be deleted, so the user program must explicitly delete all owned DL's and DR's or use Subroutine CLRDRD to clear the directory prior to calling DELDRD.

2-11. Shortening and Deleting Data Lists.

2.11.a. Subroutine SHRTND will shorten a data list. A parameter, NELCR, is passed to SHRTND which specifies the last element to retain. For one dimensional DL's, the NELCR will be the last element of the data list. For column oriented DL's, NELCR will be the last column; all subsequent columns will be deleted. Similarly, for row oriented DL's, NELCR is the last row that is retained. If NELCR = Ø for any case, the DL is emptied, but it is not deleted.

2.11.b. Subroutine DELDLD deletes data lists. In addition to deleting all elements of the DL, it also removes its entry from its owning directory. No call to SHRTND is needed prior to calling DELDLD.

2-12. Finding the Owner Directory and Determining If a Data Base is Open.

2.12.a. Subroutine PLINKD will return the DRAA of the owner of a specified directory or data list. The programs described in Section 2-8 can be used to find entities owned by a directory and PLINKD can be used to find the owner of a DL or DR. Thus, it is possible to search up and down the data base links for information.

2.12.b. Subroutine ASKD will return an indication of whether a particular DB (working or reference) is open.

2-13. Finding the Current Position of a Directory.

Subroutine CURD will return the initial DBA and ID of the current DL or DR position of a directory. If the initial DBA of the current position is stored in the first word of an array of

length two (four for directories) and the rest of the array is zero filled, then the array may be used as a DBAA (DRAA) for the entity at the current position.

2-14. Printing the Contents of a Data List or Directory.

It is possible to to get the contents of a directory or data list printed on the DBCS output file. This file is formatted with standard carriage control characters so it can be listed on a line printer.

Subroutine PRTDLD will print all or part of a data list on the output file. Use Subroutine PRTDRD to print the contents of a directory. Both of these subprograms allow the user programs to specify a message to be printed with the DB information.

2-15. Set and Retrieve DBCS Options.

Subroutine DBOPTD allows the user program to set (and retrieve the settings) of 17 options for the DBCS. Many of the options are self explanatory from the comments in Section V for Subroutine DBOPTD. Some options require explanations, however; these options are discussed here.

The DBCS has a trace option (DBOPTD options 4 and 5) which will print a history of entrances and exits to each DBCS subprogram. In other words, a message is printed each time a DBCS subprogram is entered. This option is off if the trace code is one and on if the code is two. The DBCS trace is useful only for debugging the DBCS code and has little utility in debugging application programs. It also will produce a great deal of output. Its use by application programs is not recommended, since its purpose is to aid in systems development.

The DBCS has four levels of error processing which are described in Section VII. They are coded as follows:

1 - information
2 - warning
3 - severe
4 - fatal

The error print flag option (DBOPTD option 6) is one of these numbers. Standard DBCS error messages are printed on the DBCS output file when errors of severity greater than or equal to the error print flag occur. The DBCS default for the error print flag is one. This is also the default for MOPADS.

The DB protection mode options for DBOPTD options 7 and 8 are given in Table II-6.

The SAF smoothing constant (default = 0.5) is specified with DBOPTD option 9. The lower and upper bound on the SAF update periods are specified with DBOPTD options 10 and 11. The DBCS defaults are 75 and 125, respectively. The MOPADS settings for these options are 150 and 250. The DBCS samples uniformly between these values to determine the next update point for SAF's. The default values selected here are somewhat arbitrary and may be changed as experience is gained with the DBCS.

The effect of these three parameters is as follows:

a. The values specified are used for SAF updates until they are again changed or until another working DB is opened.

b. When a DB is closed (either working or reference), the current values of SAF and the min and max update period are written to the DBF.

c. When an old DBF is opened as the working DB, the values of SAF and the min and max update period are read from the DBF. They become the current values and are used in accordance with (a) and (b) above.

With option 12 and 13, the user can specify the number of print columns and the number of lines per page for the DBCS output file and the trace file. The printing device must have at least 80 columns.

Option 14 allows the user to specify the number of words per record for the DBF. The DBCS will function with any record size greater than 16 words. This parameter should be set once for the computer installation to optimize disk access speed. Most computer systems achieve maximum disk access efficiency if the record size is an even multiple of the disk sector size.

The number of words per record is a characteristic of the DBF not the DBCS. When a new DB is opened, the DBCS will read the record size from the DBF and adjust accordingly. The exception to this is that both the working and reference DB's must have the same record size.

Options 15, 16, and 17 allow the user to adjust the sizes of the internal and temporary stores. The internal stores must have at least three records and there is no benefit to increasing their sizes.

As discussed in Section II, 3-0, the DBCS partitions its main storage arrays to provide storage for the TS and WS. The number of records specified in options 16 and 17 are allocated to the TS.

All remaining available space is allocated to the WS. This permits the user to change the proportion of space allocated to the TS and WS storage areas.

The storage allocation is a characteristic of the DBCS, not the DBF's. In other words, the DBF is not structurally affected by these parameters. Options 15, 16, and 17 may not be changed while any data base is open, however. The DBCS will raise an error if either a working or a reference DB is open when a request to change one of these options is made.

2-16. Internal and External Sequential Data Bases.

Subroutine W2RWD will create or write sequential internal or external formatted representations of data base files. These files have the following characteristics.

Internal Format	-	sequential, unformatted, variable length records. No record exceeds the length of the data part, however.
External Format	-	sequential, formatted, variable length records. No record exceeds 80 characters, however.

The internal format can be used to archive DBF's on tape or to transfer them among similar computer systems. The external format can be used to archive DBF's on tape and to transfer them among dis-similar computer system. The direct access data base file to be written or read must not be opened as the working or reference DB when W2RWD is called. Furthermore, if a sequential access file is being read to create a new direct access DBF, the DBF file will be opened with STATUS = 'NEW' and will have the record size specified on the sequential access file.

2-17. Printing the Data Store.

Subroutine PSTORD prints the contents of the DBCS data store (IS, TS, WS) to the DBCS output files. This is a debugging tool only, and should normally never be required.

VII. ERROR PROCESSING

1-0 SUBROUTINE ERRORD.

The single error processing subprogram in the DBCS is Subroutine ERRORD. When an error is detected, ERRORD is called with the error code, an text message, the name of the calling program and a list of integer and real parameters. If the error print flag permits (see 2-0 below), a message is printed on the DBCS output file. In addition, actions are performed which are conditional upon the severity of the error. This severity is determined by the value of the error code as in Table VII-1.

Table VII-1. Error Severity Codes.

Error Code	Severity Code	Severity
± 1-1999	1	Information
± 2000-2999	2	Warning
± 3000-3999	3	Severe
± 4000-	4	Fatal

Information errors indicate requested actions that cannot be performed. Warning errors are similar to Information errors but are judged to be more severe in that they reflect a serious mis-judgement or mistake by the user. The difference between these two is subjective since ERRORD treats them essentially the same. After the error message is printed (conditioned on the error print flag) a RETURN is executed from ERRORD. The calling DBCS program will normally halt its activity and return (with the error code) to the calling application program.

Severe errors imply that the user program has attempted to destroy the integrity of the DB, has violated DBCS conventions, or a computer system error has been detected. In this case, ERRORD will update the DB (i.e., make the disk version identical to the in-core version), set the DB to read-only, and return. If subsequent errors occur during these activities, execution will terminate immediately.

Fatal errors occur when indicators of faulty data bases are detected or certain capacity limitations are exceeded. ERRORD performs the same actions as for Severe error except that it terminates execution instead of RETURNing.

2-0 THE ERROR PRINT FLAG.

As discussed in Section VI, 2-15, the DBCS error print flag can be used to suppress printing from Subroutine ERRORD. The error print flag is set to one of the error severity codes in Table VII-1. Printing is then suppressed for any error with a severity code less than the error print flag. The default error print flag is one.

3-0 DBCS ERROR CODES.

3-1. Information Errors.

Error Code	Description
1	Working data base not open.
2	Reference data base not open.
3	No output file available.
4	Invalid DBA for specified purpose.
5	Attempt to access DL or DR with invalid elements.
6	Attempt to read beyond the end of a DL (see Section 4-0 below).
7	Force read flag overwirtes in-core data that is not on disk.
8	Attempt to write to read-only DB.
9	Attempt to access two dimensional DL with incorrect dimensions.
10	Attempt to access a DL with wrong data type (real, integer, etc.).
11	DBAA not current when it is required to be.
12	DBAA not a DL when it should be.
13	Not used.
14	Attempt to create a DR with incorrect or incompatible ranking codes.
15	DRAA not a DR when it should be.
16	Incorrect conditions specified for search of a directory.
17	Invalid option to DBOPTD. Examine integer parameter 1.

18	Invalid unit number. See integer parameter 1.
19	Insufficient row/column specification for DBCS output or trace file. See integer parameter 1.
20	Unit number invalid or already used. See integer parameter 1.
21	No reference DB may be open during this operation.
22	Direct access DBF not empty.

3-2. Warning Errors.

Error Code	Description
2000	No more records available.
2001	Attempt to delete a directory that is not empty.
2002	Attempt to insert incorrect type (DL or DR) into a directory.
2003	Attempt to open a DB without prior call to INITD.
2004	Attempt to change the unit number of an open DB.
2005	Both DB's must be closed in order to reconfigure the data store.
2006	Record length too short. See integer parameter 1.

3-3. Severe Errors.

Error Code	Description
3000	Attempt to open a DB with record size different from the current size.
3001	Attempt to open a DB with control part size different from the current size.
3002	Attempt to open a DB with data part size different from the current size.
3003	Attempt to open a DB with variable NWADD different from current size.
3004	System I/O error. Integer parameter 1 is the system error code.
3005	Attempt to create character DL with too many characters/elements. Maximum allowed is integer parameter 1.
3006	Internal DBCS system error. Attempt to return record number zero to the ARL.

3-4. Fatal Errors

Error Code	Description
4000	Invalid number of words/record.
4001	Insufficient storage in the data store.
4002	A record has been read which indicates a faulty DBF update (negative record number).
4003	Internal store exhausted. (See DBCFTD option 15).
4004	Mismatch between actual record type and the DBAA type. Integer parameter 1 is record number.
4005	Attempt to access invalid record (record number is .LE.0 or .GT.MXRD(2,-).
4006	Incorrect index of KNPD array sent to KPNPD.
4007	Reference DB record found in working store or attempt to swap reference DB record into WS.
4008	A DL is not in the DR specified in its control part.
4009	Insufficient internal work space. Increase size of KWORKD. Minimum needed is integer parameter 1.
4010	A DR is not in the directory specified in its control part.
4011	Duplicate records found in the data store.

4-0 SPECIAL PROCESSING OF ERROR CODE SIX.

Error code six flags an attempt to read past the end of a DL. This action is frequently done on purpose to find the end of a DL or DR. Therefore, printing of the error message is usually suppressed (regardless of the value of the error-print flag). The DBCS determines internally when to suppress printing. In general, any call from an application program that reads past the end of a DL will have the error printing suppressed (the error code value of six is still returned, however). Errors internal to the DBCS that result in such an error will not be suppressed, however.

VIII. COMMON VARIABLE DEFINITIONS

1-0 PARAMETERS

Parameter Names	Value	Definition
MXWRKD	100	length of the KWORKD array
NCPWD	4	number of characters/word for the computer
NCSTRD	6400	number of characters in the character data store (CSTORD)
NSTORD	6000	number of words in the numeric data store (KSTORD/STORD)

2-0 COMMON/DBCS1D/

Variable Name	Default Value	Definition
KCPD(3)	-	defines the partition of the character data store KCPD(1) - 1st character of the IS (2) - 1st character of the TS (3) - 1st character of the WS
KNPD(9)	-	defines the partition of the numeric data store. See Table II-4.
KSTORD(NSTORD)/ STORD(NSTORD)	-	EQUIVALENCEd arrays that hold the numeric data store

3-0 COMMON/DBCS2D/

Variable Name	Default Value	Definition
CSTORD*(NCSTRD)	-	character variable to hold the character data store

4-0 COMMON/DBCS3D/

Variable Name	Default Value	Definition
ALPHD	0.5	SAF smoothing constant
ITRCD	1	trace code
ISYDBD(4,2,2)	-	DBAA's for the master and system directories ISYDBD(IEL,IMS,NDB): IEL = elements 1 to 4 of the DBAA IMS = 1-master directory 2-system directory NDB = 1-working DB 2-reference DB
ISYSD(2,2,2)	-	DBAA's for the directory label and directory address DL's in the system directory. ISYSD(IEL,IMS,NDB) IEL = elements 1-2 of the DBAA IMS = 1-DR label data list 2-DR address data list NDB = 1-working DB 2-reference DB
JOUTD	10	DBCS output file unit number
JTRCD	10	trace output file number
KARLD(2,2)	-	DBAA's of the ARL's KARLD(IEL,NDB): IEL = elements 1 and 2 of the DBAA NDB = 1-working DB 2-reference DB
KEPFD	1	error print flag
KPRTD(2,2)	$\begin{pmatrix}132,132\\64,\ 64\end{pmatrix}$	output characteristics for the output (column 1) and the trace (column 2) files. Row 1 - columns/page Row 2 - lines/page

KSUP6D	1	error code 6 suppression flag 1 - print error 6 2 - suppress printing of error 6
KWORKD(MXWRKD)	-	internal DBCS working storage
LUD(2)	(-1,-2)	DB unit numbers ((1)-working, (2)-reference) Negative if DB is not open
MDLAD	-	number of DB accesses required till the next SAF update
MFAD(2)	-	DBCSA in working and temporary store of the first unused record slot. (1)-numeric, (2)-character
MHDLAD MLDLAD	125 75	MDLAD is uniformly distributed between MLDLAD and MHDLAD-MOPADS defaults for these variables are 150 and 250
MMSFD(2)	-	DBCSA of the WS record with the minimum SAF. (1)-numeric, (2)-character
MODED(2)	-	protection modes ((1)-working DB, (2)-reference DB)
MXRD(2,2)	-	MXRD(1,i) - maximum record number allowed (0 for no limit). MXRD(2,i) - largest record number used so far for DB i. i = (1)-working, (2)-reference
NCDPD	-	number of characters in the data part of each record
NCTSD(2)	(0,3)	number of records of storage available in the character TS. (1)-current, (2)-default value
NCWSD	-	number of records of storage for the character WS
NDLAD	-	number of DB accesses since last SAF update

NISD	4	number of records of storage in the IS (both numeric and character
NTSD(2)	(0,5)	number of records of storage available for the numeric TS (1)-current value,(2)-default
NWADD	2	number of words of addressing information in addition to the ID in the data part of a directory. In other words, the number of rows in a directory is the length of the ID's plus NWADD. See Section II, 1-2.
NWCPD	10	the number of words in the control part of each record
NWDPD	-	number of words in the data part of each record
NWPECD(2)	(0,64)	number of words per record. (1)-current, (2)-default
NWSD	-	number of records of storage available in the numeric WS.
OALPHD	-	1.-ALPHD
SAFMND(2)	-	value of the minimum SAF of any record in the WS. (1)-numeric, (2)-character
TDLAD	-	cumulative total of DB accesses

5-0 COMMON/DBCS4D/

Variable Name	Default Value	Definition
CDBFD(2)*35	blank	file names of the working and reference DB's

IX. REFERENCES

Polito, J. The MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a).

Polito, J. The MOPADS data base application programs (MOPADS/DBAP) (MOPADS Vol. 5.18). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b).

Polito, J. & Goodin II, J. R. MOPADS utility programs (MOPADS/UI) (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

X. DISTRIBUTION LIST

PERI-IB (5)
U. S. Army Research Institute Field Unit
P. O. Box 6057
Fort Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg.#1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service - Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St. Room 108
Boulder, Colorado 80301

XI. CHANGE NOTICES

APPENDIX V

MOPADS FINAL REPORT:

MOPADS FREE-FORMAT INPUT PROGRAM (MOPADS/FFIN2)

Revised 5/14/82

PREFACE

This report contains changes dated 14 May 1982 which are not upward compatible with the 29 July 1981 revisions. Users who have applications utilizing this earlier version should retain prior copies of this report for reference. The non-upward compatible features involve the use of the line terminator character and subroutines FFPAC3, FFCPYI and FFCPYO.

TABLE OF CONTENTS

Revised 10/8/82

TABLE OF CONTENTS
(Continued)

LIST OF FIGURES

SECTION I

INTRODUCTION

A set of FORTRAN 77 subprograms is presented here that permit input of data without regard to column spacing on the input device. The system maintains an internal buffer array into which input records are read. The buffer can individually address the characters in the record. The buffer is then searched for a field of data that is delineated by user specified separator characters. When a field is found it is interpreted as a datum of the appropriate type. When the user next requests a data item, the buffer is searched from the point where the last field was encountered. This continues until the internal buffer is exhausted. Then a new record is read into the buffer and the process repeats.

Several options and environment parameters can be specified by the user. A brief description of these follow.

Input and Output Files

An input file must be specified with each call to the system for data. No default input file is provided.

An output file must also be specified. The default output file is unit 6. The output file is used for printing error messages. If an invalid output file unit number is specified, a warning message will be printed and the program will abort if any data is subsequently erroneously entered. The user is responsible for providing a file with the appropriate unit number.

Data Echo File

Another file, which may be the same as the output file, may be specified as the echo file. Each record will be written to the echo file as it is read. The default condition is no echoing of data.

Error Processing

The system provides two levels of error processing: interactive and batch. For batch processing, error messages are printed and execution is terminated. With interactive processing, the user is given the opportunity to re-enter erroneous data. The default condition is interactive.

Numeric Field Validity Checking

The option for checking for valid numeric fields can be either "on" or "off." If it is on, each request to interpret a field as a number results in an examination of the field for validity. The default condition is "on" and is recommended for interactive use. The FFIN system is slower than

standard formatted data input, and it is not recommended for reading large, tabular data files. If it is used with input files which the user is confident have valid numeric fields, then this option should be turned "off."

Line Terminator Character

A character may be provided which terminates processing of a record. If this character is encountered in the buffer, the remainder of the buffer is not examined, and a new record will be read when additional fields are needed. Use of the line terminator character will speed processing on short records, and it permits comments to be placed on data records. The default line terminator is the dollar sign ($). The line terminator character is not interpreted when in a quote field (see "The Quote Character").

Field Separators

Fields are separated by separator characters. Up to five separator characters may be specified. Two are provided by default (blank and comma). There is no hierarchy among separators, and multiple, consecutive separators are treated as a single separator. Thus the following records are equivalent.

```
10, 20, 30
10,,,20 ,  ,30
10 20 30 $
```

Any character which is not a separator, a quote (see the next page) or, in some cases, a line terminator, is part of a field. No characters are ignored.

The Quote Character

Sometimes it is desirable to have a separator character or the line terminator character contained in a field. For example, blanks are by default separators; however, a blank may be desired in an alphanumeric or a numeric field. Such fields may be enclosed in quote characters. The default quote character is the double quote mark ("). Any field can be enclosed in quotes. When a quote is the first non-separator character in a field, the system stops searching for separator characters and line terminators and instead searches for a matching quote to end the field. The quote marks are not part of the field. For example, the following are not equivalent fields

```
GO TO
"GO TO"
```

GO TO is two fields; "GO TO" is one. Quote marks are useful mostly for alphanumeric or character fields, but they can be used with any field. In particular,

```
"-.   3"
```

would be interpreted as -0.0003 (BLANK=ZERO) or -.3 (BLANK = NULL).

Hollerith Character Count

FFIN2 supports Hollerith data for compatibility with older systems. The user can specify the number of characters that he wishes to read per word. This option defaults to one character per word.

Input Record Length

The user can specify the number of columns (up to 132) to be read from the input file and the number of columns to be examined for valid data. This permits the data to have identifiers at the end of each record which are not examined by FFIN2. The defaults are 90 columns read and 80 columns examined.

SECTION II

USE OF THE INPUT SUBPROGRAMS

This section describes the subprograms used for reading data with FFIN2. Section III presents several utility subprograms which can be used for changing the default options and parameters. Section IV contains programs for advanced users who need to change the normal sequence of processing the buffer.

Six types of data fields are recognized by the FFIN2 programs: character, logical, double precision, real, integer, and Hollerith. In general, FFIN2 will recognize any field within these six types that FORTRAN will accept. Alphanumeric and logical fields can be of any length so long as they fit within a record. No field may ever be split by the end of a record. All numeric fields are limited to 20 characters. Each call to one of these subprograms returns a single datum.

FFL

FFL will return the value of a logical variable. The calling sequence is:

```
SUBROUTINE FFL (LTF, JRD).
```

LTF must be type logical, and JRD is the input file. An

error is generated if the first non-blank character of the next field is not a "T" or "F."

FFD

FFD will return the value of a double precision variable. The calling sequence is:

SUBROUTINE FFD (DVAR, JRD).

DVAR must be type double precision, and JRD is the input file. The "D" notation may be used with double precision input fields, e.g., 3.4D1. Double precision fields are converted with a D20.0 specification.

FFR

FFR will return the value of a real FORTRAN variable. The calling sequence is:

SUBROUTINE FFR (RVAR, JRD).

RVAR must be type real, and JRD is the input file. Real fields are converted with a G20.0 specification.

FFI

FFI will return the value of an integer FORTRAN variable. The calling sequence is:

SUBROUTINE FFI (IVAR, JRD).

IVAR must be type integer, and JRD is the input file. Integer fields are converted with an I20 specification.

FFC

FFC will return alphanumeric data as character variables. The calling sequence is:

SUBROUTINE FFC (CHAR, NC, LCHAR, JRD).

CHAR is a CHARACTER array of length NC. FFC will return the next field in CHAR with LC = min [LCHAR, LEN(CHAR(1))] characters per "word." If the field has fewer than NC*LC characters, the remainder of CHAR is blank filled. If the field has more than NC*LC characters, the right most excess characters are ignored.

FFA

Subroutine FFA will return an alphanumeric field as Hollerith data. The calling sequence is:

SUBROUTINE FFA (LALPH, NALPH, JRD).

LALPH is an integer array of length NALPH. JRD is the input file. Say the current alphanumeric character count is NOWCHR. The first (from the left) NALPH * NOWCHR characters of the next field will be read into LALPH with NOWCHR characters per word. Any excess is ignored. If the field has fewer characters than NALPH * NOWCHR, the remaining words of LALPH will be blank filled. FFA is the only non-ANSI FORTRAN 77 subprogram in FFIN2.

FFCLR

FFCLR clears the internal buffer. The calling sequence is:

```
SUBROUTINE FFCLR.
```

Any unused fields in a record will be ignored after a call to FFCLR. The next call to one of the above FFIN2 programs will result in a new record being read.

Example

Suppose file 12 has data on children in a school. Each record contains the following information:

Last name,first name,M or F, age, height, weight.

The first three fields are alphanumeric. M or F specifies male or female. Age is an integer number and height and weight are decimal numbers.

If we want to collect statistics on the height and weight of all boys between the ages of 9 and eleven we could use the following code.

```
      CHARACTER*8 FIRST, LAST
      REWIND 12
100   CALL FFC(LAST, 1, 8, 12)
      CALL FFC(FIRST, 1, 8, 12)
```

```
      CALL FFA(MF, 1, 12)
      IF(MF.EQ.1HM) GO TO  150
120   CALL FFCLR
      GO TO 100
150   CALL FFI(IAGE,12)
      IF(IAGE.LT.9.OR.IAGE.GT.11) GO TO 120
      CALL FFR(HEIGHT, 12)
      CALL FFR(WEIGHT, 12)
        .
        .
        .
      collect statistics
        .
        .
        .
      GO TO 100
```

If a record is not for a boy, the remainder of the information is discarded at statement 120, and a new record is read. The age criteria are checked after statement 150.

SECTION III

USE OF FFIN2 UTILITY SUBPROGRAMS

Subprograms are provided for use in changing the default options and for use in identifying key words for syntax processing.

FFSET

FFSET is used to change options in the FFIN2 system. The calling sequence is:

SUBROUTINE FFSET(IOPT, LOPT, NOPT. COPT, NCR).

IOPT is the option code and has one of the following values.

1 - set output file number

2 - set echo file number

3 - set error processing mode

4 - set numeric field validity option "on" or "off"

5 - set new quote character

6 - set new line terminator character

7 - set new separators

8 - not used

9 - set the Hollerith character count

10 - set input record length parameters

LOPT is a one dimensional, integer array of length NOPT. LOPT contains the new values to be set for the options that require

numbers. COPT is a CHARACTER *1 array of length NCR and contains alphanumeric option data.

1. Output File Number. To set the output file number to N, use:

   CALL FFSET (1, N, 1, ' ', 1).

   A warning will be printed if N $\leq$ 0.

2. Echo File Number. To set the echo file to N, use:

   CALL FFSET (2, N, 1, ' ', 1).

   If N $\leq$ 0, no echo of input records will take place.

3. Error Processing Option. To set the error processing mode, use:

   CALL FFSET (3, N, 1, ' ', 1).

   If N = 0, then interactive error processing is selected. If N $\neq$ 0, batch operation is assumed, and execution will terminate if a bad field is encountered.

4. Numeric Field Checking Option. To specify the numeric field checking option use:

   CALL FFSET (4, N, 1, ' ', 1).

   If N = 0, fields will be checked. If N $\neq$ 0, no checking is done of numeric fields.

5. Quote Character. To set the quote character to, say a colon (:) use:

```
CALL FFSET (5, 0, 1, ':', 1).
```

6. Line Terminator Character. To set the line terminator character to, say a semicolon(;) use:

```
CALL FFSET (6, 0, 1, ';', 1).
```

7. New Separators. To specify 4 separators which will be, say, comma (,), slash(/), blank ( ), and period (.) use:

```
CHARACTER*1 IS(4)
DATA IS(',','/', ' ', '.')
  .
  .
  .
CALL FFSET (7, 0, 1, IS, 4).
```

The new set of separators completely replaces the old set, so the IS array above must contain all of the separators that the user wants in effect after the subroutine call. No more than five separators are permitted.

8. Not Used.

9. Hollerith Character Count. To set the alphanumeric character count to N, use:

```
CALL FFSET (9, N, 1, ' ', 1).
```

If $N \leq 0$, the Hollerith character count is set to one.

10. Input Record Length. To set the input record length parameters use:

CALL FFSET (10, LOPT, 2, ' ', 1).

LOPT(1) is the number of columns to be read from the input file ($\leq$132). LOPT(2) is the number of columns to be examined by FFIN2 ($\leq$LOPT(1)). If LOPT(1) $\leq$ 0, 132 columns will be read. If LOPT(2) $\leq$ 0, then LOPT(1) columns will be processed.

FFENV

FFENV will print the current environment (separator characters, options, etc.) on the output file. The calling sequence is:

CALL FFENV

Figure 1 is a copy of the output produced by FFENV for a typical condition.

```
FREE FORM INPUT ENVIRONMENT
     OUTPUT FILE=                          FILE 6
     ECHO FILE                               NO
     INTERACTIVE ERROR PROCESSING           YES
     NUMERIC FIELD CHECKING                 YES
     QUOTE CHARACTER=                         '
     LINE TERMINATOR CHARACTER=               $
     FFA CHARACTER COUNT=                     4
     NUMBER OF COLUMNS READ=                 80
     NUMBER OF COLUMNS PROCESSED=            80
     SEPARATORS( 3)=                          , =
```

Figure 1. Output From FFENV for a Typical Environment.

IFFIND and IFFINC

IFFIND and IFFINC are FUNCTION subprograms for use in identifying key words. The calling sequence is:

FUNCTION {IFFIND / IFFINC} (LARRY, N, ICHAR).

On return, IFFIN{D / C} = I if LARRY (I) = ICHAR, where LARRY is a one dimensional, integer (IFFIND) or character (IFFINC) array of length N. If ICHAR is not contained in LARRY, IFFIN{D / C} = 0. These functions can be used to identify key words as follows:

```
      CHARACTER*5 KEY (n), KWORD
      DATA KEY/'keyword 1','keyword 2',---/
       .
       .
       .
100   CALL FFC(KWORD, 1, 5, JRD)
      J = IFFINC (KEY, n, KWORD)
       .
       .
       .
      branch to syntax processing or error.
```

IBFIND and IBFINC

IBFIN{D / C} are FUNCTIONS similar to IFFIN{D / C} except that IBFIN{D / C} uses a binary search method. The calling sequence is:

FUNCTION IBFIN{D / C} (LARRY,N,ICHAR,NDEX).

LARRY, N, and ICHAR are as defined for IFFIN{D / C} . NDEX is a one dimensional array, NDEX(N). LARRY need not be arranged in the correct collating sequence by the user. This will be

done internally. On the first call to IBFIN{$^{D}_{C}$}, NDEX(1), must be equal to zero. This indicates to the program that LARRY must be indexed in NDEX according to the computer's collating sequence. IBFIN{$^{D}_{C}$}will call COLAT{$^{D}_{C}$} (see Section IV) to do this.

The LARRY array is not physically rearranged; NDEX is returned as a cross-reference index array. The values of NDEX must not be changed by the user. IBFIN{$^{D}_{C}$} returns the location of ICHAR in LARRY.

To use IBFINC in the example for IFFINC, add statements DIMENSION NDEX(n) and DATA NDEX(1)/0/. Change the "J=" statement to J = IBFINC (KEY,n,KWORD,NDEX).

SECTION IV

FFIN2 PROGRAMS FOR ADVANCED USERS

Several subprograms are available which permit the user to gain access to the contents of the internal buffer and to affect the sequence in which the FFIN2 programs process the fields contained on a record. Also, it is possible to use the error processing programs in FFIN2 to report user detected errors.

FFCPYO

Subroutine FFCPYO will copy a portion of the internal buffer out to a local CHARACTER variable. The calling sequence is:

SUBROUTINE FFCPYO(ILINE, NC, I1, I2).

ILINE is a CHARACTER variable of length NC. I1 and I2 are the first and last characters to be copied from the buffer. On return, ILINE(1:NC) contains buffer contents from character I1 to I2. I1 and I2 correspond to columns on the input record. If I1 is less than 1 or I1 is greater than MCOLRD*, no action is taken and no warning is given. If I2 is less than I1 or greater than MCOLRD*, I2 = MCOLRD is used. No more than NC characters will be copied into ILINE. FFCPYO permits the user to examine fields in the buffer.

* See Section V for MCOLRD.

FFCPYI

Subroutine FFCPYI will copy the contents of a local CHARACTER variable into the internal buffer. The calling sequence is:

SUBROUTINE FFCPYI(ILINE, NC, I1, I2).

ILINE is a CHARACTER variable of length NC containing information to be copied into the buffer. I1 is the first character in the buffer to be changed and I2 is the last. A maximum of I2 - I1 + 1 characters will be copied into the buffer. If I2 - I1 + 1 is less than NC, the excess in ILINE will be discarded. If it is greater, the excess in the buffer will not be altered.

If I1 is greater than I2, no copy will occur, and no warning will be given. If I1 is less than 1, FFCPYI will use I1 = 1. If I2 is greater than MCOLRD*, use I2 = MCOLRD. FFCPYI does not alter the current pointers to the buffer.

FFINI

Subroutine FFINI sets the internal pointers so that the internal buffer will be examined from its beginning. FFINI is useful when an input record is to be re-examined or when the buffer is loaded by the user. The calling sequence is:

SUBROUTINE FFINI(JRD,MTY).

JRD is the input file number. MTY is an output variable that is returned with the following values:

1 - the buffer is not empty. FFINI has terminated normally.

---

* See Section V for MCOLRD

2 - The buffer is empty (i.e., contains no data fields). Therefore, it cannot be positioned at its beginning. Upredictable results will occur if any FFIN2 program is subsequently called to examine the buffer. The options are to load the buffer again with FFCPY1 and call FFINI again or to call one of FFA, FFR, FFD, FFI, FFC, or FFL to cause a new record to be read from an input file.

FFINI will echo the contents of the buffer if option two is greater than zero.

FFINI is useful to examine records that the user has obtained or constructed in some way other than the usual method of reading from an external file. For example the following lines:

```
CHARACTER *80 STUFF
STUFF = 'THIS LINE IS USER GENERATED'
CALL FFCPYI (STUFF, 80, 1, 80)
CALL FFINI (5, MTY)
```

will insert STUFF into columns 1 through 80 of the internal buffer, and FFINI will set up FFIN2 to begin reading 'THIS'.

FFLAST

Subroutine FFLAST will return the start and end of the last processed field. The calling sequence is:

```
SUBROUTINE FFLAST(I1, I2).
```

I1 is the position of the first character in the field and I2 is the position of the last character. Quote characters will not be pointed to by I1 and I2. I1 and I2 may be used as

inputs to FFCPYO or FFCPYI, for example. Quote fields will automatically be taken care of internally.

If FFLAST is called after any of the input subprograms in Section II, I1 and I2 will point to the field that was just processed. If FFCLR was called immediately previous, then I1 = I2 = 0 will be returned. Zero values will also be returned if FFLAST is called immediately after a call to FFNEXT which causes a new record to be read. If it is called immediately after FFSKIP, I1 and I2 will point to the field which was skipped. FFNEXT and FFSKIP are discussed subsequently.

FFNEXT

Subroutine FFNEXT will return the start and end of the next field to be processed. The calling sequence is:

SUBROUTINE FFNEXT(I1, I2, IQUOT, JRD).

I1 and I2 will be returned as the first and last characters in the field that will be processed by the next call to an input subprogram. IQUOT will equal zero if the field is not in quotes and will equal one if it is. The characters pointed to by I1 and I2 are not quote marks. Characters I1 - 1 and I2 + 1 will be the quotes if IQUOT = 1. JRD is the input file. If the buffer is exhausted when FFNEXT is called, a new record will be read, and I1 and I2 will point to the first field on it.

FFSKIP

Subroutine FFSKIP will skip the next field in the buffer. The calling sequence is:

```
SUBROUTINE FFSKIP (JRD).
```

JRD is the input file. A call to FFSKIP will cause the internal pointers to be moved to the field following the next field. The skipped field becomes the last processed field and a call to FFLAST will return pointers to it.

If the current record is exhausted when FFSKIP is called, a new record will be read and the first field on it will be skipped.

FFRSET

Subroutine FFRSET will reset the internal pointers to a specified field. The calling sequence is:

```
SUBROUTINE FFRSET (I1, IERR).
```

I1 is the position in the buffer of the first character of the field the user wishes to be processed by the next call to the input subprogram. In other words, the pointers are set to process the field starting at I1 next. FFRSET will check to see if the I1 - 1 character is a quote and will process the field correctly if it is. The I1 character may not be a separator; it must be a valid field character.

IERR is an error indicator whose values have the following meaning:

1 - no error

2 - I1 $\leq$ 0 or I1 > 80

3 - The I1 character is a separator

4 - I1 is beyond the end of the record.

Internal pointers are not changed if IERR is greater than one.

FFPAC3

Subroutine FFPAC3 produces error messages and is discussed more fully in Section V where the calling sequence is given. The important point to note here is that the user may call FFPAC3 to identify a bad field. Output from FFPAC3 is shown in Figure 2 for an invalid numeric field.

```
IMPROPER NUMERIC FIELD
FREE FORM ERROR FIELD UNDERLINED

3.4E-.1
-------

CORRECTLY RE-ENTER BAD FIELD AND REMAINDER
```

Figure 2. Output from FFPAC3 for an Invalid Numeric Field

FFPAC3 output begins with the message "FREE FORM ERROR FIELD UNDERLINED." The previous message indicating a bad numeric field is printed prior to the call to FFPAC3. The user can print such messages, too, and then call FFPAC3 to underline the detected error. Use a FORMAT statement of the following form to be compatible:

```
FORMAT (5X, 'user error message')
```

Normally, FFPAC3 will prompt the user to re-enter the record starting at the bad field, e.g., "CORRECTLY RE-ENTER...." If the user prefers alternate processing, this prompt can be suppressed by sending IPRT > 1 to FFPAC3.

For example, suppose an improper menu selection is made during interactive input. The programmer may want to have the operator immediately try again, or he may want to print the menu again before reading a new menu selection. Figure 3 shows how to accomplish each of these cases.

In Case 1, the field is recovered with FFLAST and the error underlined with FFPAC3. IPRT = 1 is sent to FFPAC3, so the operator is prompted to correct his error and a transfer is made to statement 100 to try again. In Case 2, IPRT = 2 is sent, so the operator is not prompted for input in FFPAC3. After return from FFPAC3, the menu is printed again and then the transfer is made to statement 100.

```
100  CALL FFA (IL, 1, JRD)
       .
       .
       .
     IF(bad menu selection) GO TO 200
       .
       .
       .
```

Case 1

```
200 WRITE (6,900)
900 FORMAT (5X, "IMPROPER SELECTION")
    CALL FFLAST (I1, I2)
    CALL FFPAC3 (1, I1, I2, JRD)
    GO TO 100
```

Case 2

```
200  WRITE (6,900)
900  FORMAT (5X, "YOU ARE CONFUSED")
     CALL FFLAST (I1, I2)
     CALL FFPAC3 (2, I1, I2, JRD)
       .
       .
     print menu again
       .
     GO TO 100
```

Figure 3. Example of Use of FFPAC3 to Generate a User Error Messag

FFMTY

Subroutine FFMTY permits the user to determine if the buffer is exhausted. The calling sequence is:

```
SUBROUTINE FFMTY(MTY).
```

MTY is returned as 1 if more data is in the buffer. It is returned as 2 otherwise. FFMTY is useful in processing syntax which has optional fields at the end of a line. It can be used to determine when a call to an input routine will cause a new record to be read.

COLATC and COLATI

Subroutines COLAT{C/I} will order a one dimensional integer or CHARACTER array in least to greatest collating order. The calling sequence is:

```
SUBROUTINE COLAT{C/I} (ICOL,NROW,NDEX).
```

ICOL is an array dimensioned to ICOL(NROW). The NDEX array is dimensioned NDEX(NROW), and is returned as an index array for ICOL. ICOL is not physically rearranged.

Let JNEW be the rank of an entry in ICOL in the collated ordering.

Then:

NDEX(JNEW) is the physical location in ICOL.

COLAT{C/I} is not intended for frequent sorts of long arrays. It uses a simple bubble sort and will be inefficient if used for repeated sorts.

FFREAD

Subroutine FFREAD loads new records into the internal buffer. FFREAD is user written. The calling sequence is:

SUBROUTINE FFREAD (JRD,MCOL,LINE,IEOF).

JRD is a user specified, non-positive code, LINE(MCOL) is a CHARACTER *1 array and IEOF is an end-of-file indicator. If the user calls an FFIN2 program with JRD.GT.0, FFREAD is not called; FFIN2 reads records from file JRD. If JRD.LE.0, however, FFIN2 calls FFREAD when a new record is needed. It is the user's responsibility to load LINE with the desired data. If IEOF is returned as 2, FFIN2 end-of-file processing will be performed and FFREAD will be called again. FFIN2 sets IEOF = 1 and LINE = blank on input. On return, the contents of line will not be examined if IEOF = 2.

The user may need to provide alternate modes of loading the buffer. For example, input from a graphics terminal may require special subprogram calls rather than a READ statement. Also, the user may wish to perform certain error checking on the input record before allowing FFIN2 to examine it. All of this can be done in FFREAD.

To provide alternate processing, the user writes FFREAD, loads it with his programs, and calls FFIN2 programs with non-positive file numbers. The supplied version provides a guide for re-writing FFREAD. See FFINI for another method to utilize alternate processing.

NOTE

No FFIN2 programs which might cause a new record to be read should be called from FFREAD since these programs might cause an illegal circular call to FFREAD.

FFPAC5

FFPAC5 checks logical and numeric fields for validity. It is more fully discussed in Section V. It can be used to "look ahead" at the next field to determine if the field is numeric or not. Consider the following statements

```
NCODE=5
CALL FFNEXT(I1,I2,IQUOTE,JRD)
CALL FFPAC5(NCODE,I1,I2,ICOND)
IF(ICOND.EQ.0)  THEN
  CALL  FFI(IVAR,JRD)
ELSE
  alternate processing
ENDIF
```

NCODE=5 (see FFPAC) specifies an integer field. FFNEXT delineates the next field. FFPAC5 examines the field to determine if it is an acceptable integer field. If so, ICOND is returned as zero and the field is read with FFI. If not some alternate processing is performed.

Note that if option 4 (see FFSET) is set to zero, FFPAC5 will be called again for the field when FFI is called.

FFSEP

FFSEP will return the first non-blank separator (FNBS) before or after the next field to be read. The calling sequence is:

```
SUBROUTINE FFSEP(IFB,SEP,ICOND)
```

IFB is input by the user. If IFB = 1, FFSEP returns the FNBS preceding the next field. If IFB = 2, it returns the FNBS after the field. SEP is a CHAPACTER variable returned as the non-blank separator or as blank if there is none. ICOND is output as the condition detected by FFSEP.

ICOND =
- 1 - no unusual condition
- 2 - IFB = 1 and there is no previous field. If separators precede the next field, SEP = FNBS in these separators. If not, SEP = FNBS following the last field of the last record (SEP = blank if the last record was cleared with FFCLR).
- 3 - IFB = 1 and the current record is exhausted (i.e., there is no next field). SEP = FNBS on the remainder of the field or blank if none.
- 4 - IFB = 2 and the record is exhausted. SEP = blank.
- 5 - IFB = 2 and the next field is the last. SEP = FNBS at the end of the record.

Unpredictable results occur if FFSEP is called immediately after a call to FFRSET.

EXAMPLES

Consider the following record with separators "1", "2", "3", and "4".

1 FIELD1 2 FIELD2 3 FIELD3 4

Case	Next Field to be read	IFB	SEP	ICOND	Description
1	FIELD2	1	"2"	1	normal
2	FIELD2	2	"3"	1	normal
3	FIELD1	1	"1"	2	FIELD1 is the first field on the record and separators precede it.
4	record exhausted (i.e. FIELD3 has been processed already)	1	"4"	3	SEP is the FNBS in the remainder (after FIELD3) of the record
5	record exhausted	2	blank	4	no records remain so it is not meaningful to request the FNBS after the next field
6	FIELD3	2	"4"	5	FFIN2 examines the remainder of the record for the FNBS

FFGET

FFGET will return options and current conditions for the FFIN2 programs. The calling sequence is:

SUBROUTINE FFGET(IOPT,LOPT,NOPT,COPT,NCR)

IOPT is the option code input by the user and has one of the following values.

1 - return output file number

2 - return echo file number

3 - return error processing flag (0- interactive,
1 - batch)

4 - return numeric field validity checks (0 - on, 1 - off)

5 - return quote character

6 - return line terminator character

7 - return the number of separator characters

8 - not used (return with no action)

9 - return the Hollerith character count

10 - return input record length parameters
(1) - number of columns to be read
(2) - number of columns to be examined

-1 - return all separator characters

-2 - return the column position in the buffer of the
last non-separator (zero if the buffer is empty)

-3 - return the column position of the first character
of the next field (zero if the buffer is empty)
(see also FFNEXT)

-4 - return selected separators. See LOPT and COPT.

-5 - return the file number of the file last read from.

LOPT is a one dimensional, integer array to hold integer ouptut values. If only one value is returned, LOPT may be dimensioned to one. For IOPT=-4, the user must input the desired separator numbers in LOPT. For example, if LOPT(I) = 2, then COPT (I) will equal the second separator on return.

COPT is a one dimensional character array to hold output characters. It needs to be dimensional to only one if the option returns only one character value.

NOPT and NCR are the number of elements in LOPT and COPT, respectively.

IFFCHS

IFFCHS will determine if a particular character is a separator. The calling sequence is:

```
FUNCTION IFFCHS(CHS)
```

CHS is a character value on input. The values of IFFCHS are:

IFFCHS = 0 - if the first character of CHS is not a separator

= n - if the first character of CHS is separator n.

FFLNS

FFLNS will reset the internal pointer to the last non-separator character in the buffer (see COMMON variable NFF). No action is taken if the buffer is empty, otherwise, FFLNS will examine the buffer and set NFF to the last non-separator character position in the buffer. It does not affect the pointer to the current field.

SECTION V

FFIN2 INTERNAL DOCUMENTATION

The internal buffer and all other permanent information is stored in two labeled COMMON areas called FFCOM and FFCOMC.

The COMMON Area

The variables in COMMON and their usage are explained below.

IFF and NFF are pointers to the buffer array LINE. LINE (IFF) is the first character of the next field if there is one in this record. If the buffer has been exhausted,then IFF = NFF + 1. LINE (NFF) is the last available non-separator character in the record. In other words, it is the last data character of the last field on the card. If a call is made to an FFIN2 program requesting input and IFF > NFF, then a new record is read. Otherwise, the next field on the existing record is interpreted.

IF1 and IFL are pointers to the most recently processed field on the record if there is one. LINE (IF1) is the first character of the field and LINE(IFL) is the last character. If a new record has just been read, IF1 = IFL = 0.

JECHO is the echo file unit number. If JECHO $\leq$ 0, no echo of data is performed.

JRDOLD is the input file unit number from the last call to an FFIN2 input program. A call to an FFIN2 input program with a new JRD causes the buffer to be cleared and forces reading a new record from the new input file.

NCNT is a counter for input records. If the echo option is on, NCNT is printed when the record is printed on the echo file.

IERROR is the indicator for error processing. If IERROR = 0, interactive operation is assumed. If IERROR $\neq$ 0, batch error processing is done.

ICHK is the indicator for numeric field validity checking. If ICHK = 0, fields are checked. If ICHK $\neq$ 0, no checking is done.

NSEP is the number of separators in use.

LTERM is the current line terminator character.

LQUOT is the current quote character.

LBNK is the blank character.

NOWCHR is the current Hollerith character count.

NWRDS is the number of words, at NOWCHR characters per word, to hold MCOLRD characters. NWRDS is calculated in FFSET.

NNSEP is the length of the LSEP array.

LSEP is an array whose first NSEP locations contain the current separator characters.

LTEMP is a CHARACTER variable. FFIN2 packs the fields from LINE into LTEMP. The character capacity of LTEMP must

be sufficient to hold at least 20 characters. A capacity of MXCOL characters is recommended.

LNUM is an array which holds all characters that are valid in a numeric field. LNUM is used primarily when ICHK = 0.

IFMT is a CHARACTER variable which holds a character alphanumeric representation of a format statement for processing FFA.

LINE is the buffer array.

MXCOL is the length of the array LINE.

MCOLRD is the number of columns that are read from the input file.

MCOLFF is the number of columns examined by FFIN2 for valid data.

PSEP is the first non-blank separator following the last field processed. Blank if none.

FFIN2 Internal Subprograms

There are several subprograms internal to the FFIN2 which the user normally is not concerned with. These are described below.

SUBROUTINE FFPAC is callled by FFA, FFL, FFR, FFD, FFC, and FFI. On return from FFPAC, the appropriate field is packed into LTEMP. To accomplish this, FFPAC calls subroutines FFPAC1, FFPAC2, and FFPAC4. The calling sequence is:

SUBRCUTINE FFPAC (NCODE, JRD)

where:

NCODE = identifier for calling program

1 = FFA

2 = FFL

3 = FFD

4 = FFR

5 = FFI

6 = FFC

JRD = input file number.

SUBROUTINE FFPAC1 processes new records. If a current record is exhausted (IFF.GT.NFF) then a new record is read and examined to set IFF and NFF properly. If the current record is not exhausted, an immediate RETURN is executed. In either case, on return, LINE (IFF) is the first character of the next field, and LINE (NFF) is the last character of the last field of the current record. The calling sequence for FFPAC1 is:

SUBROUTINE FFPAC1 (JRD).

SUBROUTINE FFPAC2 will delineate the next field. The calling sequence is:

```
SUBROUTINE FFPAC2 (IFRST, ILAST, JRD)
```

where

IFRST	-	on return, LINE (IFRST) is the first character (not a quote) of the next field.
ILAST	-	on return, LINE (ILAST) is the last character (not a quote) of the next field.
JRD	-	input file number.

Also, FFPAC2 will reset IFF to point to the first character of the field following the one delineated by IFRST and ILAST. If there is none, then IFF will be set to NFF + 1 so that the next call to FFPAC1 will cause a new record to be read.

SUBROUTINE FFPAC3 performs standard error processing. The calling sequence is:

```
SUBROUTINE FFPAC3 (IPRT, I1, I2, JRD)
```

IPRT - 1=prompt for bad field, 2=do not prompt

I1 - LINE (I1) is the first character of an erroneous field.

I2 - LINE (I2) is the last character of an erroneous field.

JRD - input file number.

FFPAC3 will print the current record and underline the bad field on the output file and the echo file (if one exists). If batch error-processing is in effect, a STOP statement will be executed. Otherwise, if IPRT = 1 the user will be asked to re-enter the erroneous fields. If IPRT = 2 no prompt will be issued. This feature is for users who may wish to call FFPAC3 directly. See Section IV for such applications.

SUBROUTINE FFPAC4 will pack the field into LTEMP. The calling sequence is:

```
SUBROUTINE FFPAC4 (IFRST, ILAST, NCODE, JRD)
```

where

IFRST and ILAST = values set by FFPAC2

NCODE = identifier for calling program (see FFPAC)

JRD = input file number.

For alphanumeric fields, the field is packed up to 132 characters starting at the left. Any excess is discarded with no error message. For logical fields, only

the first non-blank character is packed into LTEMP. Twenty characters are packed for numeric fields. Leading blanks are filled if the field is less than 20 characters, and an error is generated if a numeric field has more than 20 characters.

SUBROUTINE FFPAC5 checks logical and numeric fields for errors. The calling sequence is:

SUBROUTINE FFPAC5 (NCODE, IFRST, ILAST, ICOND)

where

NCODE = identifier for calling program (see FFPAC)

IFRST and ILAST = values set by FFPAC2

ICODE = 0 if the field is acceptable
1 if the field has errors.

FFPAC5 is called for all calls from FFL. The first non-blank character of the field is checked to determine if it is a "T" or "F." If not, ICOND = 1 is returned.

FFPAC5 does validity checking for numeric fields and is called for this application only if ICHK = 0.

SUBROUTINES FFLOA{C/D} load an array with a single value. The calling sequence is:

SUBROUTINE FFLOA{C/D}(LARRY, N, ICHAR).

On return, the first N words of the one dimensional array LARRY are equal to ICHAR. LARRY and ICHAR are integer for FFLOAD and character for FFLOAC.

SECTION VI

CHANGES NEEDED TO RUN ON OTHER COMPUTER SYSTEMS

The FFIN2 programs are written in ANSI FORTRAN 77 (see FFA for the only exception). No changes should be required to run on any compiler which supports FORTRAN 77.

The control data FTN5 (FORTRAN 77) compiler does not process the END parameter on READ statements in the usual manner. On non-CDC systems, delete the statement X = EOF(JRD) from SUBROUTINE FFPAC1.

DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

APPENDIX W

MOPADS FINAL REPORT:

DOCUMENTATION MANUAL FOR THE AN/TSQ-73 SYSTEM MODULE

TABLE OF CONTENTS

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology and Abbreviations

BN	Battalion
Q-73	AN/TSQ-73
TD	Tactical Director
TDA	Tactical Director Assistant
TD1, TD2	The operator of a Battalion AN/TSQ-73

I. OVERVIEW OF THE MOPADS MODEL

This report documents the implementation details of the IHAWK system module. It's intended audience is the MOPADS modeler who must maintain and explain the model. A companion report (Goodin & Polito (1983)) provides user information for the MOPADS user.

The MOPADS operator tasks are modeled with subnetworks of MSAINT task nodes. Each node has a subroutine of user code associated with it where the user may write code to access the data base and modify the MSAINT data structure. These task node subroutines are named according to the task node they represent. For instance, the subroutine used to process logic associated with node 175 is named T175Q (T175G is Group Q-73) and the one for node 62 is named T62Q (T62G if Group Q-73).

The two operators in the Group Q-73 and the two operators in the Battalion Q-73 perform various tasks in the networks. The Group Q-73 network is a subset of the Battalion Q-73 network. The Group Q-73 operators perform some of the same operator tasks as the Battalion Q-73 TD (Tactical Director) operator. The operators in the Group Q-73 will do all their communications with themselves and their Battalion Q-73. A Group Q-73 will not communicate with a Battalion Q-73 that it is not controlling. The Battalion Q-73 will communicate with the Group Q-73 controlling it and the fire units under its control. This communication is carried out by sending messages, altering the data base, altering the data structure, or signaling (clearing an operator from one task node to another).

Table I-1 is a list of some of the assumptions made in constructing the MOPADS model of the Battalion and Group Q-73's.

Table I-1. Assumptions.

- Jamming and ECM are not considered.
- All setup, checkout and emplacement procedures have been performed.
- The Q-73 will not be mobile during the simulation.
- All ADP's remain operational assuming the Q-73 has not been destroyed.
- The ATDL data link is used for communicating with other AN/TSQ-73s and the Q-73's fire units.
- The TDA will perform all IFF procedures.
- A target will be positively identified if the TDA performs IFF procedures on that target.
- The CEASE FIRE command will not be modeled.
- TD will make all assignments to fire units.
- All Battalion Q-73s will have one TDA operator and one TD operator.
- All Group Q-73s will have two TD operators.
- When a Battalion Q-73 identifies a target the information is sent to all its fire units.

II. MODEL DESCRIPTION FORMS

1-0 ENTITIES

The operators that perform the work in the Battalion and Group Q-73's are the entities in their respective system modules. The forms in this seciton describe each entity, their attributes, and their resource requirements.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
31(BN) 31 (GRP) 32 (GRP)	TD-Officer for handling all engagement and FU assignment tasks (operator types 1(BN), 8(GRP), and 9(GRP))	1	Operator ID	-	
		2	Copy Row Number	-	
		3	Operator Type (1-TD, 2-TDA)	-	
		4	Current Track Column Pointer (Ø if none)	-	
		5	Internal Mark Time (For Op.Task Time Statistics)	0.0	
32(BN)	TDA-Handles tracking and identification tasks (Operator type = 2)	6	Self-Clearing Indicator (Ø-No Self Clearing >Ø-Node to Clear to)	0	
		7	Last Task Node Branched from (set and used only at release times)	-	
	NOTE: Both TD's and TDA's have the same IA definitions	8	Hooking Indicator (Ø-None, Branch to hooking, >Ø-Task Node to branch back to)	0	
		9	Item to be hooked (<Ø site or FU; =CRN; =Ø No hooking;>Ø Track column pointer)		

Name: Riley Goodin
Date: 11/15/33

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

SAINT ENTITIES

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
		10-12	Unused currently	-	
		13-14	Branching (Task Node Specific uses)	-	
		15	Information passed in or out of task sequencing (operator task specific uses)		

Name: Riley Goodin
Date: 11/15/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

SAINT ENTITIES

2-0 RESOURCES

There were no MSAINT resources used in modeling the Battalion and Group Q-73's.

3-0 VARIABLES

The forms in this section describe all the state variables, system attributes,and user variables used in the Q-73 system modules. Table II-1, shown after the forms, expands the definition for the variable TSVALQ. Note that no state variables or system attributes are used in the Q-73 modules.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
MXTSKQ	Maximum number of operator tasks for the AN/TSQ-73 (Batallion). This is a FORTRAN PARAMETER.	30	-
TSVALQ (MXTSKQ,3)	Infomation is passed back from task sequencing to MSAINT task networks through TSVALQ. For example,if task J is selected by task sequencing, the information for task J is passed back in row J of TSVALQ (see Table II-1)	-	COM1Q
MCOLQ	Maximum number of messages any operator can have simultaneously pending.	10	Parameter
MSIDQ(5, MCOLQ)	Each column is the ID of a message pending for the operator doing task sequencing.	-	COM1Q
MSAAQ(2, MCOLQ)	Each column is the data base address of a message pending for the operator doing task sequencing.	-	COM1Q
THRETQ(3,2)	Data on the tracks found to be most threating for goals 1 (column 1) and 2(column 2)	-	COM1Q

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X USER VARIABLES

NAME: Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 Battalion
PROJECT: MOPADS

SAINT VARIABLES

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
THRETQ(3,2) (continued)	row 1 - NTRACK colum of the track row 2 - track status (see column 5 of Track data) row 3 - Time till arrival		

TYPE OF VARIABLES: __ STATE VARIABLES __SYSTEM ATTRIBUTES X_USER VARIABLES

NAME: Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 Battalion
PROJECT: MOPADS

Table II-1.

DEFINITION OF TSVALQ VALUE (TSVALQ(MXTSKQ,3))			
TASK	COLUMN 1	COLUMN 2	COLUMN 3
1	-	-	-
2	-		-
3	Message number (1-hold fire, 3-cease engage)	Copy row number of FU	NTRACK column of the track to cease(negative if secondary assignment)
4	0	Message DB address, word 1	Message DB address, word 2
8	NTRACK column of a track to identify	-	-
10	NTRACK column of covered track to engage	NFSTOR column of cover FU	0
15	NTRACK column of track to assign (negative means change targets first)	NFSTOR column of fire unit to assign it to (positive is fire seciton A, negative is fire section B)	Assignment type, (1-primary, 2-secondary, -1 -cover)
20,22,26	0	Message DB address word 1	Message DB address, word 2

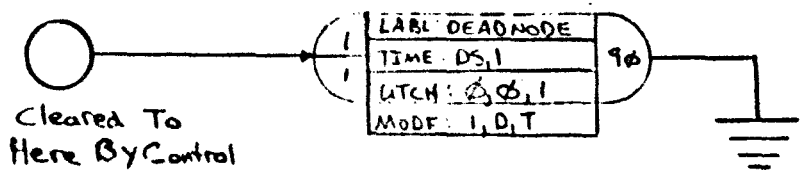
4-0 TASK NETWORKS

In this section are the forms used to document each of the operator tasks and each individual node in those tasks. Each operator task is documented by two types of forms: MSAINT TASK MODELS and MSAINT TASK NETWORKS. The MSAINT TASK MODEL forms given a written description of the activities involved at each node on the network. The MSAINT TASK NETWORKS forms contain the MSAINT subnetworks used to describe each operator task. Following these two forms for each task is the TASK NODE SPECIFIC DATA forms for each node in the operator task (MSAINT subnetwork). These forms contain the information stored in the data base for each node.

Note that the source nodes, the dead node, and the task sequencing subnetwork are also contained in this section. These particular nodes are not shown on the MSAINT TASK MODEL forms because there are no operator actions involved.

Note that some of the task nodes and task numbers are specific to the Battalion Q-73 (denoted by (BN) after the number), and some are specific to the Group Q-73 (denoted by (GRP) after the number).

Table II-2, at the end of this section, is a cross-reference table that may be used to look up which operator tasks are associated with which nodes.



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS
DEAD NODE	TD or TDA	

NSAINT TASK NETWORKS

TASK NODE: 90

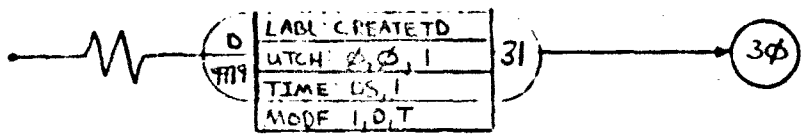
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

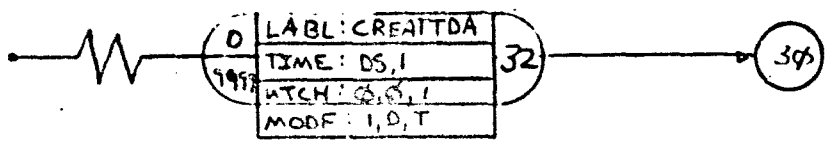
TASK NODE NUMBER	TASK LABEL	NOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
90	DEADNODE	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: ~~NOPADS~~

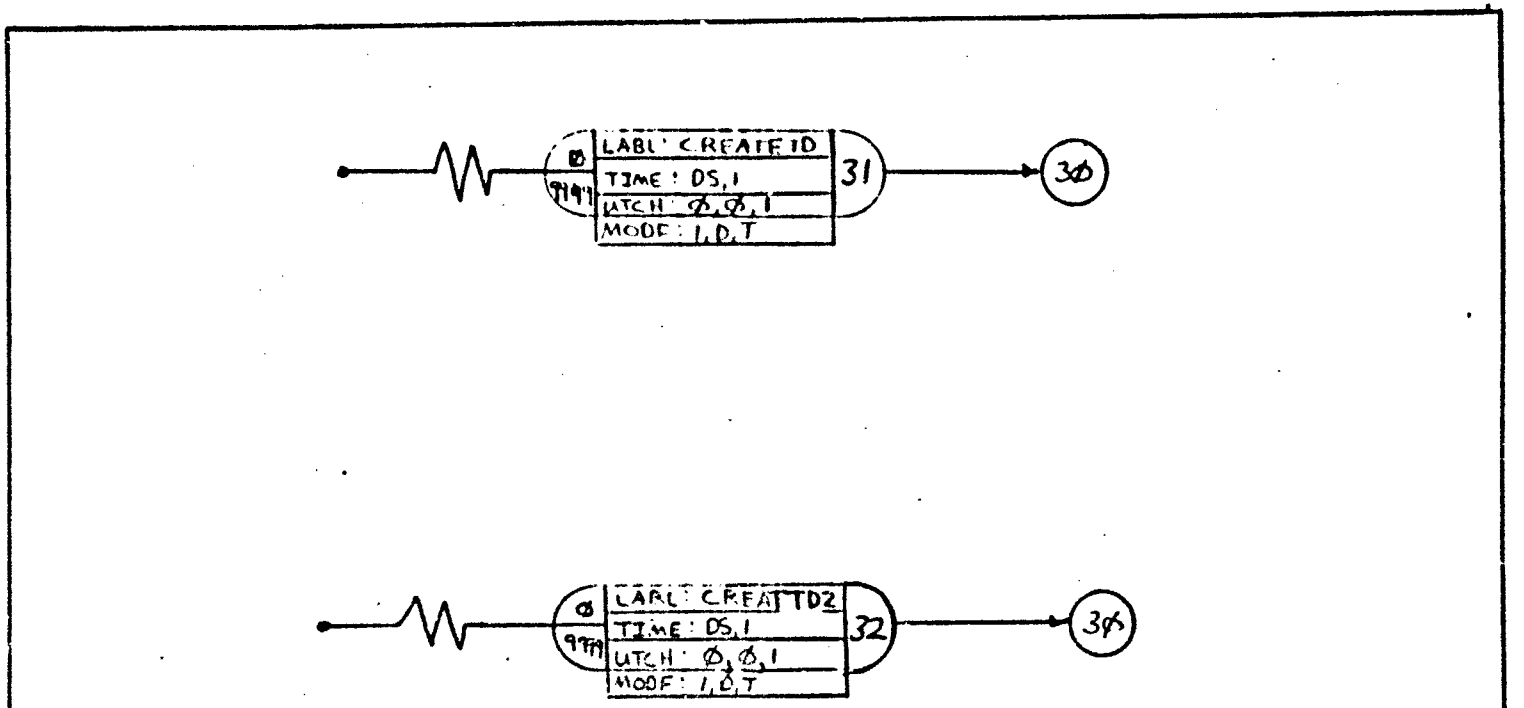
TASK NODE SPECIFIC DATA





TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
SOURCE NOTES(Battalion)	TD/TDA	PROJECT: MOPADS

MSAINT TASK NETWORKS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE:	AN/TSQ-73
SOURCE NODES(GROUP)	TD	PROJECT:	MOPADS

MSAINT TASK NETWORKS

TASK NODE: 31

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
31	CREATETD	-	(above)	(above)	(above)	1.0

NAME: J. Wiley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 32

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
32 (BN)	CREATTDA	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 32

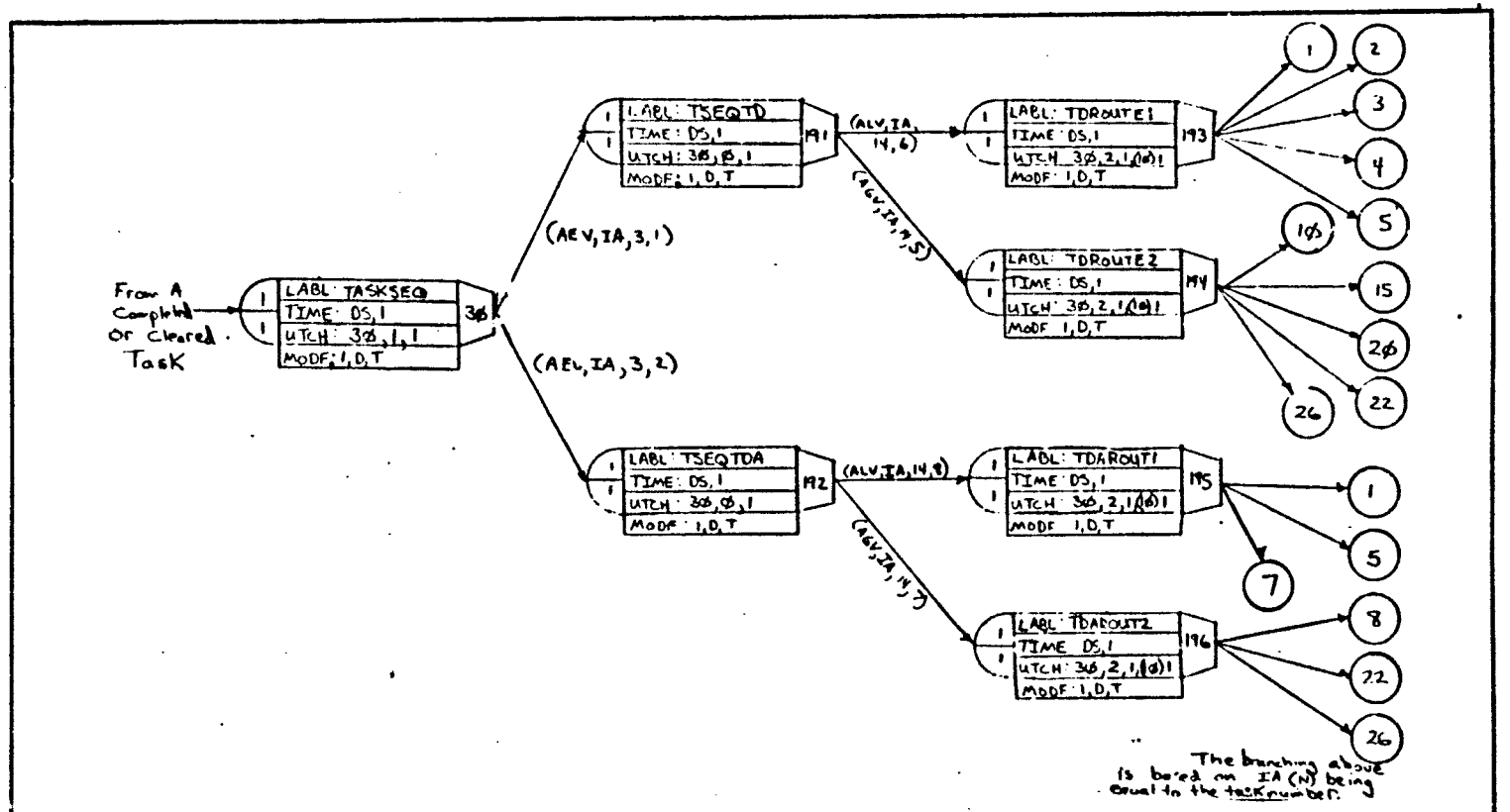
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
32 (GRP)	CREATTD2	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

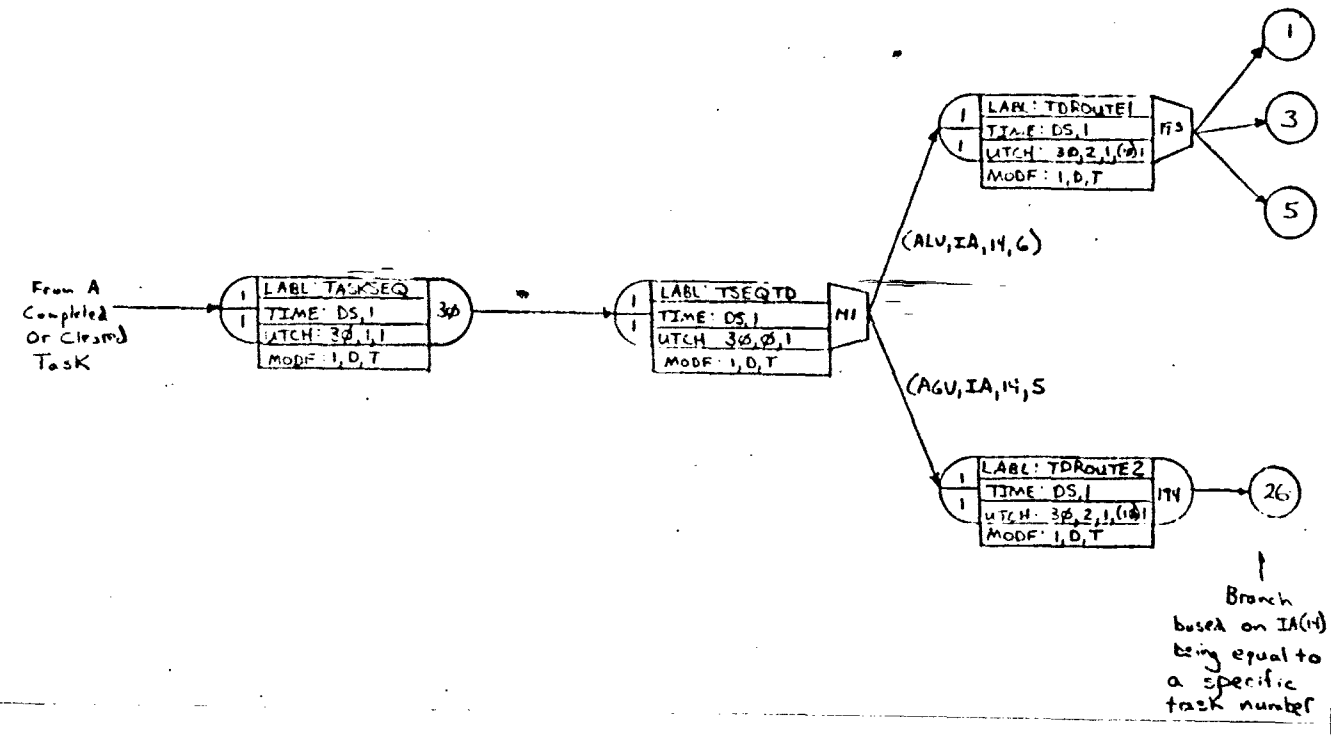
AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS
30) Task Sequencing (Battalion)	TD/TDA(Battalion)	

MSAIHI TASK NETWORKS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
30) Task Sequencing (Group)	TD1 or TD2	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 30

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+C0
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
30	TASKSEQ	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 191

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
191	TSEQTD	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 192

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV	DISTRIBUTION TYPE	
192(BN)	TSEQTDA	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 193

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
193	TDROUTE1	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 194

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
194	TDROUTE2	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 195

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
195(BN)	TDAROUT1	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 196

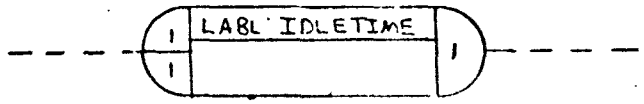
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
196(BN)	TDAROUT2	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodlin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1.



Operator scans the PPI, then the ARO, until either some command is received or he finds something to act on (new track, alerts, etc.)

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
1) Scan the Screen, Displays, etc.	TD and TDA Battalion or Group)	None

NAME: Riley Goodin DATE: 11/28/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

NSAINT TASK MODELS

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS
MSAINT TASK NETWORKS
TASK NAME & NUMBER
OPERATOR
1)Scan the Screen, Displays,etc.
TD or TDA of Battalion or Group
LARL: IDLETIME
TIME: 15.7
LATCH: 1.3.1
1
1
1
1
30
193
195

TASK NODE: 1

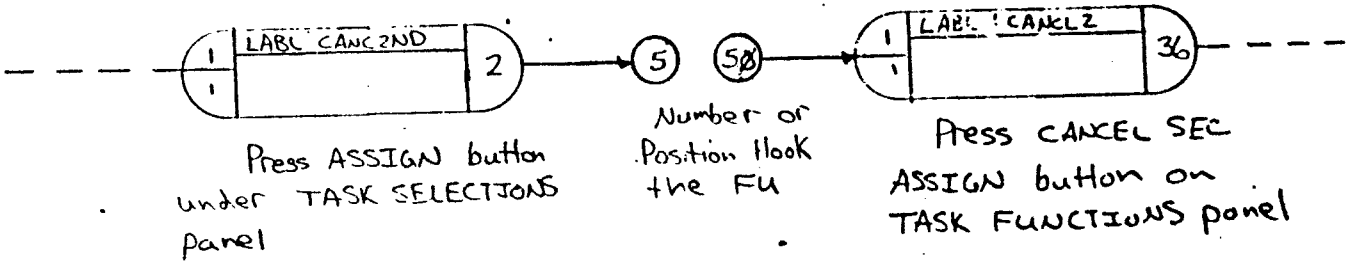
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	2.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	40.00000
SKILL-INDEX	11.00000
SKILL-WEIGHT	10.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
1	IDLETIME	1	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

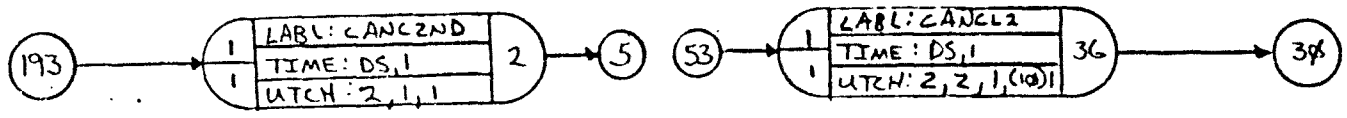
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
2) Cancel Secondary Assignments (BN only)	TD	TM 9-1430-652-10-3, Figure 4-10.1

NAME: Riley Goodin DATE: 11/30/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
2) Cancel Secondary Assignments (BM)	TD	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 2

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
2	CANC2ND	2 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 36

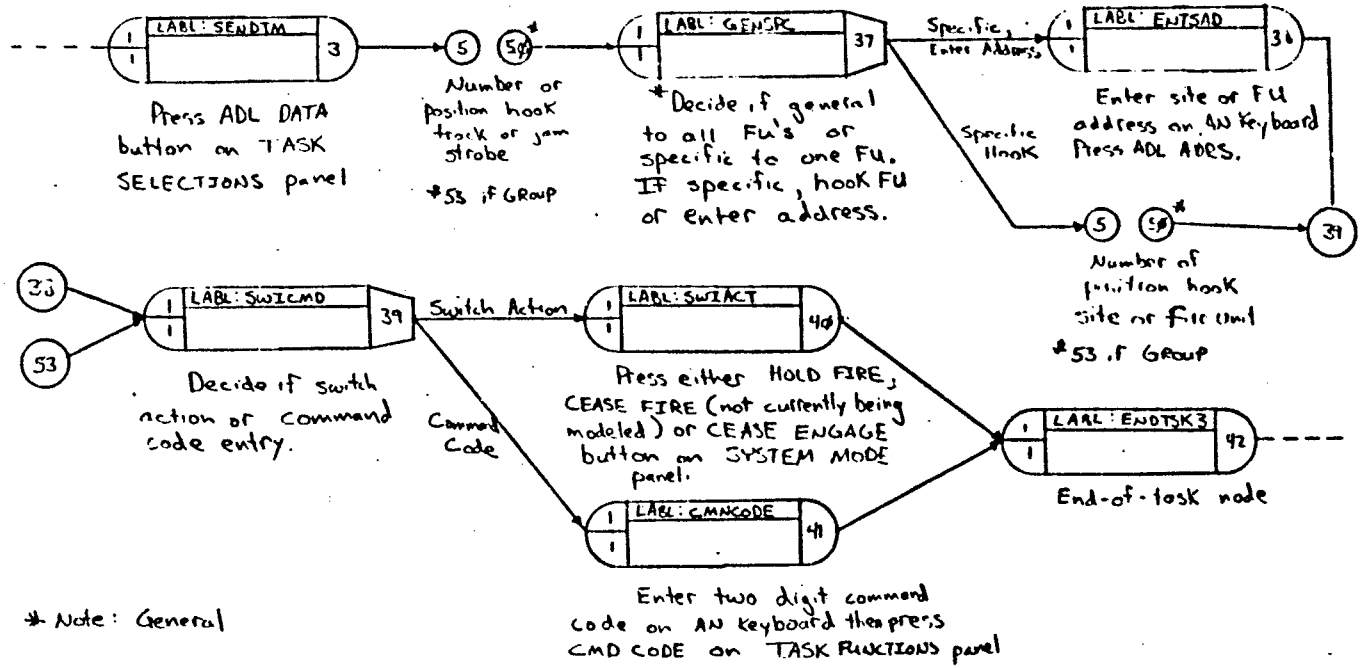
DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
36	CANCL2	2 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

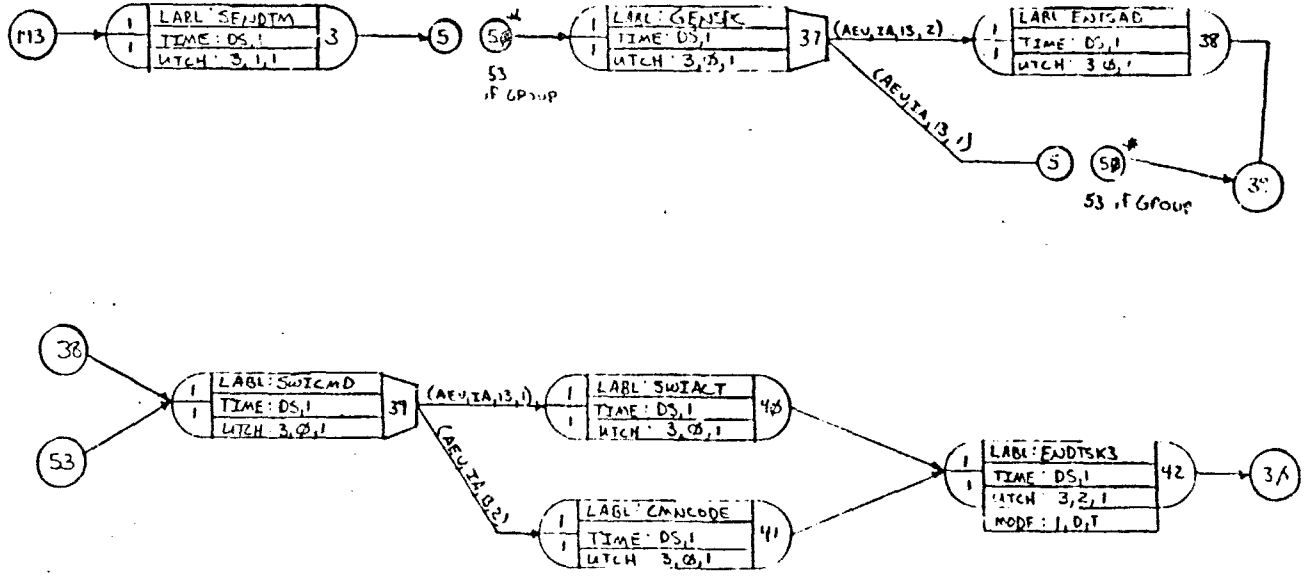
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
3) Send Terminate Commands	TD	TM 9-1430-652-10-3, Figure 4-38

NAME: Riley Gocdin DATE: 12/1/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
3) Send Terminate Command	TD	PROJECT: MOPADS

HSAINT TASK NETWORKS

TASK NODE: 3

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
3	SENDTM	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 37

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
37	GENSPC	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 38

DISTRIBUTION-TYPE	8.000000
MEAN	0.8330000E-01
STANDARD-DEVIATION	0.2920000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	90.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	10.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
38	ENTSAD	3	(above)	(above)	(above)	2.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 39

DISTRIBUTION-TYPE	9.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
39	SWICMD	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 40

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
40	SWIACT	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 41

DISTRIBUTION-TYPE	8.000000
MEAN	0.2167000
STANDARD-DEVIATION	0.7500000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	80.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
41	CMNCODE	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 42

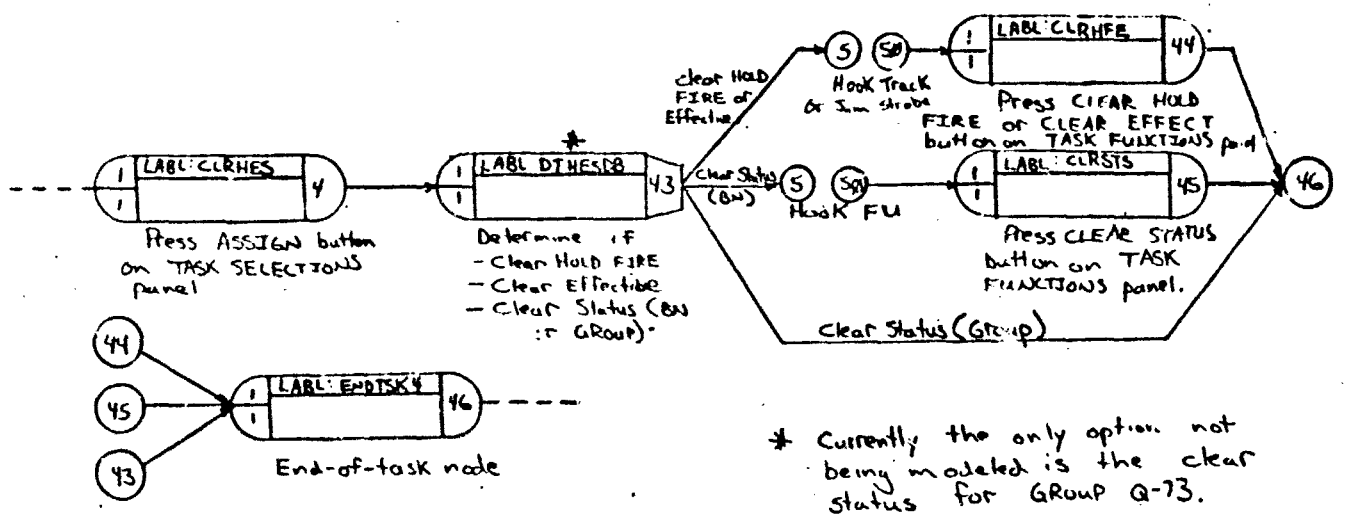
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
42	ENDTSK3	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MOBILE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

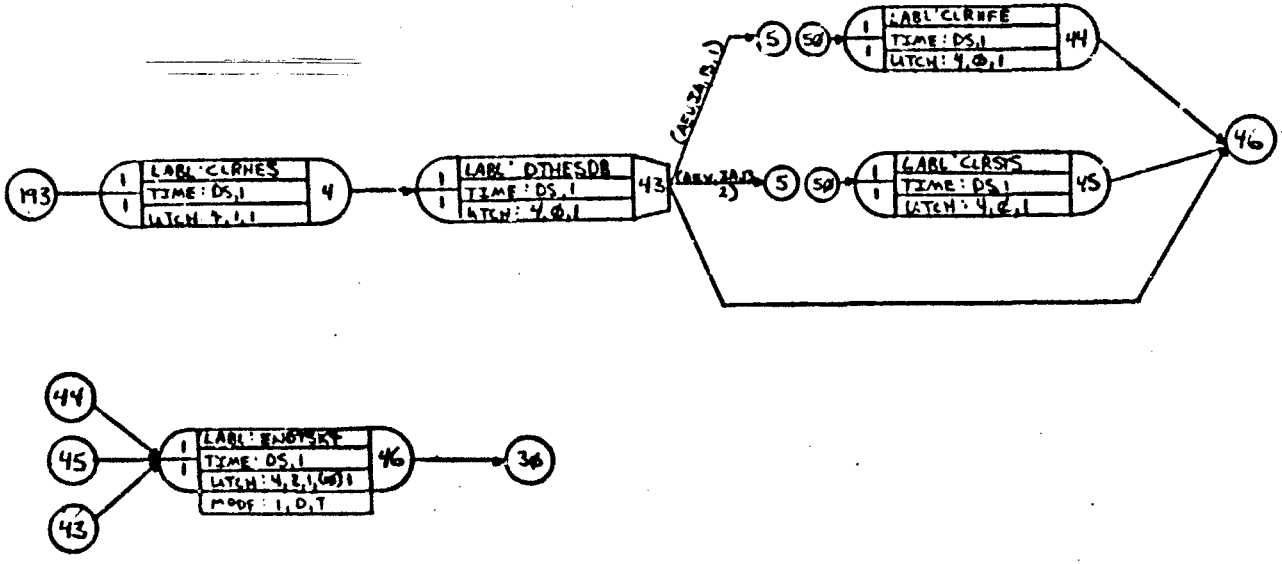
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
4) Clear Hold Fire,Effective,or Status (BATTALION only)	TD	TM 9-1430-652-10-3,Figure 4-12

NAME: Riley Goodin DATE: 12/5/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
4)Clear Hold Fire Effective Status	TD	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 4

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
4	CLRHES	4 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 43

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
43	DTHESDB	4 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 44

DISTRIBUTION-TYPE	8.000000
MEAN	0.2500000E-01
STANDARD-DEVIATION	0.8750000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
44	CLRHFE	4(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 45

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION MEAN	STD.DEV.	DISTRIBUTION TYPE	TASK ELEMENT ERROR FACTOR
45	CLRSTS	4(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 46

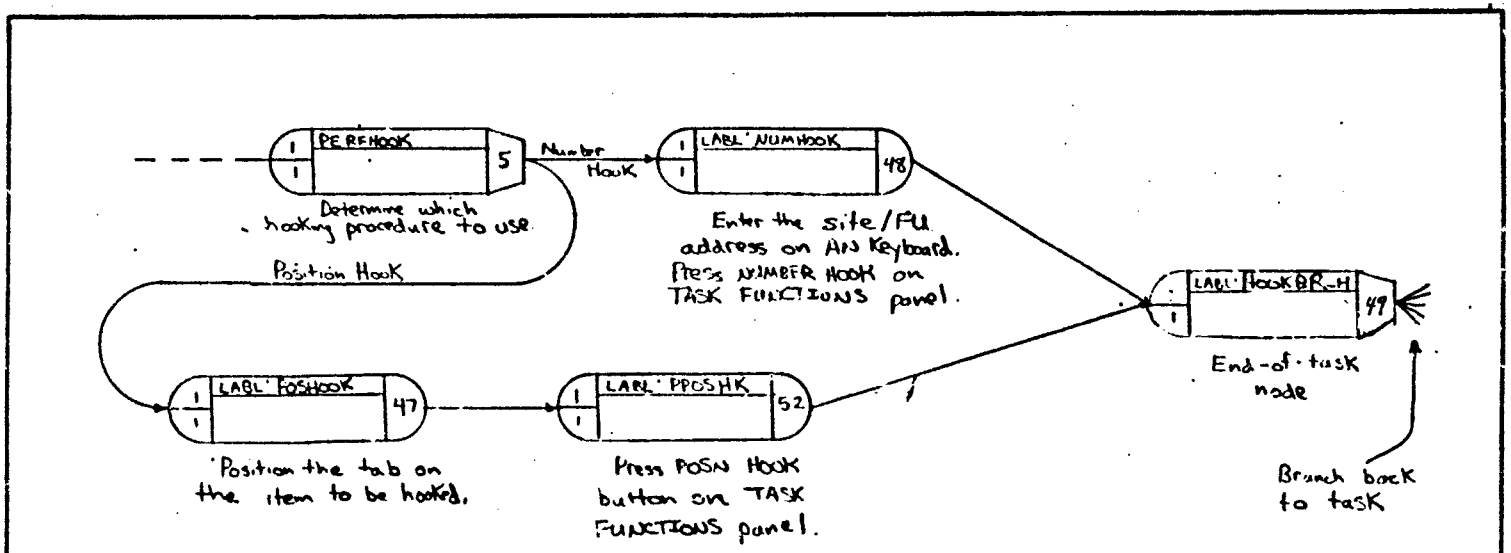
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
46	ENDTSK4	4	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

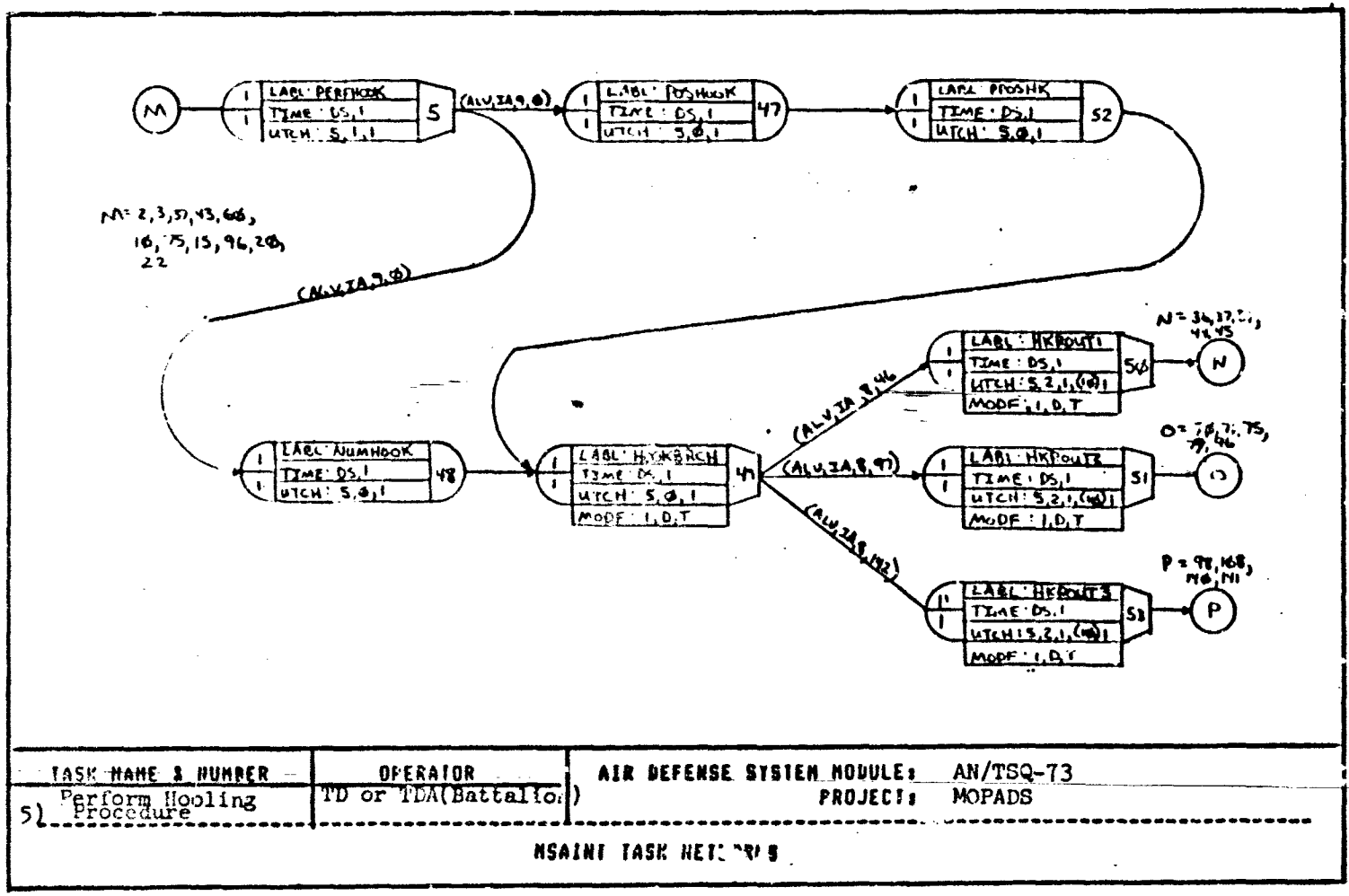
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
5) Perform Hooking Procedure	TD or TDA (Group or Battalion)	TM 9-1430-652-10-3, Fig. 4-13,14

NAME: Riley Goodin DATE: 11/30/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

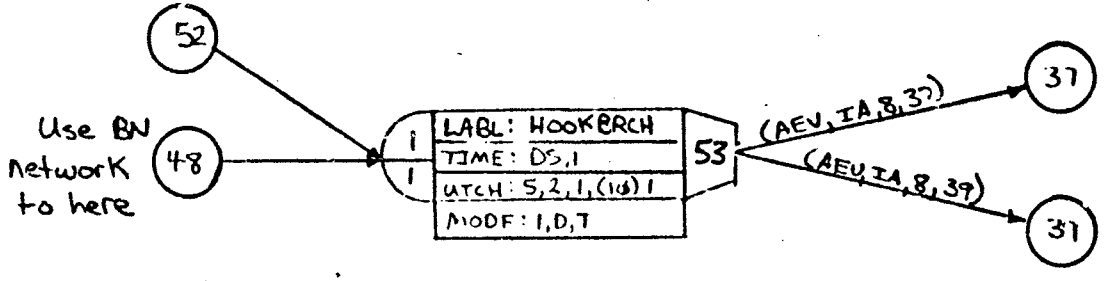
NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
51 Perform Hooling Procedure	TD or TDA(Battalio.)	PROJECT: MOPADS

NSAINT TASK NET'S

CHANGES TO TASK #5 NETWORK FOR GROUP Q-73



The only differences between BN Q-73 Task #5 and Group Q-73 Task #5 are the absence of nodes 50, 51, 52 and M is equal to 3 and 37 only (task node branched from into this task)

TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS
5) Perform Hooking Procedure	TD (Group)	

MSAINT TASK NETWORKS

TASK NODE: 5

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
5	PERFHOOK	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 47

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
47	POSHOOK	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 48

DISTRIBUTION-TYPE	8.000000
MEAN	0.7500000E-01
STANDARD-DEVIATION	0.2630000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	85.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	15.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
48	NUMHOOK	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 49

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	NOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
49 (BN)	HOOKBRCH	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: NOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 50

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
50 (BN)	HKROUT1	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 51

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
51(BN)	HKROUT2	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 52

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
52	PPOSHK	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 53

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STN.DEV.	DISTRIBUTION TYPE	
53 (BN)	HKROUT3	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 53

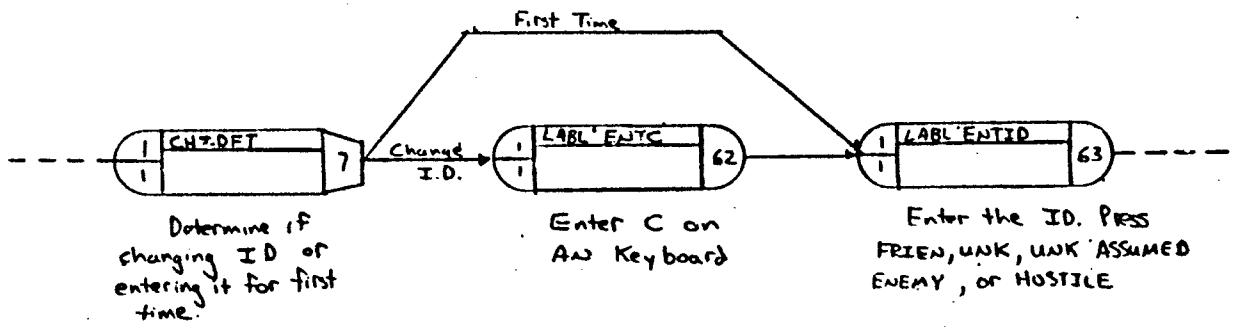
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSERVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
53(GRP)	HOOKBRCH	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

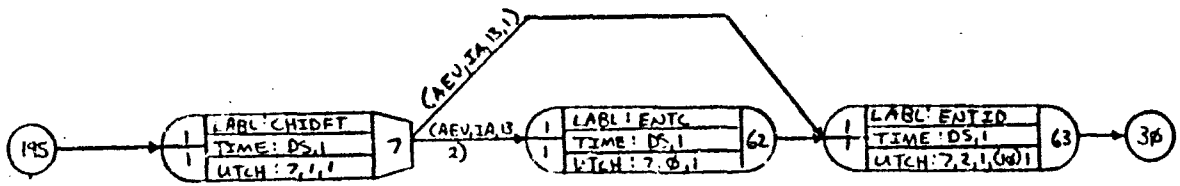
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
7) Enter ID Data	TDA	TM 9-1430-652-10-3, Figure 4-29

NAME: Riley Goodin DATE: 12/6/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS
7) Enter ID Data	TDA	

MSAINT TASK NETWORKS

TASK NODE: 7

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
7	CHIEF	7 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 62

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
62	EWTC	7(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 63

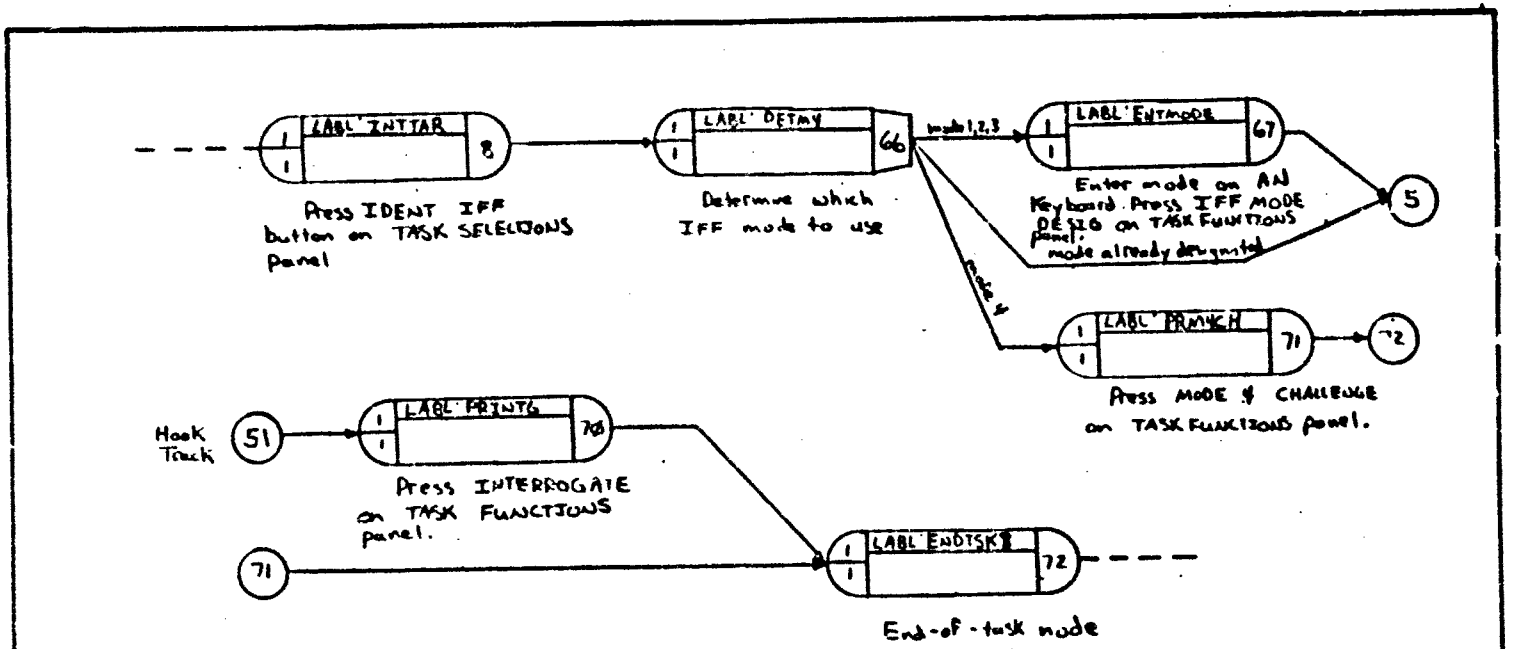
DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	30.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
63	ENTID	7(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

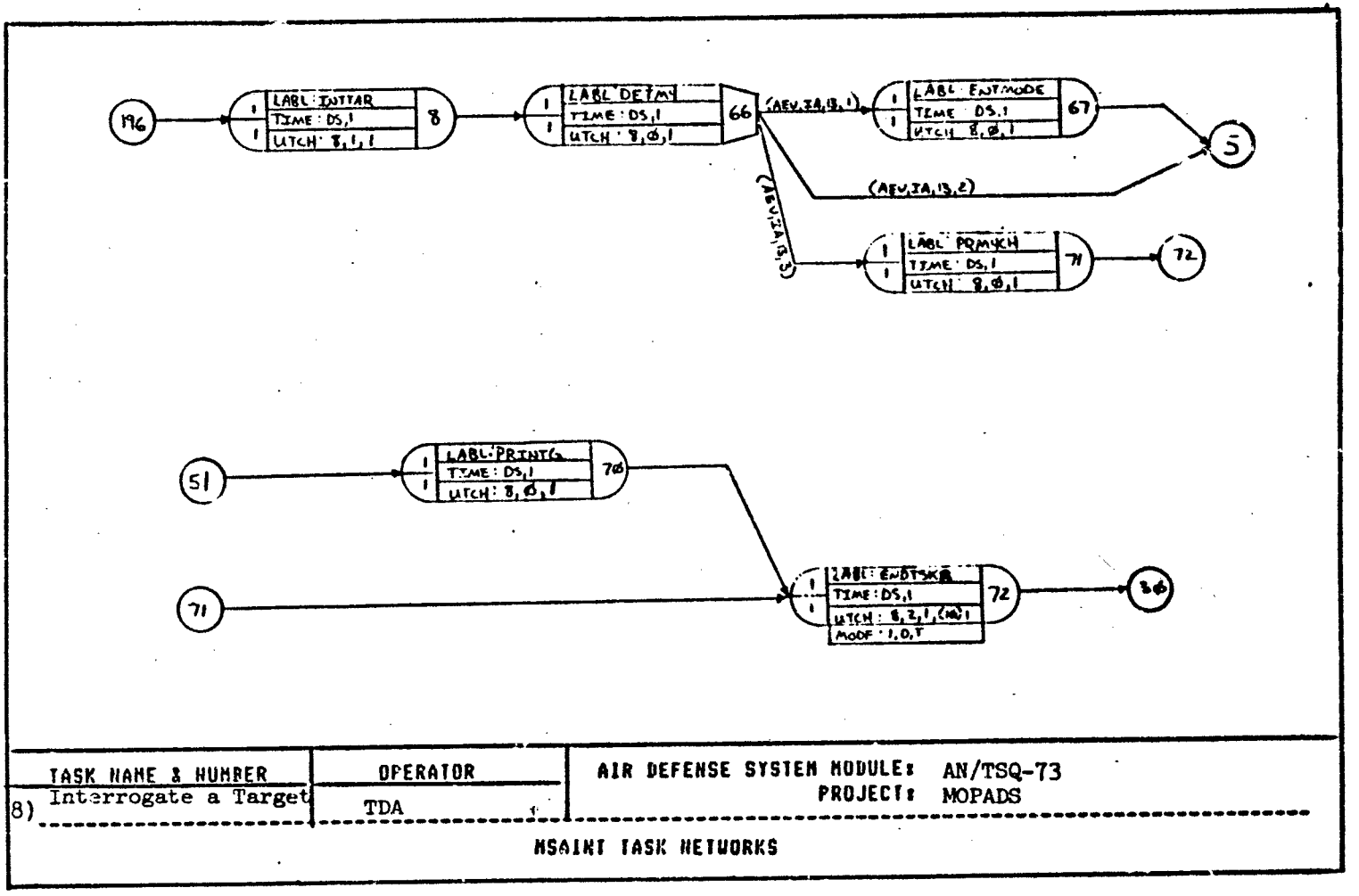
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
8) Interrogate a Target (Battalion Only)	TDA	TM 9-1430-652-10-3, Figure 4-39

NAME: Riley Goodin DATE: 12/6/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
8) Interrogate a Target	TDA	PROJECT: MOPADS

NSAINT TASK NETWORKS

TASK NODE: 8

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION MEAN	STD.DEV.	DISTRIBUTION TYPE	TASK ELEMENT ERROR FACTOR
8	INTTAP	8 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 66

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	4.000000
SKILL-WEIGHT	70.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
66	DETM4	8(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 67

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
67	ENTMODE	8(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 70

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
70	PRINT6	8 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 71

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
71	PRM4CH	8(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 72

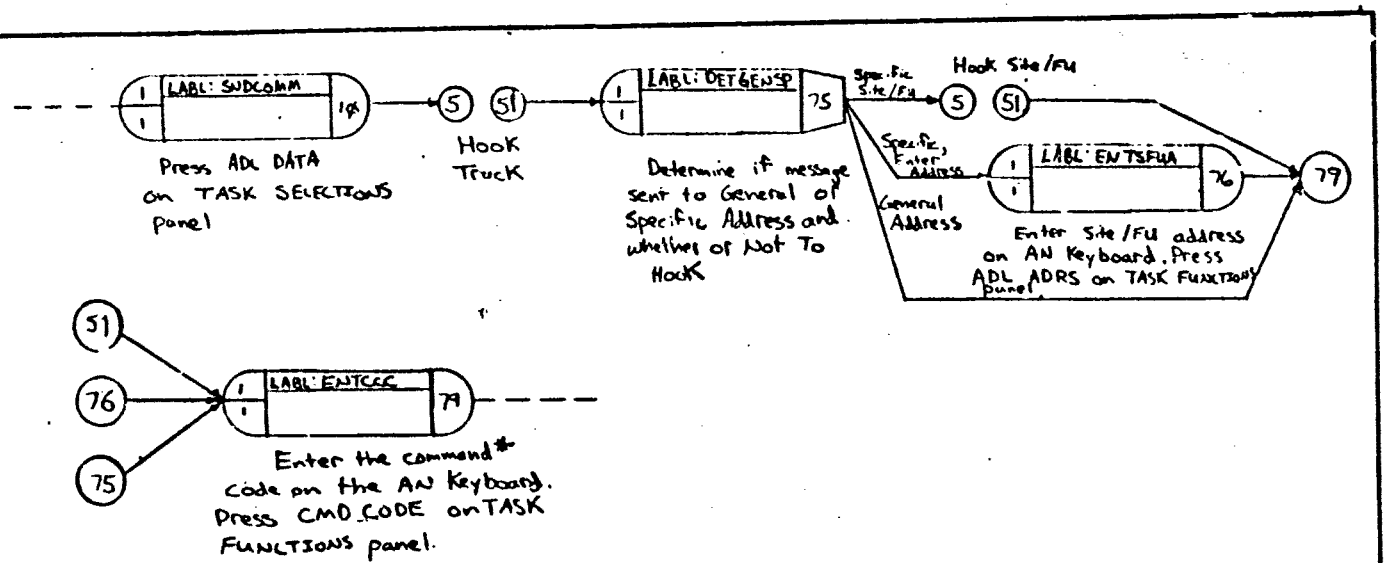
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MCPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
72	ENDTSK8	8(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

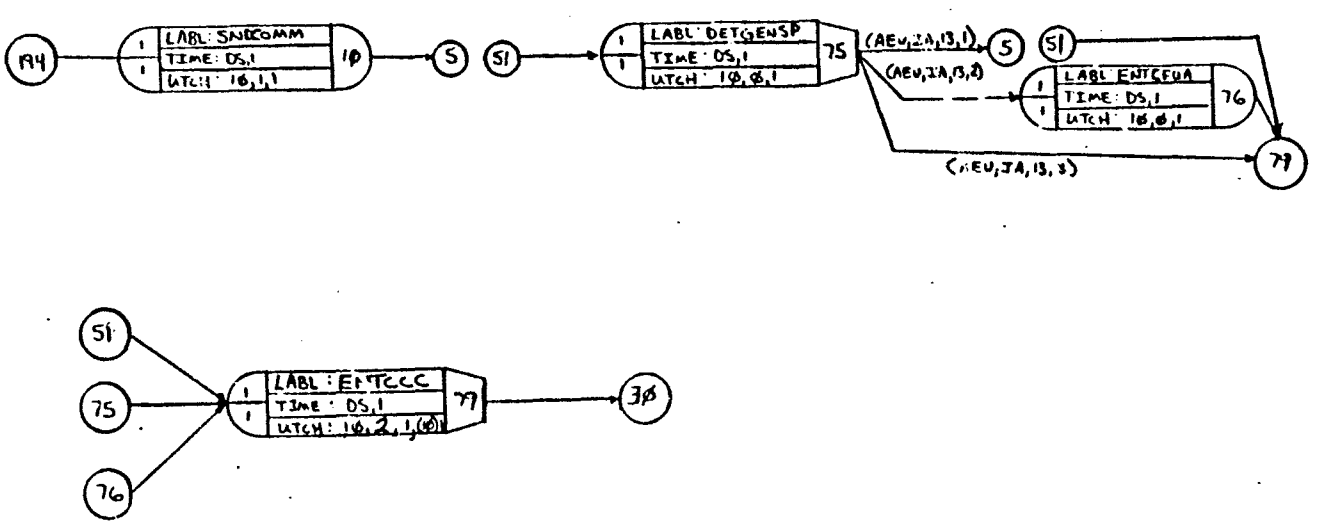


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
10) Send Command Message (Battalion Only)	TD	TM 9-1430-652-10-3,Figure 4-52.1

NAME: Riley Goodin
DATE: 12/5/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
10) Send Command Message	TD	PROJECT: MOPADS

USAINT TASK NETWORKS

TASK NODE: 10

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
10	SNDCOMM	10 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Coodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 75

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
75	DETGENSP	10 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 76

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
76	EMTSFUA	10(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 79

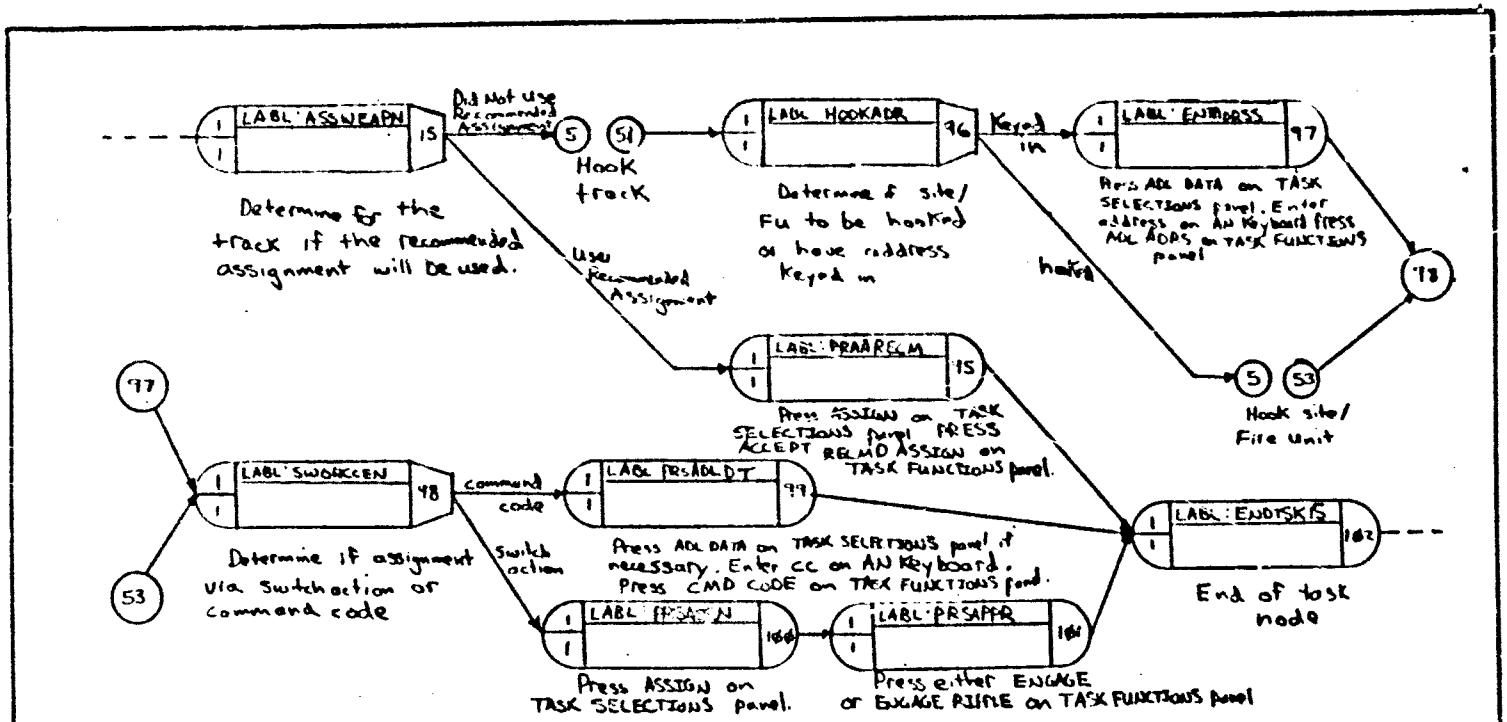
DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPABS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
79	ENTCCC	10(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPABS

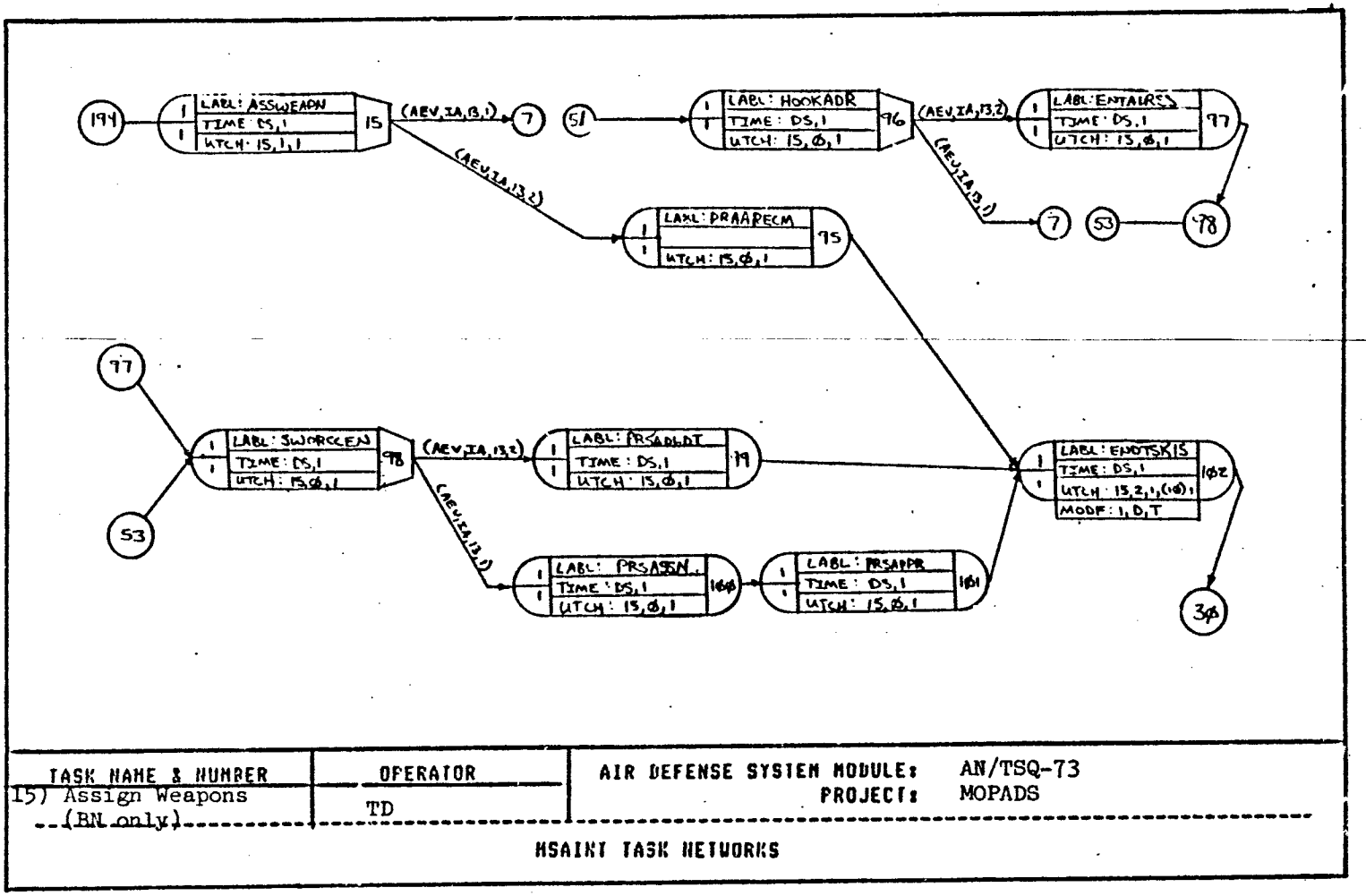
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
15) Assign Weapons (Battalion Only)	TD	TM 9-1430-652-10-3, Figure 4-10

NAME: Riley Goodin DATE: 11/30/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAIN: TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
15) Assign Weapons (BN only)	TD	PROJECT: MOPADS

HSAINT TASK NETWORKS

TASK NODE: 15

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	7.000000
SKILL-WEIGHT	20.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	40.00000
SKILL-INDEX	11.00000
SKILL-WEIGHT	40.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
15	ASSWEAPN	15 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 95

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES--OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
95	PRAARECM	15 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 96

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	65.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	35.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
96	HOOKADR	15(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 97

DISTRIBUTION-TYPE	8.000000
MEAN	0.1000000
STANDARD-DEVIATION	0.3500000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
97	ENTADRSS	15(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 98

DISTRIBUTION-TYPE	8.000000
MEAN	0.6300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
98	SWORCCEN	15 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 99

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
99	PRSADLDT	15 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 100

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
100	PRSASSN	15(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 101

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
101	PRSAPPR	15(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 102

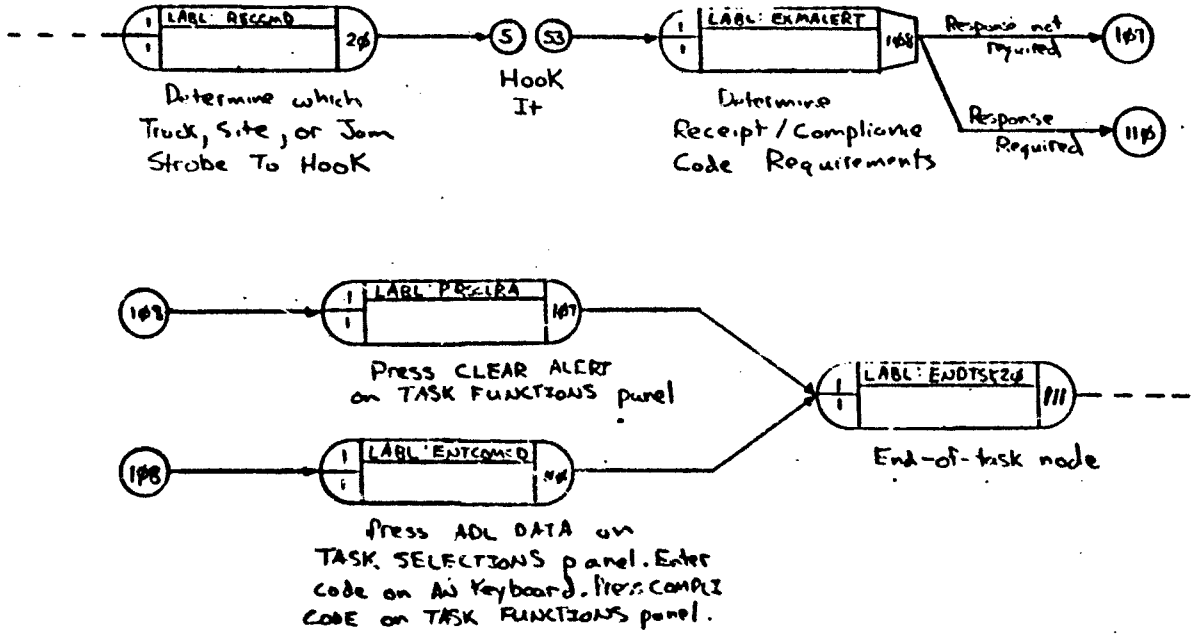
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
102	ENDTSK1	13 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

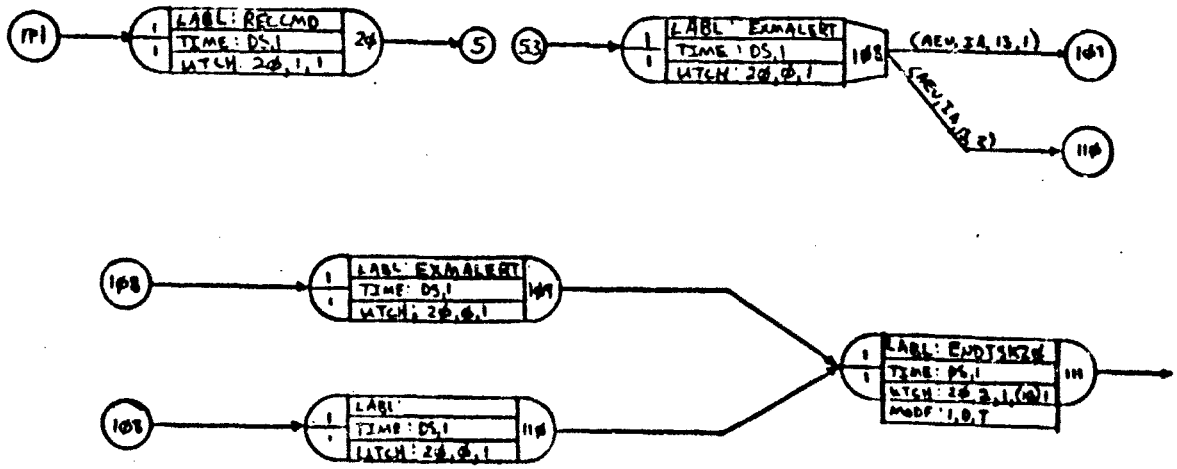


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
20) Receive Command (Battalion only)	TD	TM 9-1430-652-10-3,Figure 4-50

NAME: Riley Goodin
DATE: 12/8/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
20) Receive Command	TD	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 20

DISTRIBUTION-TYPE	8.000000
MEAN	0.3200000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	4.000000
SKILL-WEIGHT	60.00000
SKILL-INDEX	19.00000
SKILL-WEIGHT	40.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
20	RECCMD	20 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 108

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	7.000000
SKILL-WEIGHT	70.00000
SKILL-INDEX	19.00000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TAEM ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
108	EXMALERT	20 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 109

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
109	PRSCLRA	20(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 110

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
110	ENTCOMCI	20(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 111

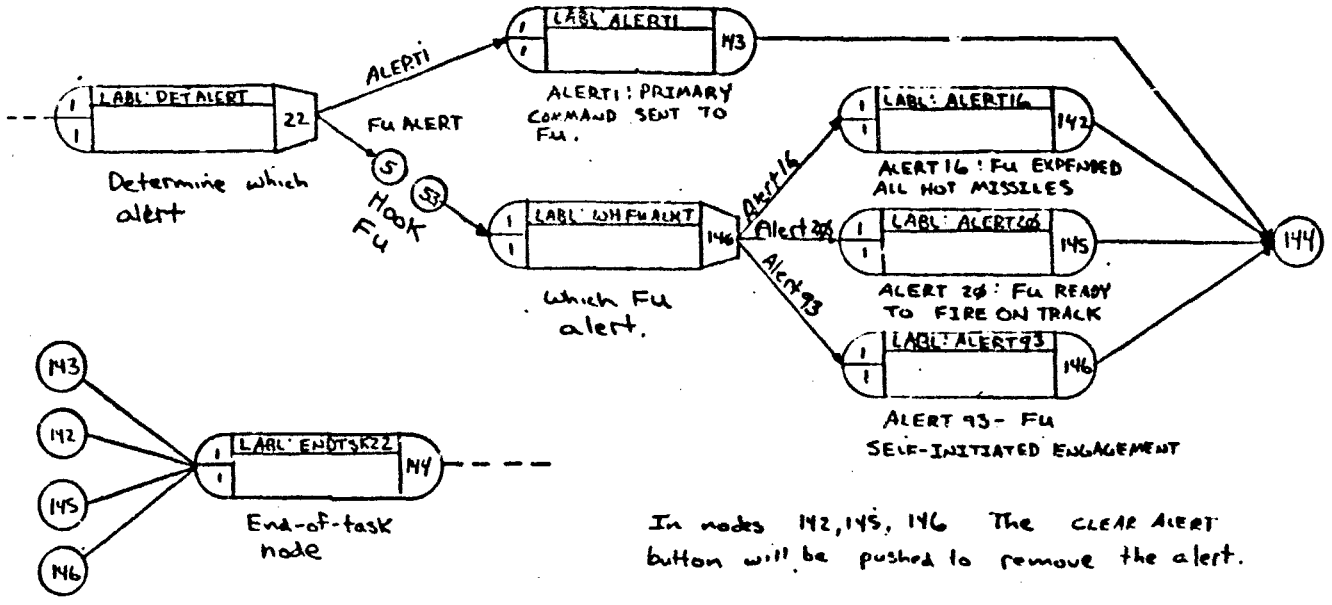
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
111	ENDTSK20	20(BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

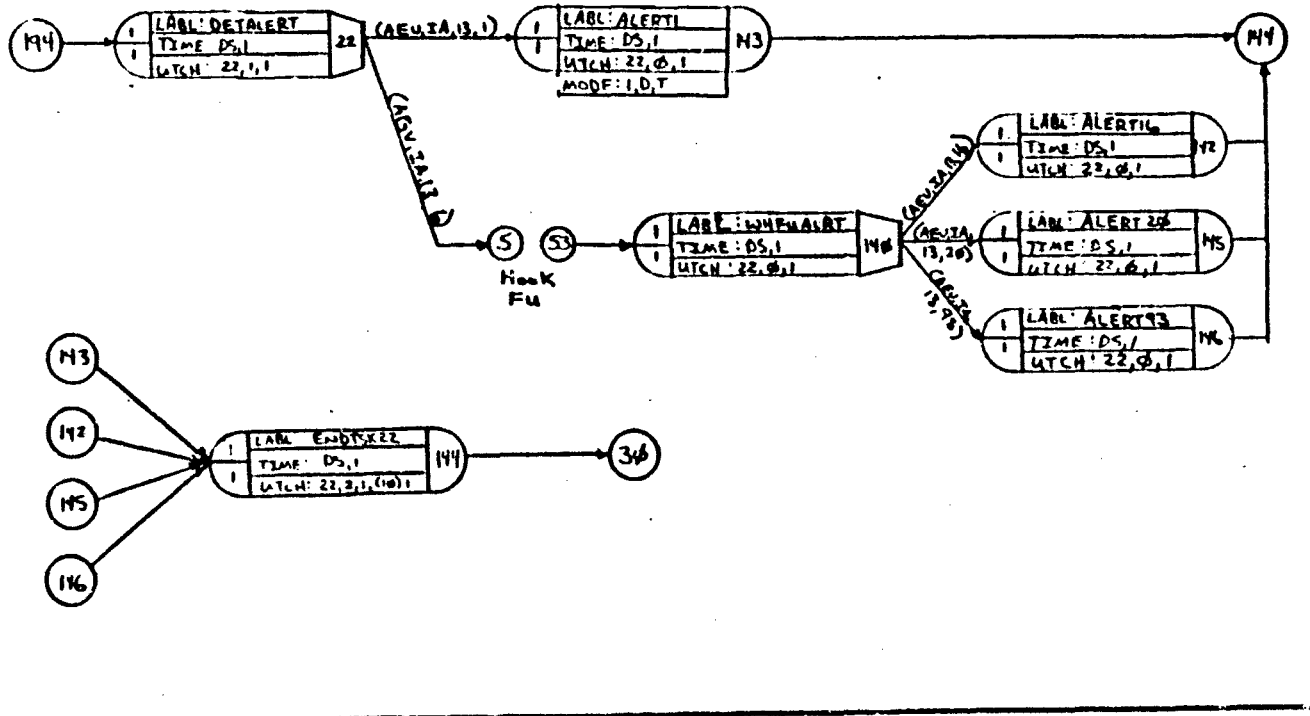


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
22) Clear Alerts (Battalion only)	TD	TM 9-1430-652-10-3,Figure 4-7

NAME: Riley Goodin
DATE: 12/6/83

AIR DEFENSE SYSTEM MODULE: AN/TSQ-73
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS
22) Clear Alerts	TD	

NSAIHI TASK NETWORKS

TASK NODE: 22

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	9.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	19.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
22	DETALERT	22	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 140

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	9.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	19.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
140	WHFUALRT	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 142

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
142	ALERT16	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 143

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
143	ALERT1	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 144

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
144	ENDTSK22	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 145

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEM^	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION: MEAN	TASK TIME INFORMATION: STD.DEV.	TASK TIME INFORMATION: DISTRIBUTION TYPE	TASK ELEMENT ERROR FACTOR
145	ALERT20	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 146

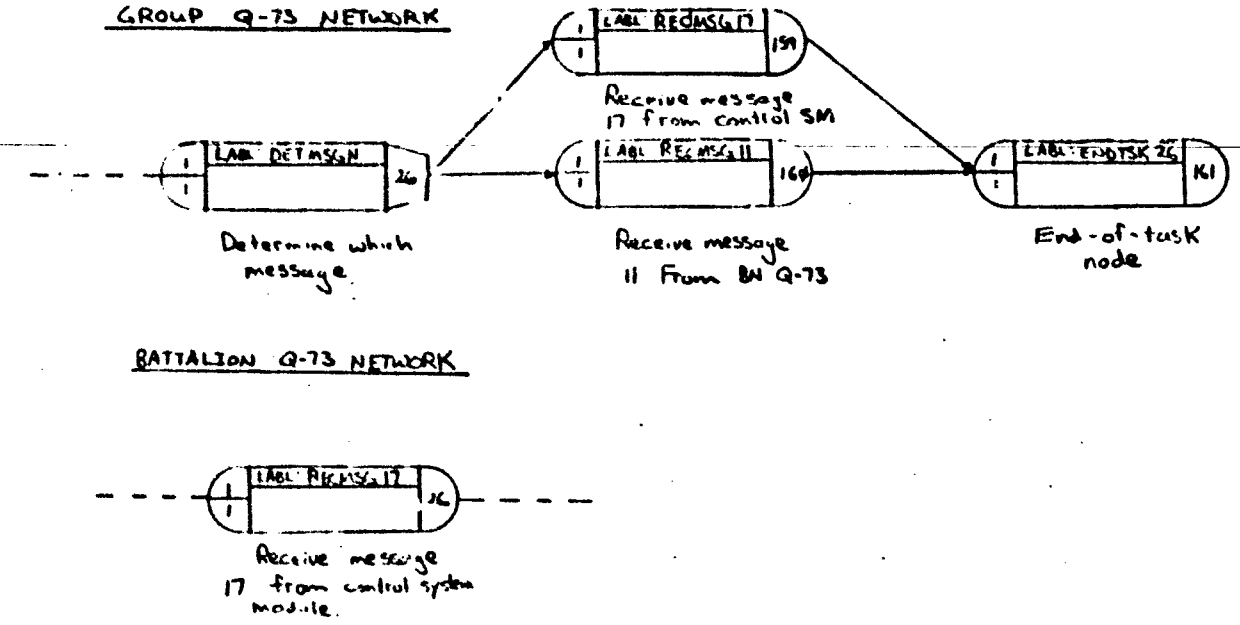
DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION MEAN	STD.DEV.	DISTRIBUTION TYPE	TASK ELEMENT ERROR FACTOR
146	ALERT93	22 (BN)	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

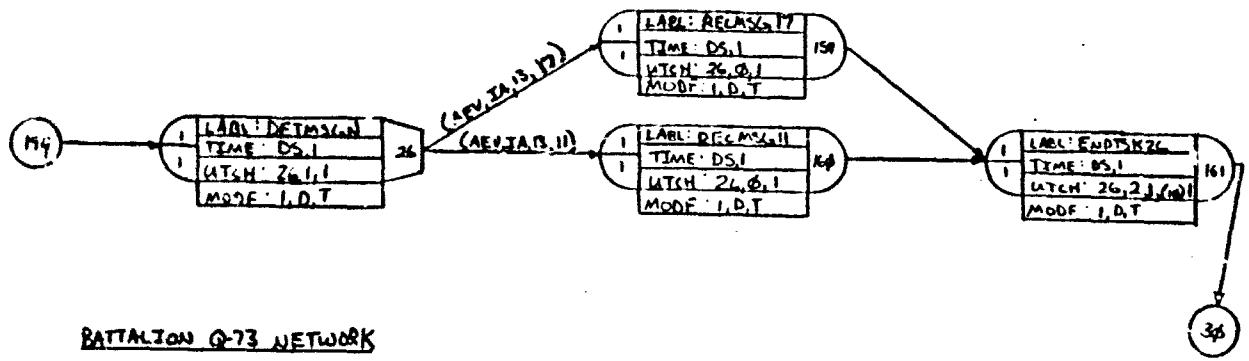


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
26) Receive Miscellaneous Messages	TD	None

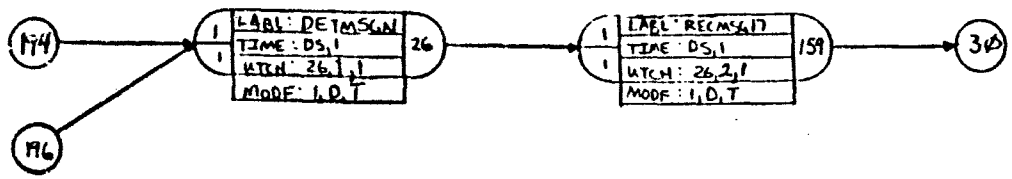
NAME: Riley Goodin DATE: 12/1/83	AIR DEFENSE SYSTEM MODULE: AN/TSQ-73 PROJECT: MOPADS

MSAINT TASK MODELS

GROUP Q-73 NETWORK



BATTALION Q-73 NETWORK



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE:	AN/TSQ-73
26) Receive Misc. Messages	TD/TDA	PROJECT:	MOPADS

MSAINT TASK NETWORKS

Zero Task Time - no skills

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
26 (BN)	RECMSG17	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 26

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
26(GRP)	DETMSGN	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 159

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
159(GRP)	RECMSG17	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 160

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
160(GRP)	RECMSG11	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 161

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
161 (CRP)	ENDTSK26	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: Q-BATTALION-AN/TSQ-73
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Table II-2.

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
AN/TSQ-73 SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
1	IDLETIME	1
2	CANC2ND	2
3	SENDTM	3
4	CLRHES	4
5	PERFHOOK	5
6		
7	CHIDFT	7
8	INTTAP	8
9		
10	SNDCOMM	10
11		
12		
13		
14		
15	ASSWEAPN	15
16		
17		
18		
19		
20	RECCMD	20
21		
22	DETALERT	22
23		
24		
25		
26	DETMSGN	26
27		
28		
29		
30	TASKSEQ	30
31	CREATETD	-
32	CREATTDA CREATTD2	
33		
34		
35		
36	CANCL2	2
37	GENSPC	3
38	ENTSAD	3
39	SWICMD	3
40	SWIACT	3
41	CMNCODE	3
42	ENDTSK3	4
43	DTHESDB	4
44	CLRHFE	4
45	CLRSTS	4
46	ENDTSK4	4
47	POSHOOK	5
48	NUMHOOK	5
49	HOOKBRCH	5
50	HKROUT1	5
51	HKROUT2	5
52	PPOSHK	5
53	HKROUT3 HOOKBRCH	5
54		
55		
56		
57		
58		
59		
60		
61		
62	ENTC	7
63	ENTID	7
64		
65		
66	DETM4	8
67	ENTMODE	8
68		
69		
70	PRINT6	8
71	PRM4CH	8
72	ENDTSK8	8
73		
74		
75	DETGENSP	10
76	ENTSFUA	10
77		
78		

Table II-2 (continued)

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
AN/TSQ-73 SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER	MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
79	ENTCCC	10	119		
80			120		
81			121		
82			122		
83			123		
84			124		
85			125		
86			126		
87			127		
88			128		
89			129		
90	DEADNODE	-	130		
91			131		
92			132		
93			133		
94			134		
95	PRAARECM	15	135		
96	HOOKADR	15	136		
97	ENTADRSS	15	137		
98	SWORCCEN	15	138		
99	PRSADLDT	15	139		
100	PRSASSN	15	140	WHFUALRT	22
101	PRSAPPR	15	141		
102	ENDTSK15	15	142	ALERT16	22
103			143	ALERT1	22
104			144	ENDTSK22	22
105			145	ALERT20	22
106			146	ALERT93	22
107			147		
108	EXMALERT	20	148		
109	PRSCLRA	20	149		
110	ENTCOMCD	20	150		
111	ENDTSK20	20	151		
112			152		
113			153		
114			154		
115			155		
116			156		
117			157		
118			158		

Table II-2 (continued)

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
AN/TSQ-73 SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER	MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
159	RECMSG17	26	199		
160	RECMSG11	26	200		
161	ENDTSDK26	26			
162					
163					
164					
165					
166					
167					
168					
169					
170					
171					
172					
173					
174					
175					
176					
178					
179					
180					
181					
182					
183					
184					
185					
186					
187					
188					
189					
190					
191	TSEQTD	30			
192	TSEQTDA	30			
193	TDROUTE1	30			
194	TDROUTE2	30			
195	TDAROUT1	30			
196	TDAROUT2	30			
197					
198					

5-0 USER FUNCTIONS

There are no user functions in the Battalion or Croup Q-73 user code.

6-0 MODERATOR FUNCTION

There is only one moderator function (referred to as Moderator Function #1) used by the Q-73 system modules. This moderator function allows the user to gain access to the human factors codes for moderating task performance time.

Any task node that has moderator function one active must have skills and distribution data stored in the data base, and the task time on the Task data card must be specified as DS,1.

7-0 MESSAGES SENT FROM Q-73 SYSTEM MODULE

The following forms describe each of the messages sent from operators in the Battalion and Group Q-73 system modules to operators within the Q-73's and to operators in the IHAWK.

MOPADS MESSAGE DESCRIPTION

Message No. 1 — MESSAGE ID ELEMENT — Page 1 of 1

Element		Description	Value
1	*	Receiver CRN	-
2		Operator Type	1 or 3
3		Functional Type	1
4		Message Subtype	1
5		Message Priority	-9

MESSAGE DATA LINK ELEMENT

Element		Description	Value
1		Communication Network 1-Voice 2-ATDL	2
2		Acknowledgement Required 1 - Yes 2 - No	1
3		Unused	-
4		ATDL Code (Unused)	-
5	*	Time Message Sent	-
6	*	Message Number	-
7	*	Sender CRN	-
8		Sender Operator Type	1
9		Sender System Module Type	2 or 3
10		Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# Subs. WORDS = 4
2	Which Fire Section(=1 F.S.A, =2 F.S.B, =3 Both)
3	NTRACK Column Number
4	NTRACK Column Number (use only if element 2 = 3)
5	Copy Row # of FS

MESSAGE SUBTYPE DESCRIPTION

Hold Fire Command. Tells receiver to stop engaging an aircraft and destroy any in-flight missiles.

Name: Riley Goodin
Date: 12/20/83
System Module: AN/TSQ-73
Project: MOPADS

From	To	From	To
GRP(3)	BN(4)		
BN(3)	HAWK(35)		

MOPADS MESSAGE DESCRIPTION

Message No. 3 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	1 or 3
3	Functional Type	1
4	Message Subtype	3
5	Message Priority	-7

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	1
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	1
9	Sender System Module Type	2 or 3
10	Task Mode Number Sent From	-

* must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# WORDS = 4
2	Which Fire Section (=1 A, =2 B, =3 Both)
3	NTRACK Column Number
4	NTRACK Column Number = 2nd F.S. (if element 2 = 3)
5	Copy Row # of FS

MESSAGE SUBTYPE DESCRIPTION

Cease Engage Command. Stop engaging, don't destroy in-flight missiles, and break the IHIPIR lock.

Name: Riley Goodin
Date: 12/20/83

System Module: AN/TSQ-73
Project: MOPADS

From	To	From	To
GRP(3)	BN(4)		
BN(3)	HAWK(37)		

MOPADS MESSAGE DESCRIPTION

Message No. 4 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	1 or 3
3	Functional Type	1
4	Message Subtype	4
5	Message Priority	+6

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	1
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	1
9	Sender System Module Type	2 or 3
10	Task Mode Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# WORDS = 2
2	Which Fire Section (=1 A, =2 B)
3	NTRACK Column Pointer

MESSAGE SUBTYPE DESCRIPTION

Engage Command on a track which was previously assigned (covered). Gives permission to fire. (Coupled with Message 26 to Q-73)

Name: Riley Goodin
Date: 12/20/83

System Module: AN/TSQ-73
Project: MOPADS

From	To	From	To
BN(10)	HAWK(32)		

MOPADS MESSAGE DESCRIPTION

Message No. 7 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	3
3	Functional Type	1
4	Message Subtype	7
5	Message Priority	+6

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	1
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	1
9	Sender System Module Type	2 or 3
10	Task Node Number Sent From	-

* Must be set at the time the message is sent *

VARIABLE MESSAGE FORMAT

Element	Description
1	# WORDS = 3
2	Which Fire Section (=Ø Either, =1 A, =2 B)
3	NTRACK Column Number
4	=1 Primary, =2 Secondary, -1 if covered

MESSAGE SUBTYPE DESCRIPTION

Assign-New Tartet - HIPIR Lock do not fire

Name: Riley Goodin
Date: 12/20/83

System Module: AN/TSQ-73
Project: MOPADS

From	To	From	To
BN(15)	P WK(32)		

MOPADS MESSAGE DESCRIPTION

Message No. 8	MESSAGE ID ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	-
2	Operator Type	3
3	Functional Type	1
4	Message Subtype	8
5	Message Priority	+5

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	1
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	1
9	Sender System Module Type	2 or 3
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# WORDS =2
2	Which Fire Section (=0 Either, =1 A, =2 B)
3	NIRACK Column Number

MESSAGE SUBTYPE DESCRIPTION

Cover New Target Command. Obtain a HIPIR lock on a new target, but do not fire.

Name: Riley Goodin	System Module: AN/TSQ-73
Date: 12/20/83	Project: MOPADS

From	To	From	To
BN(15)	HAWK(33)		

MOPADS MESSAGE DESCRIPTION

Message No. 11 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	1
3	Functional Type	4
4	Message Subtype	1
5	Message Priority	+2

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	1
9	Sender System Module Type	3
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	# WORDS = 1
2	Message Number Acknowledging

MESSAGE SUBTYPE DESCRIPTION

Acknowledge Message, type Will Comply (BN to GRP only)

Name: Riley Goodin
Date: 12/20/83

System Module: AN/TSQ-73
Project: MOPADS

From	To	From	To
BN(20)	GRP(26)		

III. USER WRITTEN PROGRAMS

1-0 OVERVIEW OF THE FLOW OF CONTROL

The user-written code for the Battalion and Group Q-73 system modules are accessed via the subroutines UTBNQ and UTGRPG, respectively. When a node is being processed, MSAINT calls subroutin UTASK, UTASK determines if it is a Battalion or Group Q-73 node. If so, UTASK calls UTBNQ or URGRPG. These subroutines then call the subroutine that performs processing for that node (e.g., T10Q, T150Q, etc.)(T160G, etc. if Group).

The Q-73 MSAINT networks regulate the flow of the entitites through the task nodes. Branching is accomplished either deterministically where the operator follows a logical sequence of actions or conditionally where the operator may do one node verses another node depending on the state of the system.

2-0 EXTERNAL FILE USAGE

There is no direct external file access in the Battalion and Group Q-73 system modules.

3-0 SUBPROGRAM DESCRIPTIONS

This section contains a copy of the subprogram descriptions for each of the Battalion and Group Q-73 user code subprograms. These subprograms are on the follcwing pages. Only four subroutines are specific to Group Q-73. They are INITG, T160G, T161G, and UTGRPG. Since the operatorsin the Group Q-73 do some of the same tasks as the operators in the Battalion Q-73 the user code associated with a node in a network for the Battalion may be used for the same node in the Group Q-73.

```
      SUBROUTINE CLTDQ (ID)
C
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.13
C
C**PURPOSE:   THIS SUBROUTINE IS USED TO COMPUTE THE SCREEN CLUTTER FOR
C             A Q-73 OPERATOR.THIS VALUE IS THEN USED TO UPDATE THE
C             OPERATOR STATE VECTOR.
C
C**INPUT PARAMETERS:   ID=OPERATOR ID
C
```

```
      SUBROUTINE GEV1Q(IDOP,NCOPI,IOPR,GS1)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      GEV1Q WILL EVALUATE THE GOAL STATE FOR GOAL 1 FOR THE
C      AN/TSQ-73 OPERATORS. THE FIRST GOAL IS SELF PRESERVATIO
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GS1-GOAL STATE FOR GOAL 1.  IF THE OPERATOR DOES NOT HAVE
C          GOAL 1, GS1 IS UNCHANGED.
```

```
      SUBROUTINE GEV2Q(IDOP,NCOPI,IOPR,GS2)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      GEV2Q WILL EVALUATE THE GOAL STATE FOR GOAL 2 FOR THE
C      AN/TSQ-73 OPERATORS. THE SECOND GOAL IS TO PROTECT CRITICAL
C      ASSETS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GS2-GOAL STATE FOR GOAL 2.  IF THE OPERATOR DOES NOT HAVE
C          GOAL 2, GS2 IS UNCHANGED.
```

```
      SUBROUTINE GEV3Q(IDOP,NCOPI,IOPR,GS3)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      GEV3Q WILL EVALUATE THE GOAL STATE FOR GOAL 3 FOR THE
C      AN/TSQ-73 OPERATORS. THE 3-RD GOAL IS REDUCE THE NUMBER OF
C      UNASSIGNED, HOSTILE TRACKS OR COVERED TRACKS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GS3-GOAL STATE FOR GOAL 3.  IF THE OPERATOR DOES NOT HAVE
C          GOAL 3, GS3 IS UNCHANGED.

      SUBROUTINE GEV4Q(IDOP,NCOPI,IOPR,GS4)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      GEV4Q WILL EVALUATE THE GOAL STATE FOR GOAL 4 FOR THE
C      AN/TSQ-73 OPERATORS. THE 4-TH GOAL IS REDUCE THE NUMBER OF
C      UNIDENTIFIED TRACKS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GS4-GOAL STATE FOR GOAL 4.  IF THE OPERATOR DOES NOT HAVE
C          GOAL 4, GS4 IS UNCHANGED.

      SUBROUTINE GEV5Q(IDOP,NCOPI,IOPR,GS5)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      GEV5Q WILL EVALUATE THE GOAL STATE FOR GOAL 5 FOR THE
C      AN/TSQ-73 OPERATORS. THE 5-TH GOAL IS REDUCE THE NUMBER OF
C      VIDEO CONTACTS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS5-GOAL STATE FOR GOAL 5.  IF THE OPERATOR DOES NOT HAVE
C           GOAL 5, GS5 IS UNCHANGED.

------------------------------------------------------------------------

      SUBROUTINE GEV60(IDOP,NCOPI,IOPR,GS6)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       GEV60 WILL EVALUATE THE GOAL STATE FOR GOAL 6 FOR THE
C       AN/TSQ-73 OPERATORS. THE 6-TH GOAL IS MAXIMIZE MISSILES.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS6-GOAL STATE FOR GOAL 6.  IF THE OPERATOR DOES NOT HAVE
C           GOAL 6, GS6 IS UNCHANGED.

------------------------------------------------------------------------

      SUBROUTINE GEV70(IDOP,NCOPI,IOPR,GS7)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       GEV70 WILL EVALUATE THE GOAL STATE FOR GOAL 7 FOR THE
C       AN/TSQ-73 OPERATORS. THE 7-TH GOAL IS RECEIVE MESSAGES..
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS7-GOAL STATE FOR GOAL 7.  IF THE OPERATOR DOES NOT HAVE
C           GOAL 7, GS7 IS UNCHANGED.

------------------------------------------------------------------------

      SUBROUTINE GEV80(IDOP,NCOPI,IOPR,GS8)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       GEV80 WILL EVALUATE THE GOAL STATE FOR GOAL 8 FOR THE
C       AN/TSQ-73 OPERATORS. THE 8-TH GOAL IS DO NOT SHOOT AT FRIENDS.
```

```
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR'S UNIT
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS8-GOAL STATE FOR GOAL 8.  IF THE OPERATOR DOES NOT HAVE
C          GOAL 8, GS8 IS UNCHANGED.

------------------------------------------------------------------------

      SUBROUTINE GEVALQ(IDOP,NADSM,NCOPI,NGS,IOPR,GS,NNG)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       GEVALQ WILL EVALUATE AN OPERATOR'S GOAL STATE VECTOR
C       FOR THE AN/TSQ-73 BATTALION.
C
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NADSM-SYSTEM MODULE TYPE
C       NCOPI-COPY ROW NUMBER
C       NGS-ACTUAL LENGTH OF GS
C--INPUT/OUTPUT PARAMETERS:
C       IOPT(2)-DBAA OF THE OPERATOR STATE VECTOR
C--OUTPUT PARAMETERS:
C       GS(NGS)-GOAL STATE VECTOR
C       NNG-THE NUMBER OF GOALS THE OPERATORS OF THE SYSTEM HAVE.
C            (I.E. ONLY THE FIRST NNG ELEMENTS OF GS HAVE MEANINGFUL
C             INFORMATION). IF THE OPERATOR DOES NOT HAVE GOAL I,
C             THEN GS(I)=-1.E10.

------------------------------------------------------------------------

      FUNCTION IFFQ(ID,NTRK)
C
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C
C**PURPOSE:   THIS SUBROUTINE IS USED TO PERFORM IFF OPERATIONS ON A
C             TRACK FOR THE Q-73.
C
C**INPUT PARAMETERS:   ID=OPERATOR ID
C                      NTRK=NTRACK COLUMN NUMBER
C
C**OUTPUT PARAMETERS:  IFFQ=1 IF HOSTILE
C                          =2 IF FRIEND
C                          =3 IF UNKNOWN
C
```

```
      FUNCTION IFMODQ (IOPP)
C
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C
C**PURPOSE:  THIS SUBROUTINE IS USED TO DETERMINE WHICH IFF MODE SHOULD
C            BE SELECTED BY THE TDA IN THE Q-73.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C
C**OUTPUT PARAMETERS:  IFMODQ=1 CHANGE TO MODE 1,2,OR 3
C                             =2 REMAIN IN CURRENT MODE
C                             =3 SELECT MODE 4
C
```

```
      SUBROUTINE INITQ(NRUN)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      INITQ PERFORMS INITIALIZATION FOR THE BATTALION AN/TSQ-73
C      SYSTEM MODULE
C--INPUT PARAMETERS:
C      NRUN-RUN NUMBER
C           0-DO ONE-TIME INITIALIZATION BEFORE ANY RUNS
C           N-INITIALIZE FOR RUN N.
```

```
      SUBROUTINE OEVALQ(IDOF,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ,*)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      OEVALQ WILL COMPUTE THE EXPECTED GOAL STATE VECTOR FOR
C      THE AN/TSQ-73 OPERATORS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OEVALQ WILL
C           EVALUATE EXPECTED GOAL STATES GIVEN THAT THE
C           OPERATOR PERFORMS TASK JTASK.
```

```
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM PERFORMING
C               TASK JTASK. ON INPUT, GSJ=GS.
C--OUTPUT PARAMETERS:
C      TJ-EXPECTED TIME (MINUTES) TO PERFORM TASK JTASK.  IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C         THEN TJ=-1.
C--ALTERNATE RETURNS:
C      1-JTASK IS GREATER THAN THE MAXIMUM TASK NUMBER THAT THE OPERATOR
C        PERFORMS.  OEVALQ WILL BE CALLED REPEATEDLY WITH GREATER
C        VALUES OF JTASK UNTIL THIS CONDITION OCCURS.

------------------------------------------------------------------------

      SUBROUTINE OTS15Q(IDOP,NADSM,NCOPI,GS,NNG,IOPR,GSJ,TJ)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C      OTS15Q PERFORMS EVALUATION OF OPERATOR TASK 15 FOR THE
C      BATTALION Q-73 OPERATORS. OTS15Q IS CALLED BY OEVALQ ONLY.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROU NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM PERFORMING
C               TASK 15. ON INPUT, GSJ=GS.
C--OUTPUT PARAMETERS:
C      TJ-EXPECTED TIME (MINUTES) TO PERFORM TASK 15.  IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK 15,
C         THEN TJ=-1.

------------------------------------------------------------------------

      SUBROUTINE Q73EQ(KODE)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--INPUT PARAMETERS:
C      Q73EQ PERFORMS SPECIAL ERROR PROCESSING FOR MOPADS ERROR
C      CODES 5000-5999 WHICH ARE GENERATED IN THE Q-73 SYSTEM
C      MODULE CODE
C--INPUT PARAMETERS:
C      KODE-ERROR CODE VALUE
```

```
      SUBROUTINE T---Q (NPLACE)
C
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C
C**PURPOSE: USER CODE FOR BN Q-73 TASK NODE ---
C
C**INPUT PARAMETER: NPLACE - TASK OCCURRENCE TIME
C

 --------  THIS PROGRAM IS TYPICAL OF ALL  -----------
                  NODE SUBROUTINES
```

```
      FUNCTION USERFQ(ICODE)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       USERFQ  EVALUATES  THE USER FUNCTION FOR THE GROUP AND
C       BATTALION AN/TSQ-73
C--INPUT PARAMETERS:
C      ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C      USERFQ-VALUE OF THE USER FUNCTION
```

```
      FUNCTION USERNQ(ICODE)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       USERNQ EVALUATES THE INPUT USER FUNCTION FOR THE GROUP AND
C       BATTALION AN/TSQ-73
C--INPUT PARAMETERS:
C      ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C      USERNQ-VALUE OF THE USER FUNCTION
```

```
      SUBROUTINE UTBNQ(NT,NPLACE)
C--MODULE: MOPADS BATTALION AN/TSQ-73 SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.15
C--PURPOSE:
C       UTBNQ PROCESSES CALLS FROM UTASK FOR THE BATTALION AN/TSQ-73
C       SYSTEM MODULE.
C--INPUT PARAMETERS:
C       NT-TASK NODE NUMBER
C       NPLACE-TASK NODE OCCURRANCE TIME(SEE UTASK)
```

```
      SUBROUTINE INITG(NRUN)
C--PURPOSE:
C       INITG PERFORMS INITIALIZATION FOR THE GROUP AN/TSQ-73
C       SYSTEM MODULE
C--INPUT PARAMETERS:
C       NRUN-RUN NUMBER
C            0-DO ONE-TIME INITIALIZATION BEFORE ANY RUNS
C            N-INITIALIZE FOR RUN N.
```

```
      SUBROUTINE T160G (NPLACE)
C
C
C**PURPOSE: USER CODE FOR GROUP Q-73 TASK NODE 160
C
C**INPUT PARAMETER: NPLACE - TASK OCCURRENCE TIME
C
```

```
      SUBROUTINE T161G (NPLACE)
C
C
C**PURPOSE: USER CODE FOR GROUP Q-73 TASK NODE 161
C
C**INPUT PARAMETER: NPLACE - TASK OCCURRENCE TIME
C
```

```
      SUBROUTINE UTGRPG(NT,NPLACE)
C--PURPOSE:
C       UTGRPG PROCESSES CALLS FROM UTASK FOR THE GROUP AN/TSQ-73
C       SYSTEM MODULE.
C--INPUT PARAMETERS:
C       NT-TASK NODE NUMBER
C       NPLACE-TASK NODE OCCURRANCE TIME(SEE UTASK)
```

4-0 ERROR PROCESSING

When an error involving the Q-73 user code is detected a call is made to subroutine SERR. The only parameter this subroutine has is the error code. The error codes 5000 to 5999 are reserved for errors detected in either the Group or Battalion Q-73. The error codes 5500 to 5749 are reserved for errors solely in the Group Q-73. The error codes 5750 to 5999 are reseved for errors solely in the Battalion Q-73. Table III-1 lists all the error codes and their definitions.

Table III-1. Battalion and Group Q-73 Error Messages.

Error Code	Description	Programs
5020	Invalid message id for message to be received by Q-73.	T26Q
5021	Unable to find Q-73 operator	T159Q
5022	Unable to clear Q-73 operator	T159Q
5023	Unable to self-clear	T159Q
5200	Increase NEVEL	GEV8A
5201	Incorrect operator type	OEVALQ
5202	Data Base error	INITQ
5203	Can't find covered Track in Track Data	OEVALQ
5204	Increase dimension of NFASS array	OTS15Q
5205	Incorrect user function code	USERFQ, USERNQ
5206	Incorrect node number	UTBNQ

IV. MSAINT NETWORK DATA

1-0 BN Q-73 LISTING

The following is a listing of the MSAINT network data for the Battalion Q-73.

```
GEN,BATT Q73,9,7,83,3*
POP,0,0,15,10,10*
BIS,1,CO,0.0*
BIS,2,NO,.1,.02,.2,.07*
BIS,3,CO,0.001*
UTI,1,NUMINCOV*
IND,1,A*
*
*    DEAD NODE FOR ROUTING ENTITIES TO WHEN THE Q73 HAS BEEN DESTROYED
*    BY ENEMY AIRCRAFT.
*
TAS,90,DEADNODE,1,1,DS,1*
UTC,90,0.0,0.0,1.0*
MOD,90,1,D,T*
* NO BRANCH,ENTITY IS ELIMINATED FROM NETWORK
*
*     SOURCE TASKS
*
TAS,31,CREATETD,,,DS,1,,,SO*
UTC,31,0.0,0.0,1.0*
MOD,31,1,D,T*
DET,31,30*
*
TAS,32,CREATTDA,,,DS,1,,,SO*
UTC,32,0.0,0.0,1.0*
MOD,32,1,D,T*
DET,32,30*
*
*    OPERATOR TASK 1, SCAN THE SCREEN,DISPLAYS,ETC. (TD/TDA)
*
TAS,1,IDLETIME,1,1,DS,1*
UTC,1,1.0,3.0,1.0*
DET,1,30*
*
*    OPERATOR TASK 2, CANCEL SECONDARY ASSIGNMENT (TD)
*
TAS,2,CANC2ND,1,1,DS,1*
UTC,2,2.0,1.0,1.0*
DET,2,5*
*
TAS,36,CANCL2,1,1,DS,1*
UTC,36,2.0,2.0,1.0,(12)1.0*
DET,36,30*
*
*    OPERATOR TASK 3, SEND TERMINATE COMMANDS (TD)
*
```

```
TAS,3,SENDTM,1,1,DS,1*
UTC,3,3.0,1.0,1.0*
DET,3,5*
*
TAS,37,GENSPC,1,1,DS,1*
UTC,37,3.0,0.0,1.0*
CFI,37,38,AEV,2.0,13,IA,,5,AEV,1.0,13,IA*
*
TAS,38,ENTSAD,1,1,DS,1*
UTC,38,3.0,0.0,1.0*
DET,38,39*
*
TAS,39,SWICMD,1,1,DS,1*
UTC,39,3.0,0.0,1.0*
CFI,39,40,AEV,1.0,13,IA,,41,AEV,2.0,13,IA*
*
TAS,40,SWIACT,1,1,DS,1*
UTC,40,3.0,0.0,1.0*
DET,40,42*
*
TAS,41,CMNCODE,1,1,DS,1*
UTC,41,3.0,0.0,1.0*
DET,41,42*
*
TAS,42,ENDTSK3,1,1,DS,3*
UTC,42,3.0,2.0,1.0,(12)1.0*
MOD,42,1,D,T*
DET,42,30*
*
*    OPERATOR TASK 4, CLEAR HOLD FIRE,EFFECTIVE,STATUS (TB)
*
TAS,4,CLRHES,1,1,DS,1*
UTC,4,4.0,1.0,1.0*
DET,4,43*
*
TAS,43,BTHESDB,1,1,DS,1*
UTC,43,4.0,0.0,1.0*
CFI,43,5,AEV,1.0,13,IA,,
       5,AEV,2.0,13,IA,,
       46,AEV,3.0,13,IA*
*
TAS,44,CLRHFE,1,1,DS,1*
UTC,44,4.0,0.0,1.0*
DET,44,46*
*
TAS,45,CLRSTS,1,1,DS,1*
```

```
UTC,45,4.0,0.0,1.0*
DET,45,46*
*
TAS,46,ENDTSK4,1,1,DS,3*
UTC,46,4.0,2.0,1.0,(12)1.0*
MOD,46,1,D,T*
DET,46,30*
*
*    OPERATOR TASK 5, PERFORM HOOKING PROCEDURE (TD/TDA)
*
TAS,5,PERFHOOK,1,1,DS,1*
UTC,5,5.0,1.0,1.0*
CFI,5,48,ALV,0.0,9,IA,,47,AGV,0.0,9,IA*
*
TAS,47,POSHOOK,1,1,DS,1*
UTC,47,5.0,0.0,1.0*
DET,47,52*
*
TAS,52,PPOSHK,1,1,DS,1*
UTC,52,5.0,0.0,1.0*
DET,52,49*
*
TAS,48,NUMHOOK,1,1,DS,1*
UTC,48,5.0,0.0,1.0*
DET,48,49*
*
TAS,49,HOOKBRCH,1,1,DS,1*
UTC,49,5.0,0.0,1.0*
MOD,49,1,D,T*
CFI,49,50,ALV,46.0,8,IA,,
       51,ALV,97.0,8,IA,,
       53,ALV,142.0,8,IA*
*
TAS,50,HKROUT1,1,1,DS,1*
UTC,50,5.0,2.0,1.0,(12)1.0*
MOD,50,1,D,T*
CFI,50,36,AEV,36.0,8,IA,,
       37,AEV,37.0,8,IA,,
       39,AEV,39.0,8,IA,,
       44,AEV,44.0,8,IA,,
       45,AEV,45.0,8,IA*
*
TAS,51,HKROUT2,1,1,DS,1*
UTC,51,5.0,2.0,1.0,(12)1.0*
MOD,51,1,D,T*
CFI,51,70,AEV,70.0,8,IA,,
```

```
       75,AEV,75.0,8,IA,,
       79,AEV,79.0,8,IA,,
       96,AEV,96.0,8,IA*
*
TAS,53,HKROUT3,1,1,DS,1*
UTC,53,5.0,2.0,1.0,(12)1.0*
MOD,53,1,D,T*
CFI,53, 98,AEV, 98.0,8,IA,,
       108,AEV,108.0,8,IA,,
       140,AEV,140.0,8,IA*
*
*    OPERATOR TASK 7, ENTER ID DATA (IDA)
*
TAS,7,CHIDFT,1,1,DS,1*
UTC,7,7.0,1.0,1.0*
CFI,7,63,AEV,1.0,13,IA,,62,AEV,2.0,13,IA*
*
TAS,62,ENTC,1,1,DS,1*
UTC,62,7.0,0.0,1.0*
DET,62,63*
*
TAS,63,ENTID,1,1,DS,1*
UTC,63,7.0,3.0,1.0,(12)1.0*
DET,63,30*
*
*    OPERATOR TASK 8, INTERROGATE TARGET (IDA)
*
TAS,8,INTTAR,1,1,DS,1*
UTC,8,8.0,1.0,1.0*
DET,8,66*
*
TAS,66,DETM4,1,1,DS,1*
UTC,66,8.0,0.0,1.0*
CFI,66,67,AEV,1.0,13,IA,,
        5,AEV,2.0,13,IA,,
       71,AEV,3.0,13,IA*
*
TAS,67,ENTMODE,1,1,DS,1*
UTC,67,8.0,0.0,1.0*
DET,67,5*
*
TAS,70,PRINTG,1,1,DS,1*
UTC,70,8.0,0.0,1.0*
DET,70,72*
*
TAS,71,PRM4CH,1,1,DS,1*
```

```
UTC,71,8.0,0.0,1.0*
DET,71,72*
*
TAS,72,ENDTSKB,1,1,DS,3*
UTC,72,8.0,2.0,1.0,(12)1.0*
MOD,72,1,D,T*
DET,72,30*
*
*    OPERATOR TASK 10, SEND COMMAND MESSAGE (TD)
*
TAS,10,SNDCOMM,1,1,DS,1*
UTC,10,10.0,1.0,1.0*
DET,10,5*
*
TAS,75,DETGENSP,1,1,DS,1*
UTC,75,10.0,1.0,1.0*
CFI,75,5,AEV,1.0,13,IA,,
      76,AEV,2.0,13,IA,,
      79,AEV,3.0,13,IA*
*
TAS,76,ENTSFUA,1,1,DS,1*
UTC,76,10.0,0.0,1.0*
DET,76,79*
*
TAS,79,ENTCCC,1,1,DS,1*
UTC,79,10.0,2.0,1.0,(12)1.0*
DET,79,30*
*
*    OPERATOR TASK 15, ASSIGN WEAPONS
*
TAS,15,ASSWEAPN,1,1,DS,1*
UTC,15,15.0,1.0,1.0*
CFI,15,5,AEV,1.0,13,IA,,95,AEV,2.0,13,IA*
*
TAS,95,PRAARECM,1,1,DS,1*
UTC,95,15.0,0.0,1.0*
DET,95,102*
*
TAS,96,HOOKADR,1,1,DS,1*
UTC,96,15.0,0.0,1.0*
CFI,96,5,AEV,1.0,13,IA,,97,AEV,2.0,13,IA*
*
TAS,97,ENTADRSS,1,1,DS,1*
UTC,97,15.0,0.0,1.0*
DET,97,98*
*
```

```
TAS,98,SWORCCEN,1,1,DS,1*
UTC,98,15.0,0.0,1.0*
CFI,98,100,AEV,1.0,13,IA,,99,AEV,2.0,13,IA*
*
TAS,99,PRSADLDT,1,1,DS,1*
UTC,99,15.0,0.0,1.0*
DET,99,102*
*
TAS,100,PRSASSN,1,1,DS,1*
UTC,100,15.0,0.0,1.0*
DET,100,101*
*
TAS,101,PRSAPPR,1,1,DS,1*
UTC,101,15.0,0.0,1.0*
DET,101,102*
*
TAS,102,ENDTSK15,1,1,DS,3*
UTC,102,15.0,2.0,1.0,(12)1.0*
MOD,102,1,D,T*
DET,102,30*
*
*    OPERATOR TASK 20, RECEIVE COMMANDS (TD)
*
TAS,20,RECCMD,1,1,DS,1*
UTC,20,20.0,1.0,1.0*
DET,20,5*
*
TAS,108,EXMALERT,1,1,DS,1*
UTC,108,20.0,0.0,1.0*
CFI,108,109,AEV,1.0,13,IA,,110,AEV,2.0,13,IA*
*
TAS,109,PRSCLRA,1,1,DS,1*
UTC,109,20.0,0.0,1.0*
DET,109,111*
*
TAS,110,ENTCOMCD,1,1,DS,1*
UTC,110,20.0,0.0,1.0*
DET,110,111*
*
TAS,111,ENDTSK20,1,1,DS,3*
UTC,111,20.0,2.0,1.0,(12)1.0*
MOD,111,1,D,T*
DET,111,30*
*
*    OPERATOR TASK 22, CLEAR ALERTS
*
```

```
TAS,22,DETALERT,1,1,DS,1*
UTC,22,22.0,1.0,1.0*
CFI,22,143,AEV,1.0,13,IA,,140,AGV,16.0,13,IA*
*
TAS,143,ALERT1,1,1,DS,1*
UTC,143,22.0,0.0,1.0*
DET,143,144*
*
TAS,140,UNFUALRT,1,1,DS,1*
UTC,140,22.0,0.0,1.0*
CFI,140,142,AEV,16.0,13,IA,,
        145,AEV,20.0,13,IA,,
        146,AEV,93.0,13,IA*
*
TAS,142,ALERT16,1,1,DS,1*
UTC,142,22.0,0.0,1.0*
DET,142,144*
*
TAS,145,ALERT20,1,1,DS,1*
UTC,145,22.0,0.0,1.0*
DET,145,144*
*
TAS,146,ALERT93,1,1,DS,1*
UTC,146,22.0,0.0,1.0*
DET,146,144*
*
TAS,144,ENDTSK22,1,1,DS,3*
UTC,144,22.0,2.0,1.0,(12)1.0*
MOD,144,1,D,T*
DET,144,30*
*
*    OPERATOR TASK 26 TO RECEIVE MISCELLANEOUS MESSAGES
*
TAS,26,DETMSGN,1,1,DS,1*
UTC,26,26.0,1.0,1.0*
MOD,26,1,D,T*
DET,26,159*
*
TAS,159,RECMSG17,1,1,DS,1*
UTC,159,26.0,0.0,1.0*
MOD,159,1,D,T*
DET,159,30*
*
*   OPERATOR TASK 30, TASK SEQUENCING
*
TAS,30,TASKSEQ,1,1,DS,1*
```

```
UTC,30,30.0,1.0,1.0*
MOD,30,1,D,T*
CFI,30,191,AEV,1.0,3,IA,,192,AEV,2.0,3,IA*
*
TAS,191,TSEQTD,1,1,DS,1*
UTC,191,30.0,0.0,1.0*
MOD,191,1,D,T*
CFI,191,193,ALV,6.0,14,IA,,194,AGV,5.0,14,IA*
*
TAS,193,TDROUTE1,1,1,DS,1*
UTC,193,30.0,2.0,1.0,(12)1.0*
MOD,193,1,D,T*
CFI,193,1,AEV,1.0,14,IA,,
        2,AEV,2.0,14,IA,,
        3,AEV,3.0,14,IA,,
        4,AEV,4.0,14,IA,,
        5,AEV,5.0,14,IA*
*
TAS,194,TDROUTE2,1,1,DS,1*
UTC,194,30.0,2.0,1.0,(12)1.0*
MOD,194,1,D,T*
CFI,194,10,AEV,10.0,14,IA,,
        15,AEV,15.0,14,IA,,
        20,AEV,20.0,14,IA,,
        22,AEV,22.0,14,IA,,
        26,AEV,26.0,14,IA*
*
TAS,192,TSEQTDA,1,1,DS,1*
UTC,192,30.0,0.0,1.0*
MOD,192,1,D,T*
CFI,192,195,ALV,8.0,14,IA,,196,AGV,7.0,14,IA*
*
TAS,195,TDAROUT1,1,1,DS,1*
UTC,195,30.0,2.0,1.0,(12)1.0*
MOD,195,1,D,T*
CFI,195,1,AEV,1.0,14,IA,,
        5,AEV,5.0,14,IA,,
        7,AEV,7.0,14,IA*
*
TAS,196,TDAROUT2,1,1,DS,1*
UTC,196,30.0,2.0,1.0,(12)1.0*
MOD,196,1,D,T*
CFI,196, 8,AEV, 8.0,14,IA,,
        22,AEV,22.0,14,IA,,
        26,AEV,26.0,14,IA*
```

2-0 GROUP Q-73 LISTING

The following is a listing of the MSAINT network data for the Group Q-73.

```
GEN,GROUP Q73,9,7,83,3*
POP,0,0,15,10,10*
DIS,1,CO,0.0*
DIS,2,NO,.1,.02,.2,.07*
DIS,3,CO,0.001*
IMU,1,A*
*
*    DEAD NODE FOR ROUTING ENTITIES TO WHEN THE Q73 HAS BEEN DESTROYED
*    BY ENEMY AIRCRAFT.
*
TAS,90,DEADNODE,1,1,DS,1*
UTC,90,0.0,0.0,1.0*
MOD,90,1,D,T*
* NO BRANCH,ENTITY IS ELIMINATED FROM NETWORK
*
*     SOURCE TASKS
*
TAS,31,CREATETD,,,DS,1,,,SO*
UTC,31,0.0,0.0,1.0*
MOD,31,1,D,T*
DET,31,30*
*
TAS,32,CREATTD2,,,DS,1,,,SO*
UTC,32,0.0,0.0,1.0*
MOD,32,1,D,T*
DET,32,30*
*
*    OPERATOR TASK 1,SCAN THE SCREEN,DISPLAYS,ETC.
*
UTC,1,1.0,3.0,1.0*
DET,1,30*
*
*    OPERATOR TASK 3, SEND TERMINATE COMMANDS (TD)
*
TAS,3,SENDTM,1,1,DS,1*
UTC,3,3.0,1.0,1.0*
DET,3,5*
*
TAS,37,GENSPC,1,1,DS,1*
UTC,37,3.0,0.0,1.0*
CFI,37,38,AEV,2.0,13,IA,,5,AEV,1.0,13,IA*
*
TAS,38,ENTSAD,1,1,DS,1*
UTC,38,3.0,0.0,1.0*
DET,38,39*
*
```

```
TAS,39,SWICMD,1,1,DS,1*
UTC,39,3.0,0.0,1.0*
CFI,39,40,AEV,1.0,13,IA,,41,AEV,2.0,13,IA*
*
TAS,40,SWIACT,1,1,DS,1*
UTC,40,3.0,0.0,1.0*
DET,40,42*
*
TAS,41,CMNCODE,1,1,DS,1*
UTC,41,3.0,0.0,1.0*
DET,41,42*
*
TAS,42,ENDTSK3,1,1,DS,3*
UTC,42,3.0,2.0,1.0,(12)1.0*
MOD,42,1,D,T*
DET,42,30*
*
*    OPERATOR TASK 5, PERFORM HOOKING PROCEDURE
*
TAS,5,PERFHOOK,1,1,DS,1*
UTC,5,5.0,1.0,1.0*
CFI,5,48,ALV,0.0,9,IA,,47,AGV,0.0,9,IA*
*
TAS,47,POSHOOK,1,1,DS,1*
UTC,47,5.0,0.0,1.0*
DET,47,52*
*
TAS,52,PPOSHK,1,1,DS,1*
UTC,52,5.0,0.0,1.0*
DET,52,53*
*
TAS,48,NUMHOOK,1,1,DS,1*
UTC,48,5.0,0.0,1.0*
DET,48,53*
*
TAS,53,HOOKBRCH,1,1,DS,1*
UTC,53,5.0,2.0,1.0,(12)1.0*
MOD,53,1,D,T*
CFI,49,37,AEV,37.0,8,IA,,
       39,AEV,39.0,8,IA*
*
*    OPERATOR TASK 26 TO RECEIVE MISCELLANEOUS MESSAGES
*
TAS,26,DETMSGN,1,1,DS,1*
UTC,26,26.0,1.0,1.0*
MOD,26,1,D,T*
```

```
CFI,26,159,AEV,17.0,13,IA,,160,AEV,11.0,13,IA*
*
TAS,159,RECMSG17,1,1,DS,1*
UTC,159,26.0,0.0,1.0*
MOD,159,1,D,T*
DET,159,161*
*
TAS,160,RECMSG11,1,1,DS,1*
UTC,160,26.0,0.0,1.0*
MOD,160,1,D,T*
DET,160,161*
*
TAS,161,ENDTS26,1,1,DS,1*
UTC,161,26.0,2.0,1.0*
MOD,161,1,D,T*
DET,161,30*
*
*   OPERATOR TASK 30, TASK SEQUENCING
*
TAS,30,TASKSEQ,1,1,DS,1*
UTC,30,30.0,1.0,1.0*
MOD,30,1,D,T*
DET,30,191*
*
TAS,191,TSEQTD,1,1,DS,1*
UTC,191,30.0,0.0,1.0*
MOD,191,1,D,T*
CFI,191,193,ALV,6.0,14,IA,,194,AGV,5.0,14,IA*
*
TAS,193,TDROUTE1,1,1,DS,1*
UTC,193,30.0,2.0,1.0,(12)1.0*
MOD,193,1,D,T*
CFI,193,1,AEV,1.0,14,IA,,
        3,AEV,3.0,14,IA,,
        5,AEV,5.0,14,IA*
*
TAS,194,TDROUTE2,1,1,DS,1*
UTC,194,30.0,2.0,1.0,(12)1.0*
MOD,194,1,D,T*
DET,194,26*
$
```

V. REFERENCES

Goodin II, J. R. & Polito, J. User guide for the AN/TSQ-73 system module (MOPADS Vol. 3.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

VI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1E-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VII. CHANGE NOTICES

APPENDIX X

MOPADS FINAL REPORT:

DOCUMENTATION MANUAL FOR THE IHAWK SYSTEM MODULE

TABLE OF CONTENTS

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are HAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology

ADP	Automatic Data Processor
ASO	Azimuth Speed Operator
ATDL	Automatic Tactical Data Link
BCC	Battery Control Central
ECM	Electronic Countermeasure
FCO	Firing Console Operator
PCP	Platoon Command Post

TCA	Tactical Control Assistant
TCC	Tactical Control Console
TCO	Tactical Control Officer

I. OVERVIEW OF THE MOPAIS MODEL

This report documents the implementation details of the IHAWK system module. It's intended audience is the MOPADS modeler who must maintain and explain the model. A companion report (Goodin & Polito (1983)) provides user information for the MOPADS user.

The MOPADS operator tasks are modeled with subnetworks of MSAINT task nodes. Each node has a subroutine of user code associated with it where the user may write code to access the data base and modify the MSAINT data structure. These task node subroutines are named according to the task node they represent. For instance, the subroutine used to process logic associated with node 175 is named T175W and the one for node 62 is named T62W.

The five operators perform various task nodes in the network. The two Firing Console Operators (FCOs) are the only operators who may perform the same task nodes; other operators perform a unique set of tasks. The operators communicate with one another by sending messages, signaling (clearing one operator to another node), altering the data base, or altering the MSAINT data structure.

Table I-1 is a list of some of the assumptions made in constructing the MOPADS model of the IHAWK.

Teble I-1. Assumptions.

1)	Jamming and ECM are not considered
2)	All setup, checkout and emplacement procedures have been performed.
3)	The IHAWK will not be mobile during the simulation.
4)	The automatic firing mode will not be employed.
5)	All ADP's remain operational assuming the IHAWK has not been destroyed.
6)	The use of the IHAWK in the PCP (Platoon Command Post) is not represented.
7)	The ATDL data link is used for communicating with Battalion AN/TSQ-73.
8)	The TCO receives all messages from Battalion.
9)	No communication occurs between IHAWK units except for those passed up through the Battalion.
10)	The IHIPIR radars will always acquire lock.
11)	A target will be positively identified if the TCA performs IFF procedures on that target.
12)	All targets fired upon will be destroyed unless the CHANGE TARGETS task is performed by the FCO.
13)	The TCO must ask permission to fire from Battalion before engaging a target other than a self-initiated engagement.
14)	The FCO must manually select launchers.
15)	The CEASE FIRE command will not be modeled.
16)	TCO will make all weapon assignments.

II. MODEL DESCRIPTION FORMS

1-0 ENTITIES

The operators that perform the work in the Battery Control Center (BCC) are the entities in the IHAWK system module. The forms in this section describe each entity, their attributes, and their resource requirements.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
45	TCO	1	Operator ID	-	None
		2	Copy Row Number	-	
		3	Operator Type	3.0	
		4	Current Track Column Pointer	0.0	
		5	Internal Mark Time	0.0	
		6	Self-Clearing Indicator	0.0	
		7	Last Task Node Branched From	0.0	
		8	Track Status (1=Manual, 2=ADP, Ø=Neither)	0.0	
		9	Current Fire Section (Ø=Neither, 1=A, 2=B, 3=Both)	0.0	
		10	Not used	0.0	
		11	Not used		
		12	Not used	0.0	
		13, 14, 15	Branching/Task Node Specific		

Name: Riley Goodin
Date: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT ENTITIES

Page 1 of 1

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
46	Tactical Control Assistant	1	Operator ID	-	-
		2	Copy Row Number	-	
		3	Operator Type	4	
		4	Current Track Column Pointer	0	
		5	Internal Mark Time	0	
		6	Self-Clearing Indicator	0	
		7	Last Task Node Branched From	0	
		8	Unused	0	
		9	IFF Mode (1=Manual, 2=Auto) Not currently used (All IFF is manual)	1.0	
		10	Unused	0.0	
		11	Unused	0.0	
		12	Unused	0.0	
		13, 14, 15	Branching	0.0	

Name: Riley Goodin
Date: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT ENTITIES

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
47	ASO (Azimuth Speed Operator)	1	Operator ID	-	-
		2	Copy Row Number	-	
		3	Operator Type	3.0	
		4	Current Track Column Pointer	0.0	
		5	Internal Mark Time	0.0	
		6	Self Clearing Indicator	0.0	
		7	Last Task Node Branched From	0.0	
		8	TCC Alert Outstanding (Ø=No, 1=Yes)	0.0	
		9	TCC Alerted Track Column Pointer	0.0	
		10	TCC Alert Track Priority (Threat)	0.0	
		11	Preempting Lower Pricrity Target (Ø=No, 1=Yes)	0.0	
		12	If IA(11)=1 this is THREAT of preempting target, Ø otherwise	0.0	
		13	Task specific	0.0	
		14	Task specific	0.0	
		15	Task specific	0.0	

Name: Riley Goodin
Date: 10/28/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT ENTITIES

Page 1 of 1

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
48	Firing Console Operator A	1	Operator ID	-	Resources Required: ILCHR1A ILCHR2A ILCHR3A
		2	Copy Row Number	-	
		3	Operator Type	6.0	
		4	Current Track Column Pointer	0.0	
		5	Internal Mark Time	0.0	
		6	Self Clearing Indicator	0.0	
		7	Last Task Node Branched From	0.0	
		8	Track Status (Ø=Neither, 1=ADP, 2=Manual)	0.0	
		9	Current Fire Method (1=Hold, or Inactive, 2=Shoot-Look-Shoot, 3=Ripple Fire)	1.0	
		10	Select Launcher Resource Number	0.0	
		11	Branch to Select Launcher From Fire Missiles Task	0.0	
		12	Not Used	0.0	

Name: Riley Goodin
Date: 10/20/83

AIR DEFENSE SYSTEM MODULE: HAWK
PROJECT: MOPADS

SAINT ENTITIES

Page 1 of 2

This page intentionally left blank.

NODE(S) WHERE CREATED	DESCRIPTION	INFORMATION ATTRIBUTES			RESOURCE REQUIREMENTS
		ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE (IF APPROPRIATE)	
		13	Task specific	0.0	
		14	Task specific	0.0	
		15	Task specific	0.0	
49	Fire Control Operator B	Same as FCO-A above except	-	-	Resources Required: ILCHR1B ILCHR2B
		3	Operator Type	7.0	ILCHR3B

Name: Riley Goodin
Date: 10/20/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT ENTITIES

2-0 RESOURCES

The launchers are modeled as resources in the IHAWK system module. There are six launchers for each IHAWK (three for each fire section). The following forms describe each of the resources and their attributes used during the simulation.

RESOURCE NUMBER	RESOURCE LABEL	DESCRIPTION OF USE	RESOURCE ATTRIBUTES			TASK NUMBERS REQUIRING THE RESOURCE
			ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE	
1	ILCHR1A	Launcher #1 for Fire Section A	1	# Hot Missiles	3	
			2	# Cold Missiles To Ready To Load on Launcher	3	
			3	Operator ID To Clear After Reloading	0	
			4	Waiting To Reload 0 = No 1 = Yes	0	
			5	Time Launcher Went Out of Action For Remainder of the Simulation		
2	ILCHR2A	Launcher #2 for Fire Section A	(same as above)	--	--	
3	ILCHR3A	Launcher #3 for Fire Section A	(same as above)	--	--	

NAME: Riley Goodin
DATE 10/20/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT RESOURCES

RESOURCE NUMBER	RESOURCE LABEL	DESCRIPTION OF USE	RESOURCE ATTRIBUTES			TASK NUMBERS REQUIRING THE RESOURCE
			ATTRIBUTE NUMBER	DEFINITION	INITIAL VALUE	
4	ILCHR1B	Launcher #1 for Fire Section B	(same as above)	--	--	
5	ILCHR2B	Launcher #2 for Fire Section B	(same as above)	--	--	
6	ILCHR3B	Launcher #3 for Fire Section B	(same as above)	--	--	

NAME: Riley Goodin
DATE 10/20/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT RESOURCES

Page 2 of 2

3-0 VARIABLES

The forms in this section describe all the state variables, system attributes, and user variables used in the IHAWK system module. Table II-1, shown after the forms, expands the definition for the variable TSVALW.

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
SA(1)	Method of Control =1 Centralized =2 Decentralized =3 Independent =4 Autonomous	1	
SA(2)	Weapons Control Status =1 Weapons Tight =2 Weapons Hold =3 Weapons Free	1	
SA(3)	Not used		
SA(4)	Not used		
SA(5)	TCC Alert Status =1 On =2 Off	2	

TYPE OF VARIABLES: __ STATE VARIABLES X SYSTEM ATTRIBUTES __USER VARIABLES

NAME: Riley Goodin
DATE: 9/13/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT VARIABLES Page 1 of 1

VARIABLE	DEFINITION AND/OR DEFINING EQUATION	INITAL VALUE	COMMON BLOCK (USER VARIABLES ONLY)
MXTSKW	Maximum number of operator tasks for the 1HAWK. This is a FORTRAN PARAMETER.	40	-
TSVALW (MXTSKW,3)	Information is passed back from task sequencing to MSAINT task networks through TSVALW. For example, if task J is selected by task sequencing, the information for task J is passed back in row J of TSVALW (see Table II-1)	-	COM1Q
MCOLW	Maximum number of messages any operator can have simultaneously pending(PARAMETER)	10	COM1W
MSIDW (5,MCOLW)	Each column is the ID of a message pending for the operator doing task sequencing	-	COM1W
MSAAW (2,MCOLW)	Each column is the data base address of a pending message for the operator doing task sequencing	-	COM1W
THRETW(3,4)	Data on the Tracks found to be most threating for goals 2 (column 1),3(column 2), 8(column 3), and 1(column 4) Row 1 = MTRACK column of the track, Row 2 = Track status (column 5 of track data), Row 3 = Time till arrival (minutes)	-	COM1W

TYPE OF VARIABLES: __ STATE VARIABLES __ SYSTEM ATTRIBUTES X USER VARIABLES

NAME: J. Polito
DATE: 8/25/83

AIR DEFENSE SYSTEM MODULE: 1HAWK
PROJECT: NOPADS

SAINT VARIABLES

Page 1 of 1

Table II-1

TSVALW DEFINITIONS BY IHAWK OPERATOR TASKS

TASK	COLUMN 1	COLUMN 2	COLUMN 3
1			
2	NTRACK column of track	THREAT	-
3			
4			
5			
6	Message priority of highest priority message	First word of the data base address	Second word of the data base address
7			
8	Message priority of Message 28	First word of the data base address	Second word of the data base address
9			
10			
11			
12			
13	Message priority of Message 30	First word of the data base address	Second word of the data base address
14	Message priority of Message 31	First word of the data base address	Second word of the data base address

Table II-1 (continued)

TSVALW DEFINITIONS BY HAWK OPERATOR TASKS			
TASK	COLUMN 1	COLUMN 2	COLUMN 3
15	Message priority of Message 32	First word of data base address	Second word of data base address
16			
17			
18	Message priority of TCC Alert message	First word of data base address of message	Second word of data base address of message
19	NTRACK column of Track to identify	-	-
20			
21			
22	NTRACK column of the Track to process	Fire section to assign	Track data row of the track
23	NTRACK column of the Track to process	Fire section to assign	Track data row of the track
24			
25			
26	NTRACK column of the Track to assign	Fire section to preempt	Track data row of the track
27			

Table II-1 (continued)

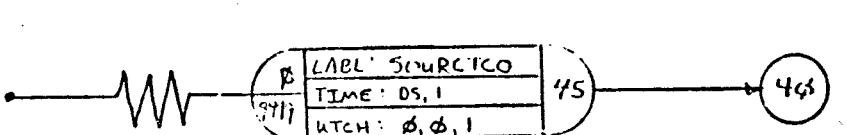
TSVALW DEFINITIONS BY IHAWK OPERATOR TASKS			
TASK	COLUMN 1	COLUMN 2	COLUMN 3
28	NTRACK column of friendly track being engaged	Fire section to preempt	-
29			
30			
31			
32	Message priority of highest priority message	First word of data base address of the message	Second word of the data base address of the message
33			
34			
35			
36			
37			
38			
39			
40			

4-0 TASK NETWORKS

In this section are the forms used to document each of the operator tasks and each individual node in those tasks. Each operator task is documented by two types of forms: MSAINT TASK MODELS and MSAINT TASK NETWORKS. The MSAINT TASK MODEL forms give a written description of the activities involved of each node on the network. The MSAINT TASK NETWORKS forms contain the MSAINT subnetworks used to describe each operator task. Following these two forms for each task is the TASK NODE SPECIFIC DATA forms for each node in the operator task (MSAINT subnetwork). These forms contain the information stored in the data base for each node.

Note that the source nodes, the dead node, and the task sequencing subnetwork are also contained in this section. These particular nodes are not shown on the MSAINT TASK MODEL forms because there are no operator actions involved.

Table II-2, at the end of this section, is a cross-reference table that may be used to look up which operator tasks are associated with which nodes.



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
TCO Source Node	TCO	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 45

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

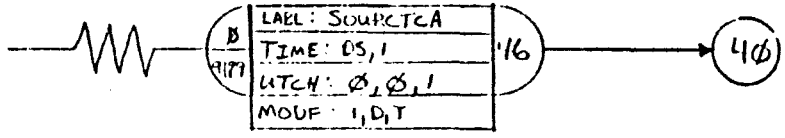
TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
45	SOURCTCO	—	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Page 1 of 1



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
TCA Source Node	TCA	

MSAINT TASK NETWORKS

TASK NODE: 46

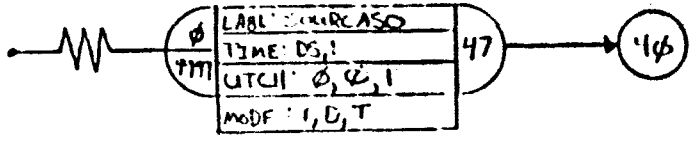
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NON-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
46	SOUPCTCA	—	(above)	(above)	(above)	1.0

NAME: J. Riley Coodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
ASO Source Node	ASO	

MSAINT TASK NETWORKS

TASK NODE: 47

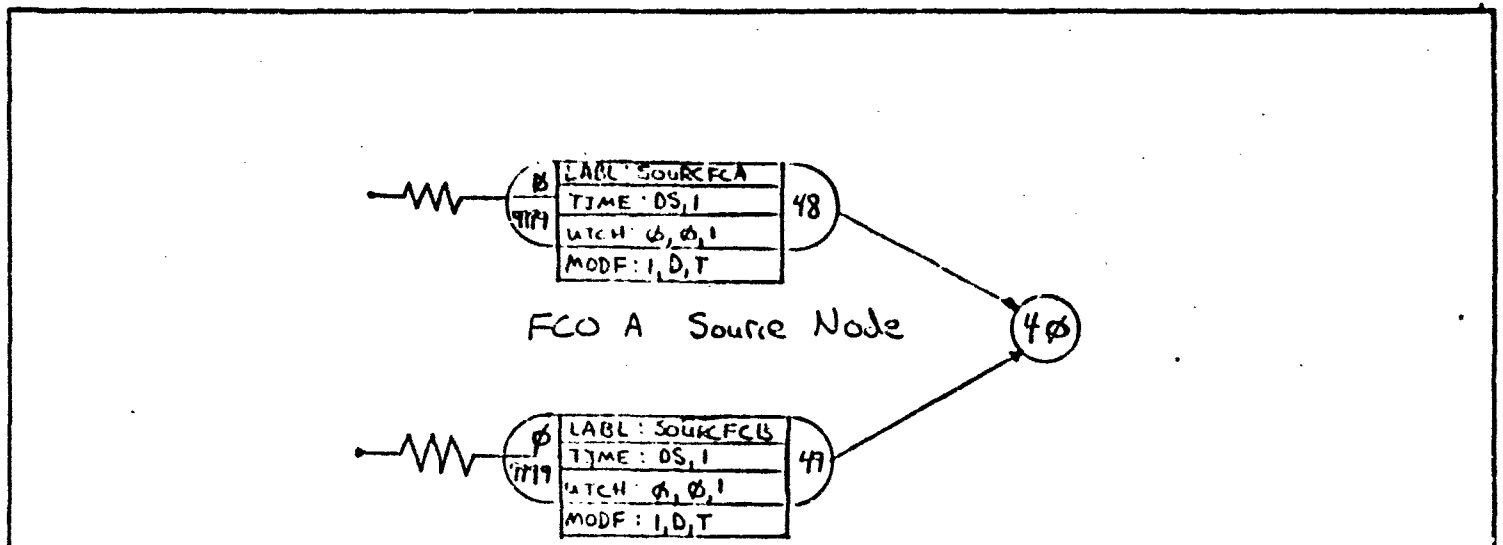
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	1.000000
SKILL-WEIGHT	10.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
47	SOURCASS	—	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
FCO A and FCO B Source Tasks	FCO A/FCO B	

MSAINT TASK NETWORKS

TASK NODE: 48

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSERVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
48	SOURCFCA	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 49

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

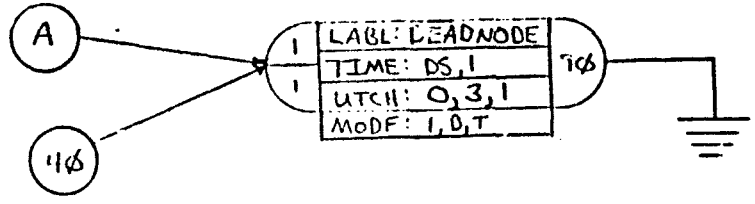
TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
49	SOURCFCB	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

A= All other operators besides the one at Node 40 that are routed to the dead node.



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
DEAD NODE	All IHAWK Operators	

MSAINT TASK NETWORKS

TASK NODE: 90

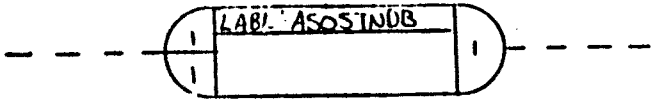
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
90	DEADNODE	-	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1



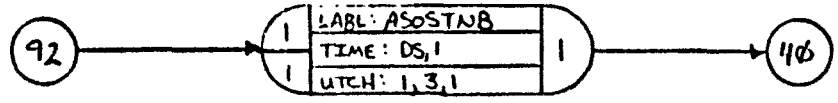
Observe CRT for new target video or track other targets

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
1) ASO Standby, Wait for Action	ASO	TH 9-1430-1526-12-1, Table 2-26

NAME: Riley Goodin
DATE: 10/38/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
1)ASO Standby, Wait for Action	ASO	

MSAINT TASK NETWORKS

TASK NODE: 1 ,

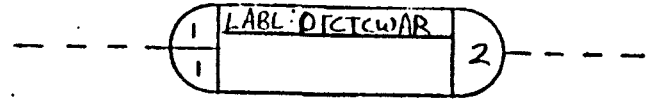
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	50.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
1	ASOSTNB	1	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

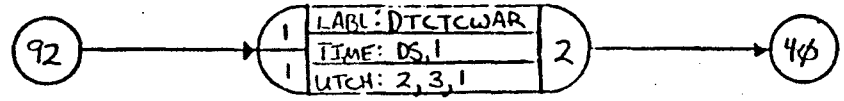


Observe target video (blip) on CRT. and hear target doppler in head set.

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
2) Detect Target on CWTDC	ASO	TM 9-1430-1526-12-1,Table 2-26

NAME: Riley Goodin DATE: 10/31/83	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
2)Detect Target on CWTDC	ASO	

MSAINT TASK NETWORKS

TASK NODE: 2

DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
2	DTCTCWAR	2	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

Observe indicators (velocity, range, azimuth sector) and establish target priority.

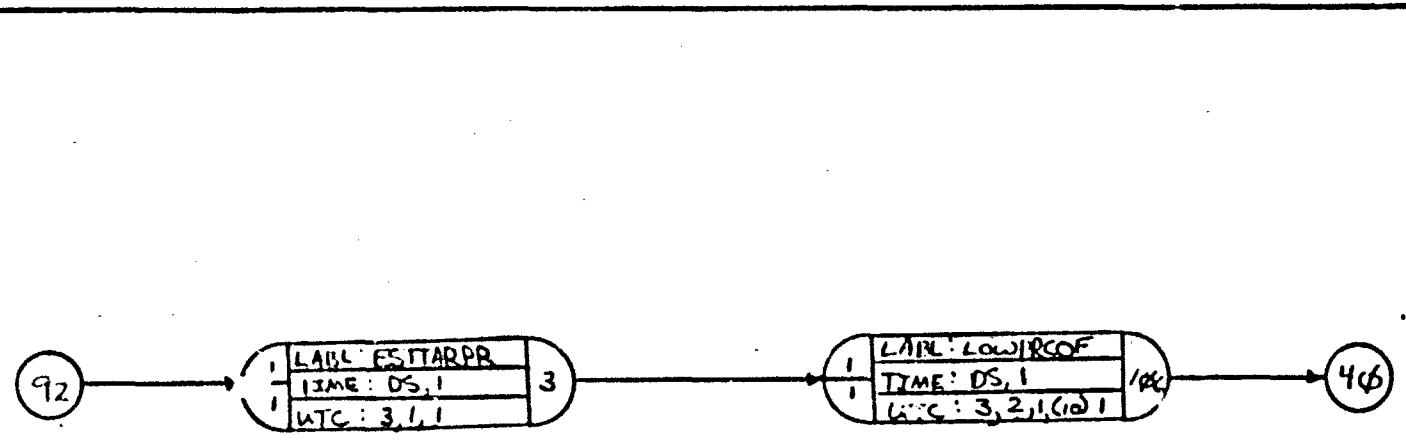
Has a lower priority target been confirmed.

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
3) Establish Target Priority	ASO	TM 9-1430-1526-12-1, Table 2-26

NAME: Riley Goodin
DATE: 11/2/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
3)Establish Target Priority	SDO	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 3

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	50.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	NOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
3	ESTTARPI	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: NOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 106

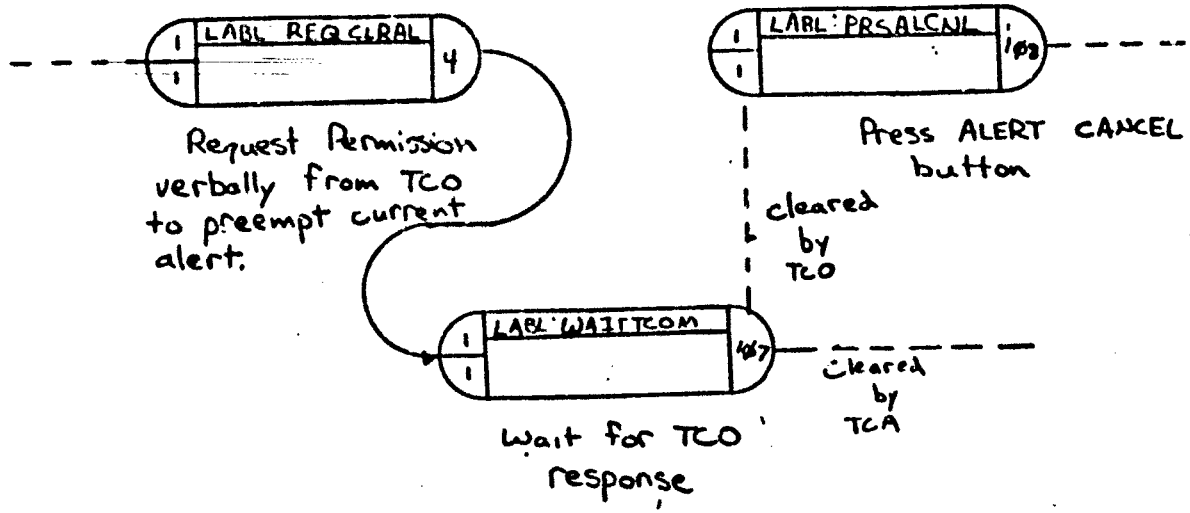
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
13	OWPRCOF	3	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

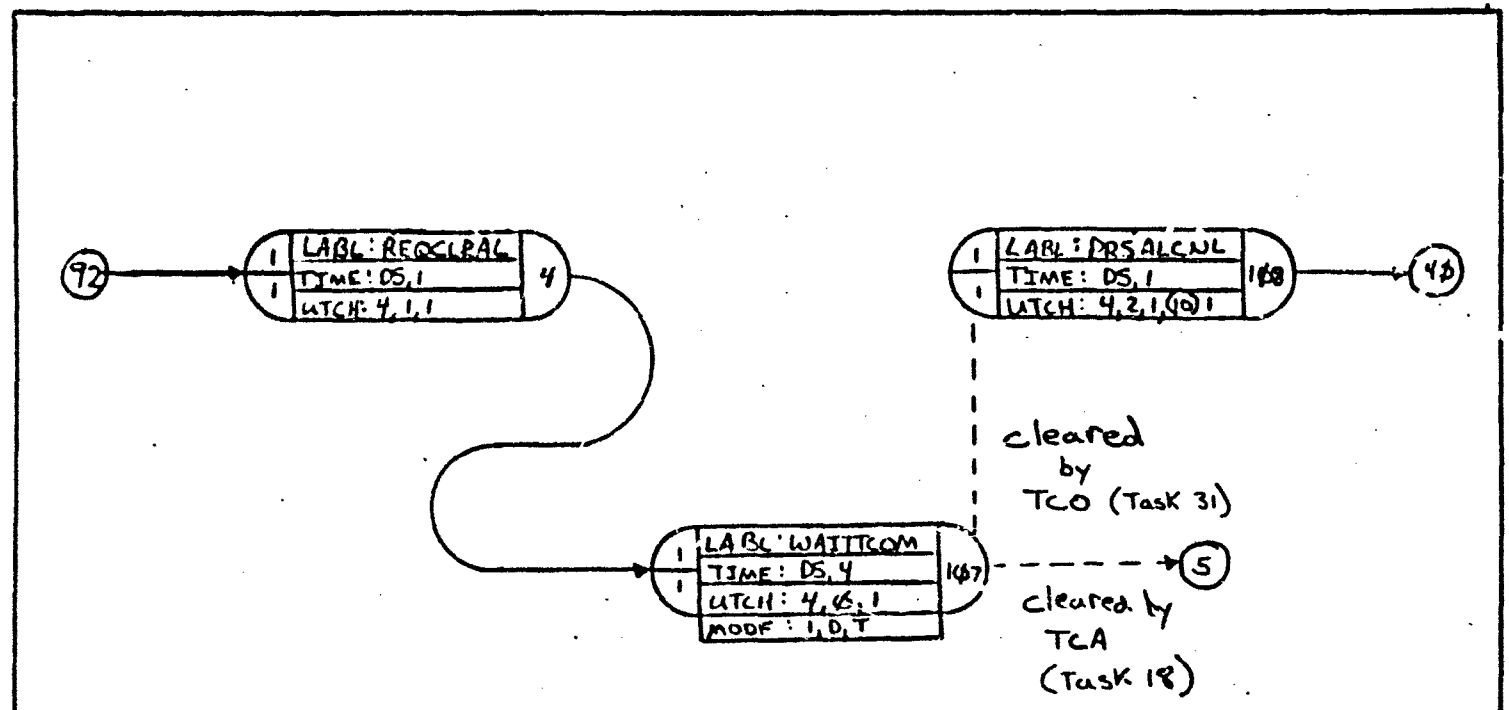


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
4) Preempt Lower Priority Target	ASO	TM 9-1430-1526-12-1,Table 2-26

NAME: Riley Goodin
DATE: 11/2/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
4)Preempt Lower Priority Target	ASO	

NSAINT TASK NETWORKS

Zero time - no skills

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
4	REQLCRAL	4	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 107

DISTRIBUTION-TYPE	1.000000
MEAN	0.1000000E+11
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
107	WAITTCOM	4	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 108

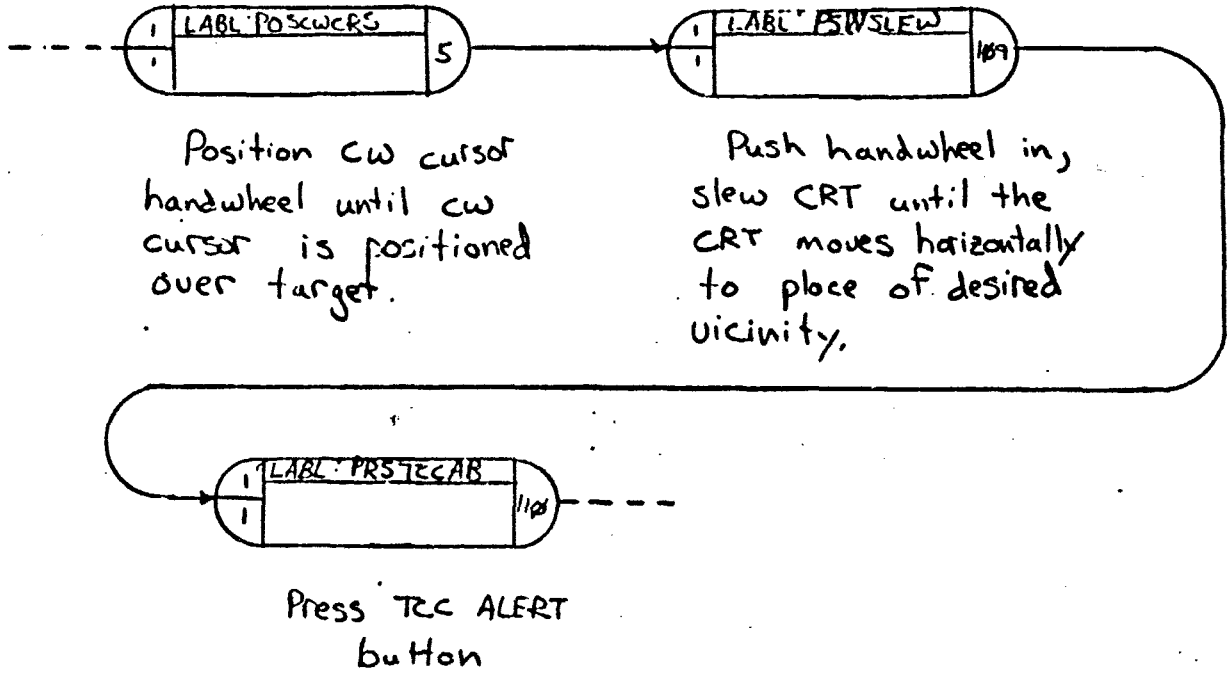
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
108	PRSALCNL	4	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK.
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

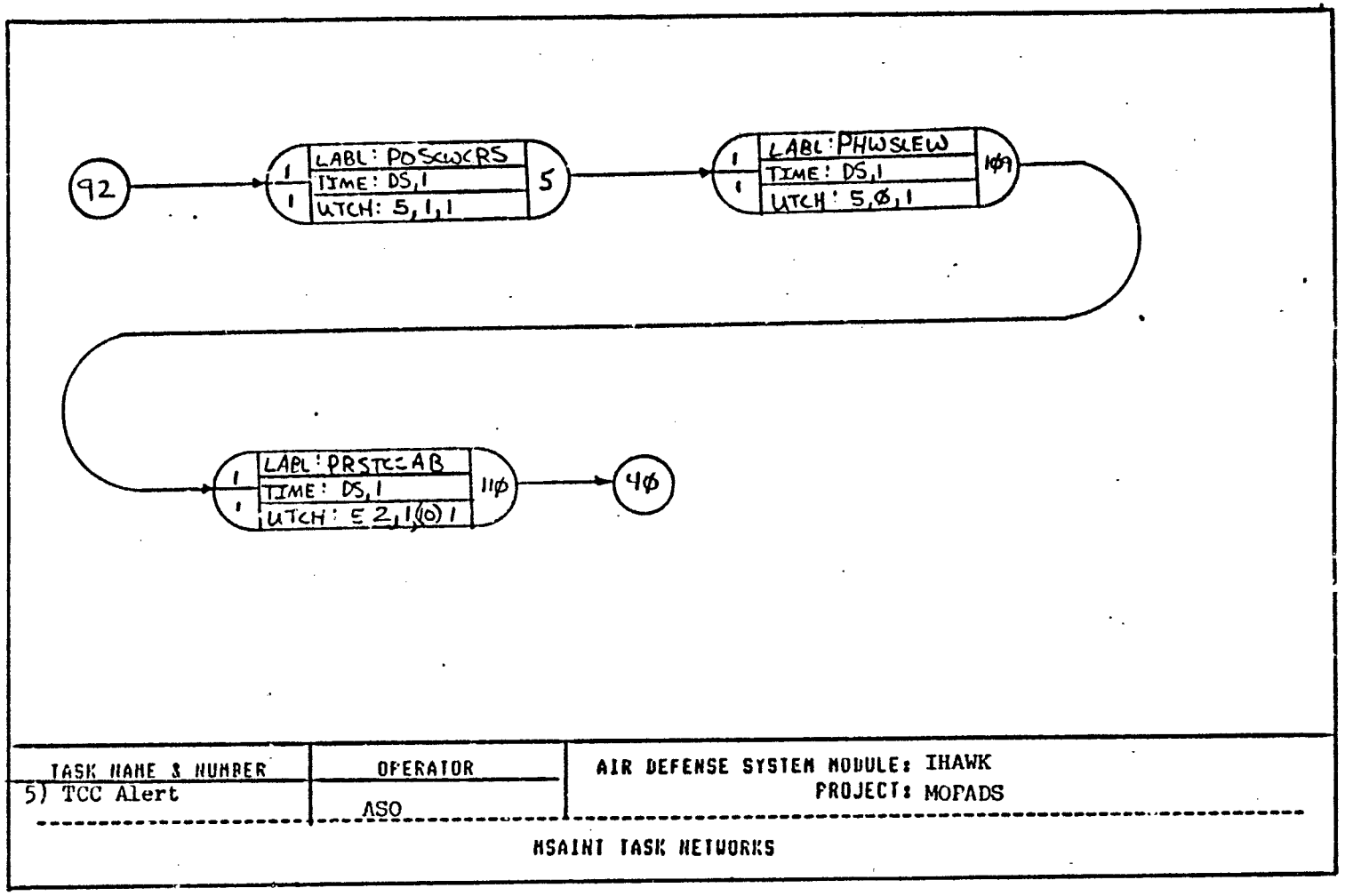


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
5) TCC Alert	ASO	TM 9-1430-1526-12-1,Table 2-26

NAME: Riley Goodin
DATE: 11/1/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
5) TCC Alert	ASO	

MSAINT TASK NETWORKS

TASK NODE: 109

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
109	PSWSLEW	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 110

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
110	PRSTCCAB	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 5

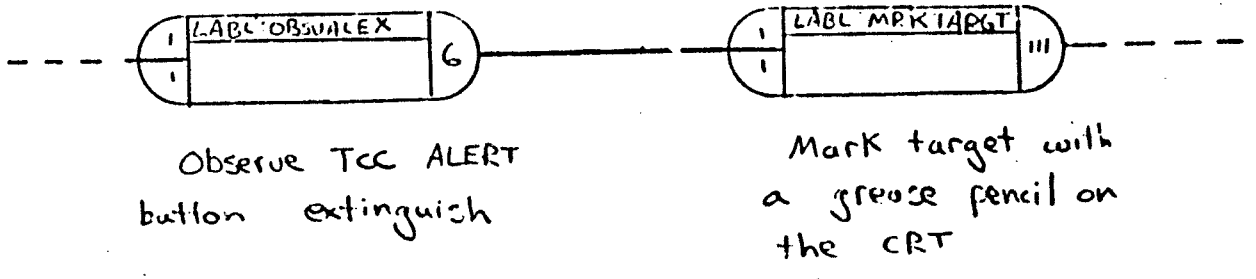
DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
5	POSCWCRS	5	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
6) Mark Target As Accepted By TCC	ASO	TM 9-1430-1526-12-1, Table 2-26

NAME: Riley Goodin
DATE: 11/1/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS

TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
6)Mark Target As Accepted by TCC	ASO	

MSAINT TASK NETWORKS

TASK NODE: 6

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
6	OBSVALEX	6	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 111

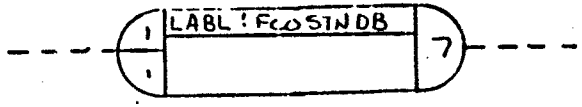
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
111	MRKTARGT	6	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

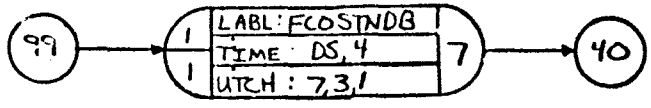


Standby wait for action. Observe CRT & panels. Also monitor engagement status (if one exist for the File Section)

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
7) FCO Standby, Wait for Action	FCO A/B	-

NAME: Riley Goodin DATE: 5/20/83	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM/MODULE: IHAWK PROJECT: MOPADS
()FCO Standby, Wait for Action	FCO A or B	

MSAINT TASK NETWORKS

TASK NODE: 7

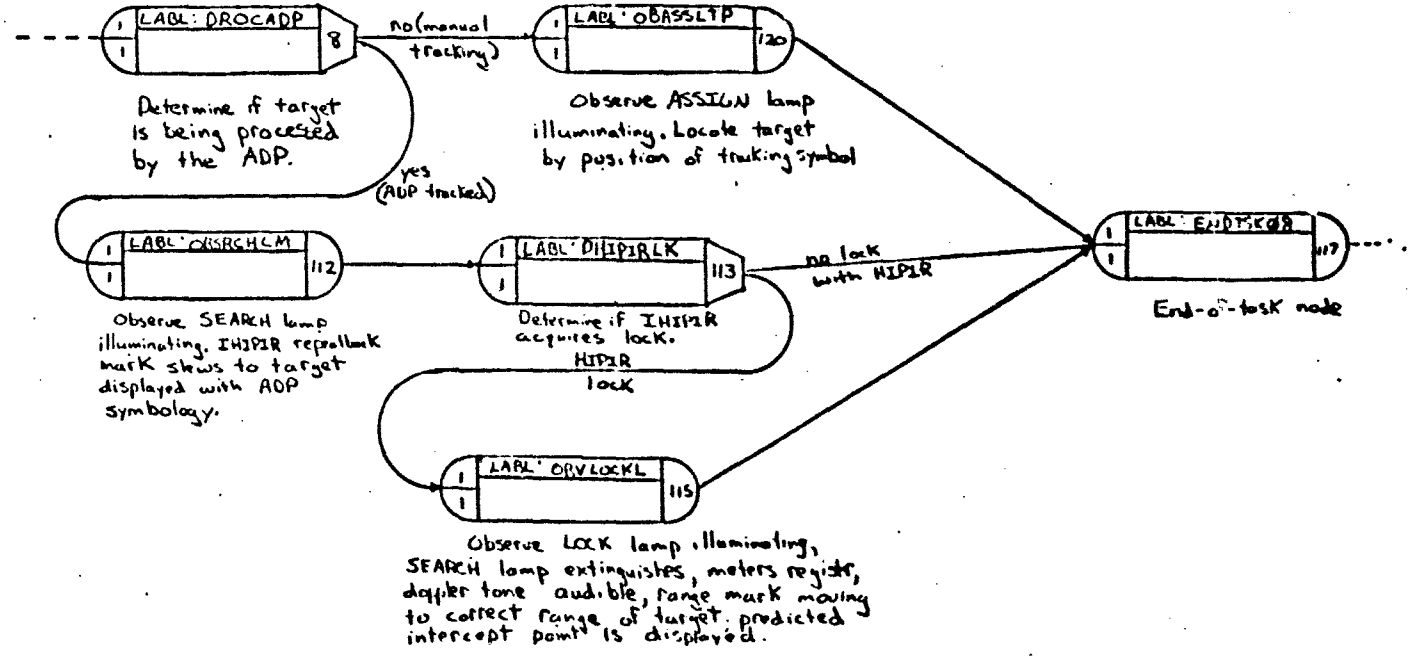
DISTRIBUTION-TYPE	1.000000
MEAN	0.1000000E+11
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
7	FCOSTNDB	7	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

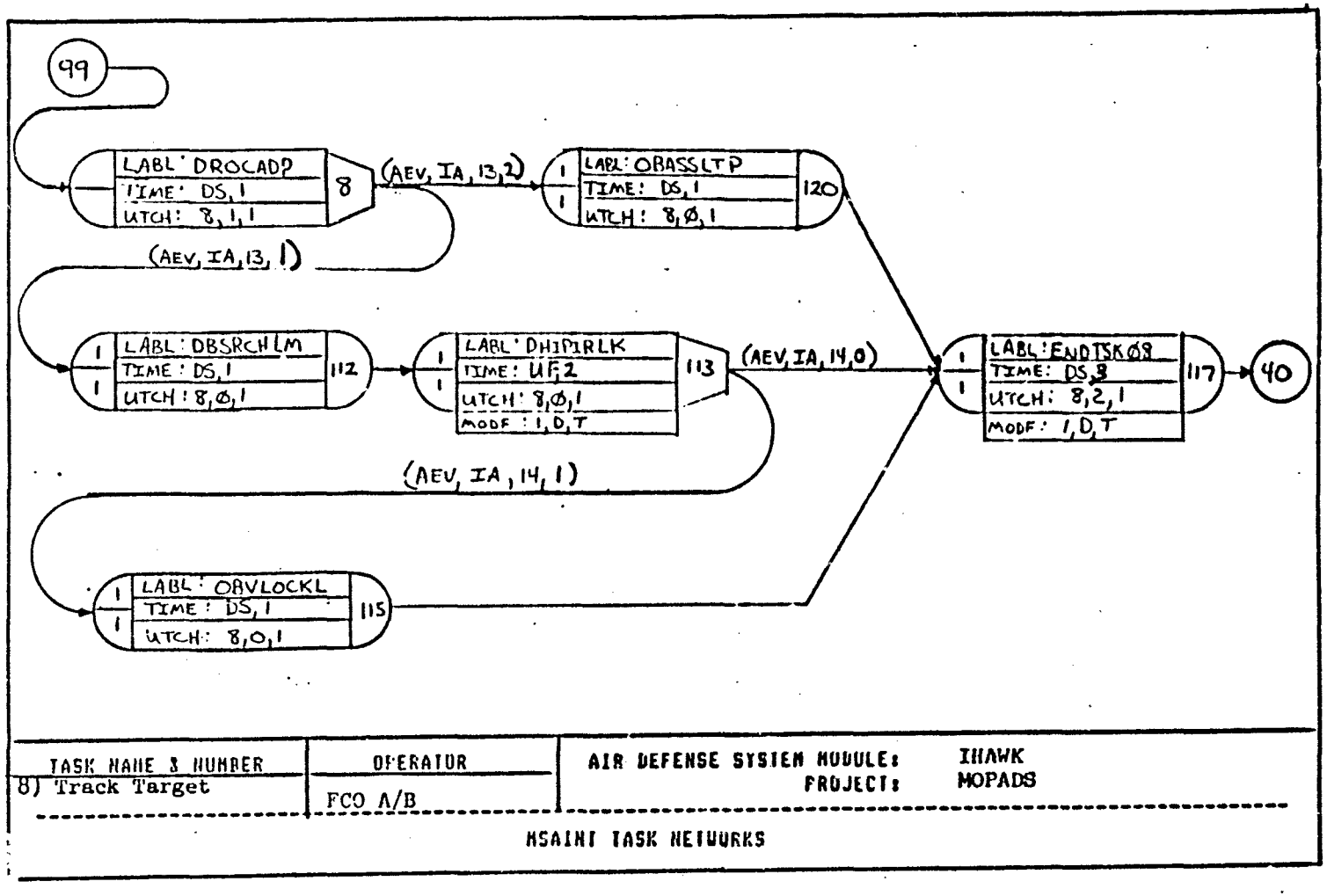


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
8) Track Target	FCO A/B	TM 9-1430-1526-12-1, Table 2-25

NAME: Riley Goodin
DATE: 10/19/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS


99
LABL: DROCADP
TIME: DS,1
UTCH: 8,1,1
8
(AEV, IA, 13, 2)
1
1
LABL: OBASSLTP
TIME: DS,1
UTCH: 8,0,1
120
(AEV, IA, 13, 1)
1
1
LABL: DBSRCHLM
TIME: DS,1
UTCH: 8,0,1
112
1
1
LABL: DHIPIRLK
TIME: UF,2
UTCH: 8,0,1
MODF: 1,D,T
113
(AEV, IA, 14, 0)
1
1
LABL: ENDTSK08
TIME: DS,8
UTCH: 8,2,1
MODF: 1,D,T
117
40
(AEV, IA, 14, 1)
1
1
LABL: OBVLOCKL
TIME: DS,1
UTCH: 8,0,1
115
TASK NAME & NUMBER
8) Track Target
OPERATOR
FCO A/B
AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS
MSAINT TASK NETWORKS

TASK NODE: 8

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
8	DROCADP	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 112

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
112	OBSRCHLM	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 113

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
113	IHIPIRLK	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1985

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 115

DISTRIBUTION-TYPE	8.000000
MEAN	0.5830000E-01
STANDARD-DEVIATION	0.2040000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	80.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
115	OBVLOCKL	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 117

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
117	ENDTSKØ8	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 120

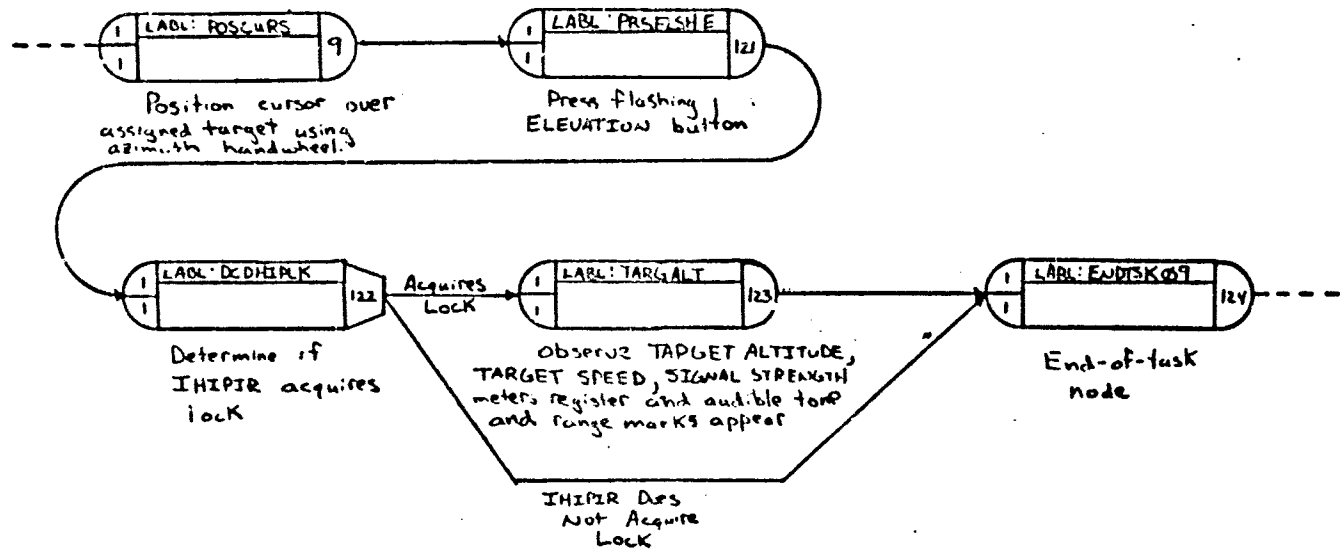
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
120	OBASSLTP	8	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

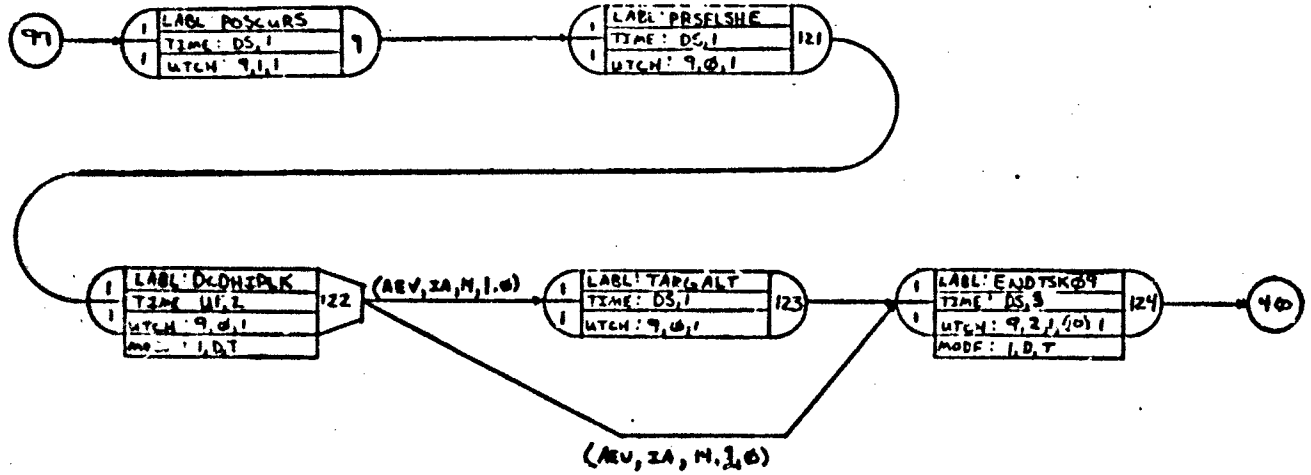


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
9) Obtain Lock On Target Manually	FCO A/B	TM 9-1430-1526-12-1,Table 2-25

NAME: Riley Goodin
DATE: 10/19/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
9)Obtain Lock On ..Target Manually	FCO A/B	

MSAINT TASK NETWORKS

TASK NODE: 9

DISTRIBUTION-TYPE	8.000000
MEAN	0.2500000E-01
STANDARD-DEVIATION	0.8750000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSERVE-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
9	POSCURS	9	(above)	(above)	(above)	1.0

NAME: J. Riley Coodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: M-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 121

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
121	PRSFLSHE	9	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 122

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
122	DCDHIPLK	9	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 123

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000
SKILL-INDEX	0.000000
SKILL-WEIGHT	00.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
123	TARGALT	9	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 124

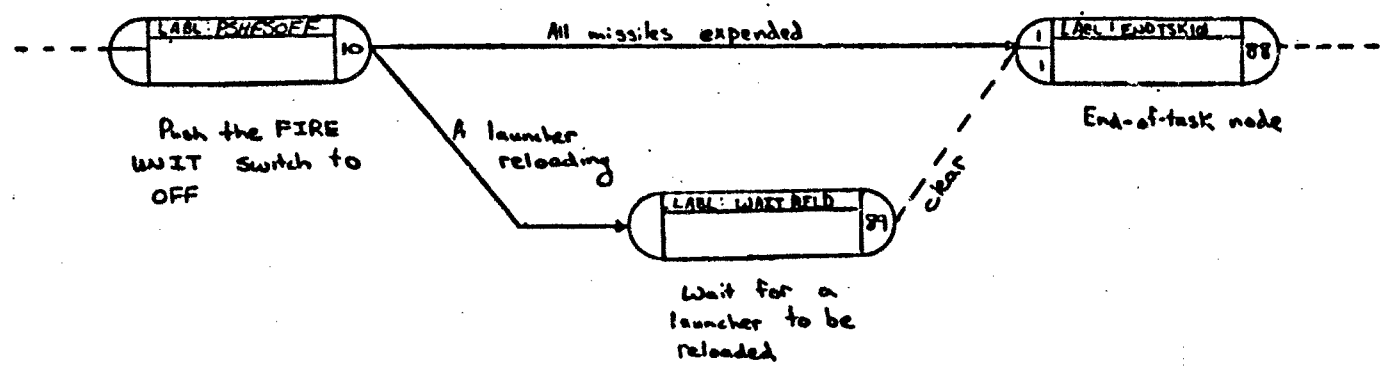
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
124	ENDTSKØ9	9	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

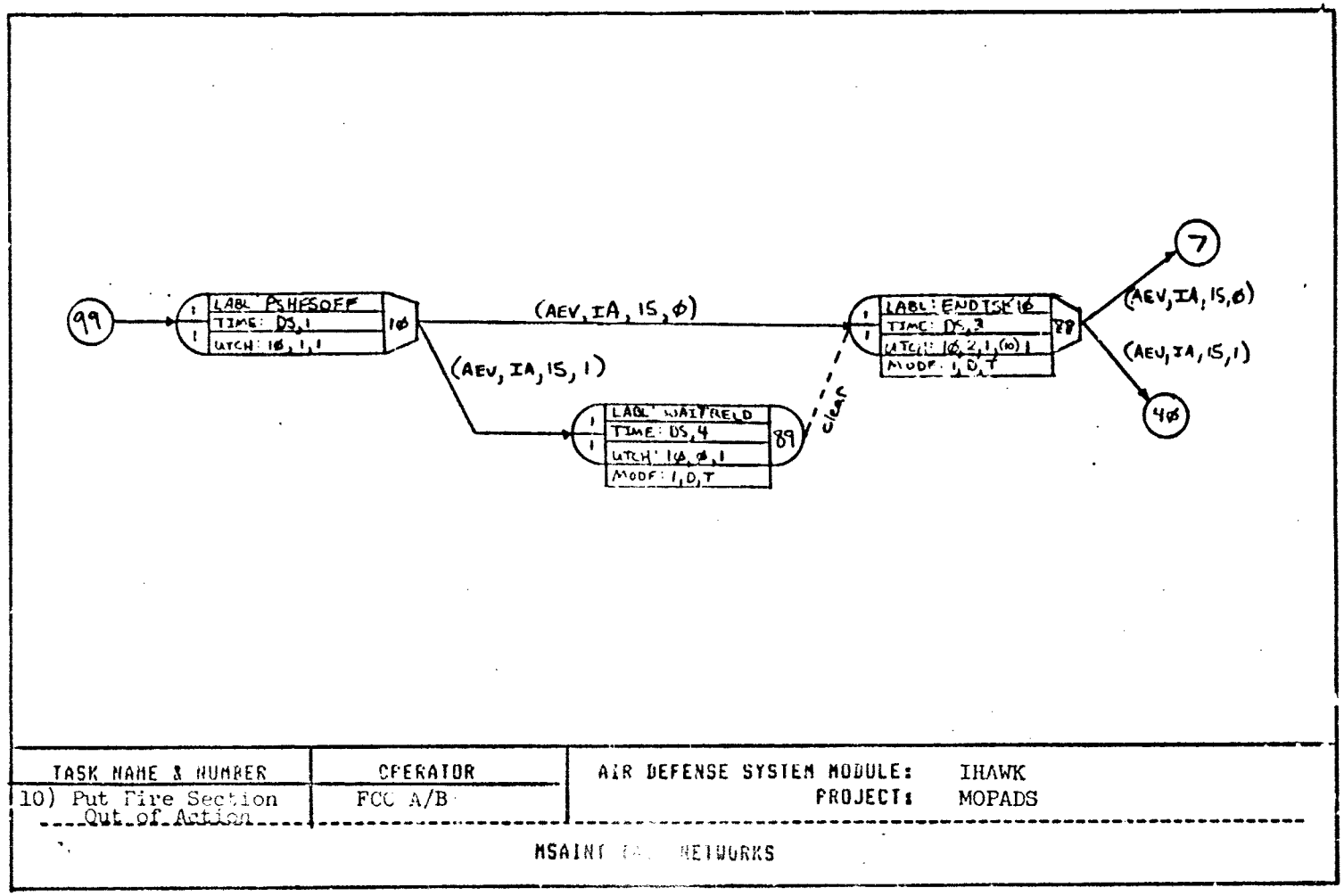


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
10) Put Fire Section Out of Action	FCO A/B	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 10/25/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
10) Put Fire Section Out of Action	FCC A/B	

MSAINT [illegible] NETWORKS

TASK NODE: 10

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRV-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
10	PSHFSOFF	10	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 88

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
88	ENDTSK10	10	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 89

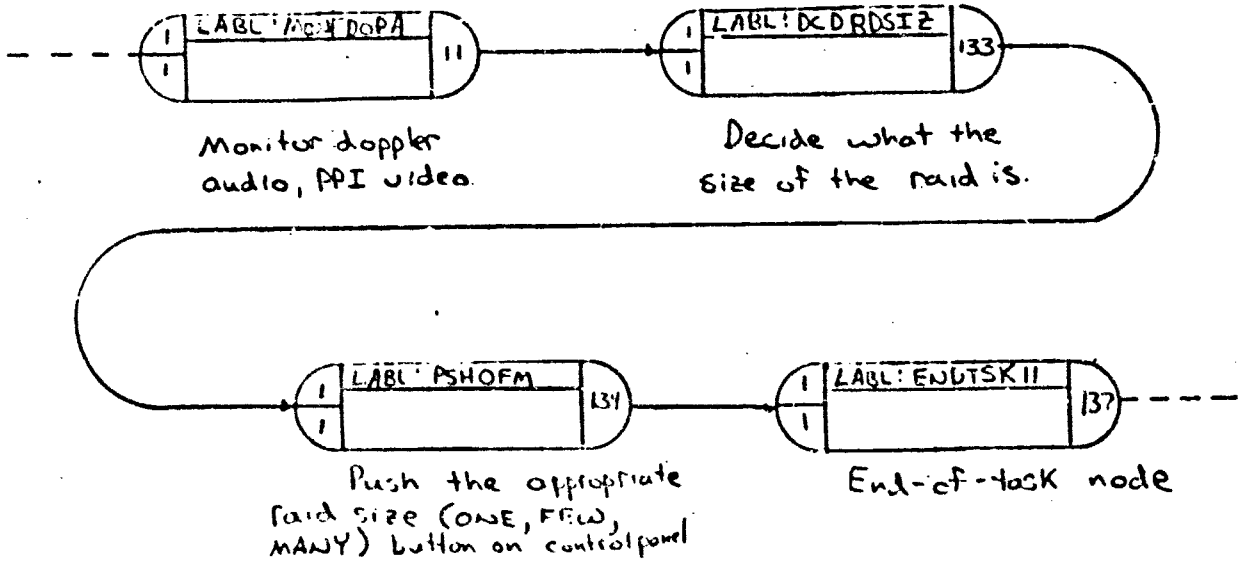
DISTRIBUTION-TYPE	1.000000
MEAN	0.1000000E+11
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
89	WAITRELI	89	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

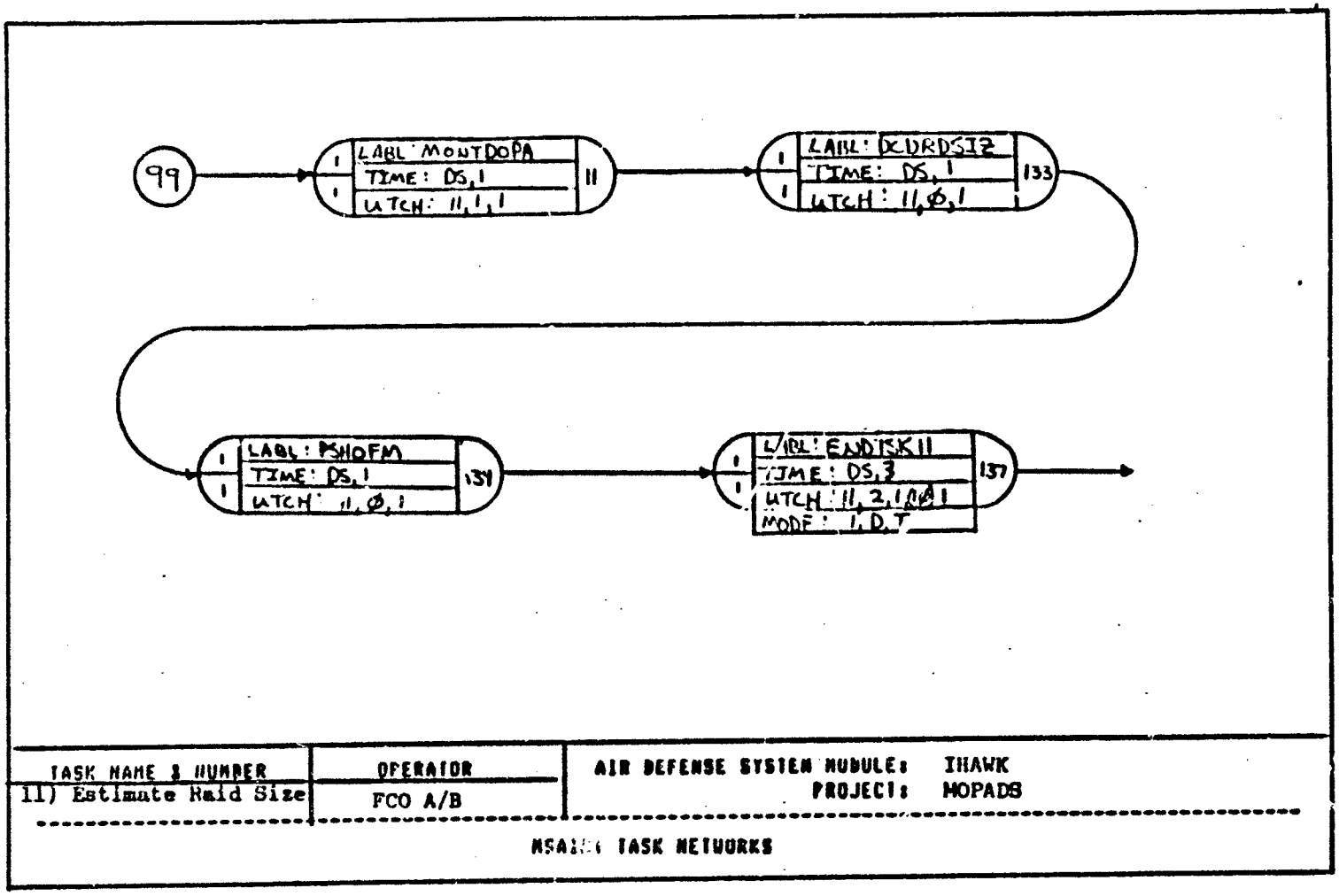


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
11) Estimate Raid Size	FCO A/B	TM 9-1430-1526-12-1,Table 2-25

NAME: Riley Goodin
DATE: 10/19/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
11) Estimate Raid Size	FCO A/B	PROJECT: MOPADS

NSA[illegible] TASK NETWORKS

TASK NODE: 11

DISTRIBUTION-TYPE	3.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
11	MONTDOPA	11	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 133

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	0.0000000E+00
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	5.000000
SKILL-WEIGHT	20.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	40.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	40.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
133	DCDRDSIZ	11	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 134

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	NOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
134	SHOFM	11	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: NOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 137

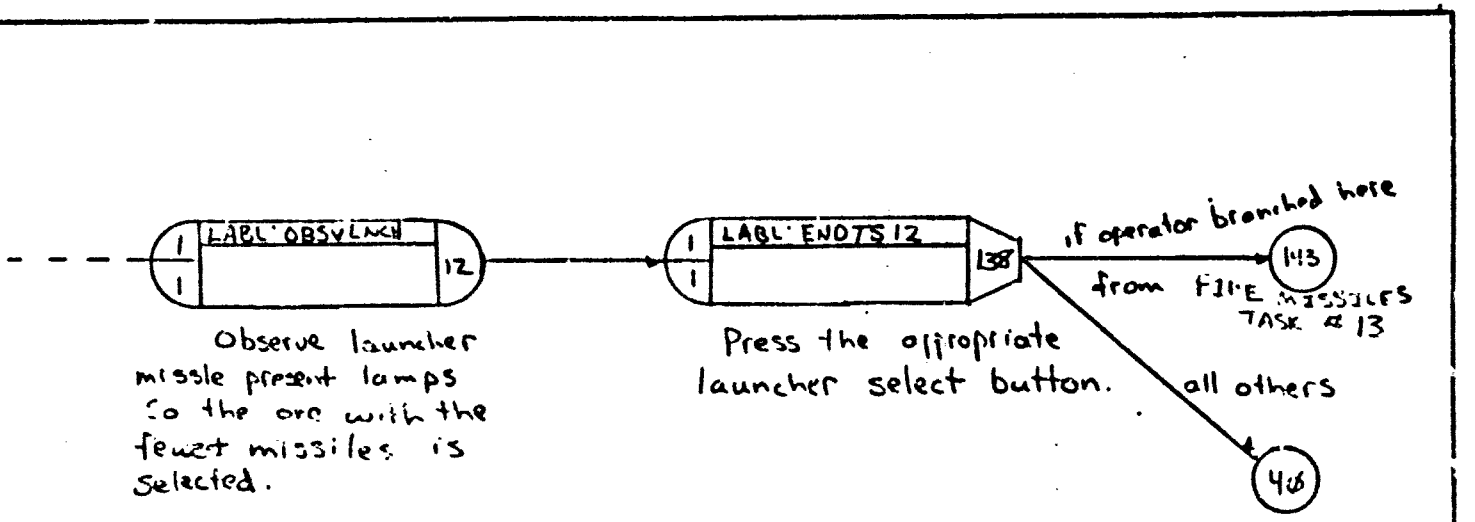
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
137	FHITSK11	11	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

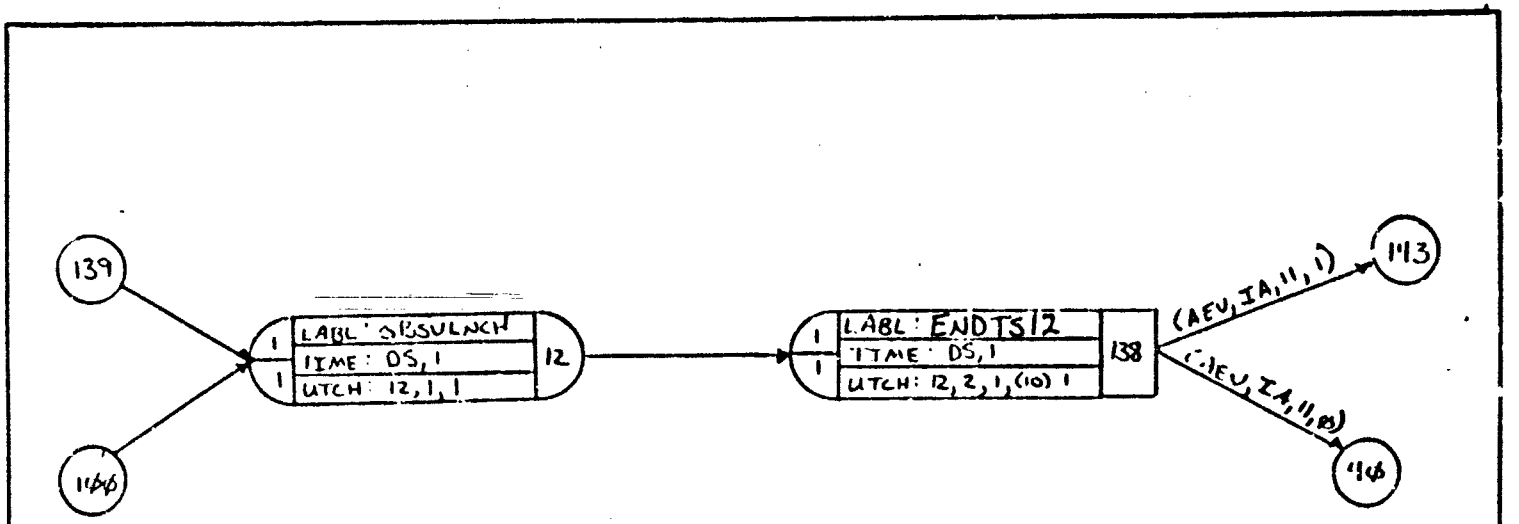


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
12) Select Launcher	FCO A/B	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 10/20/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE:	IHAWK
12) Select Launcher	FCQ_A/B	PROJECT:	MOPADS

MSAINT TASK NETWORKS

TASK NODE: 12

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	5.000000
SKILL-WEIGHT	40.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	60.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
12	OBSVLNCH	12	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IIIAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 138

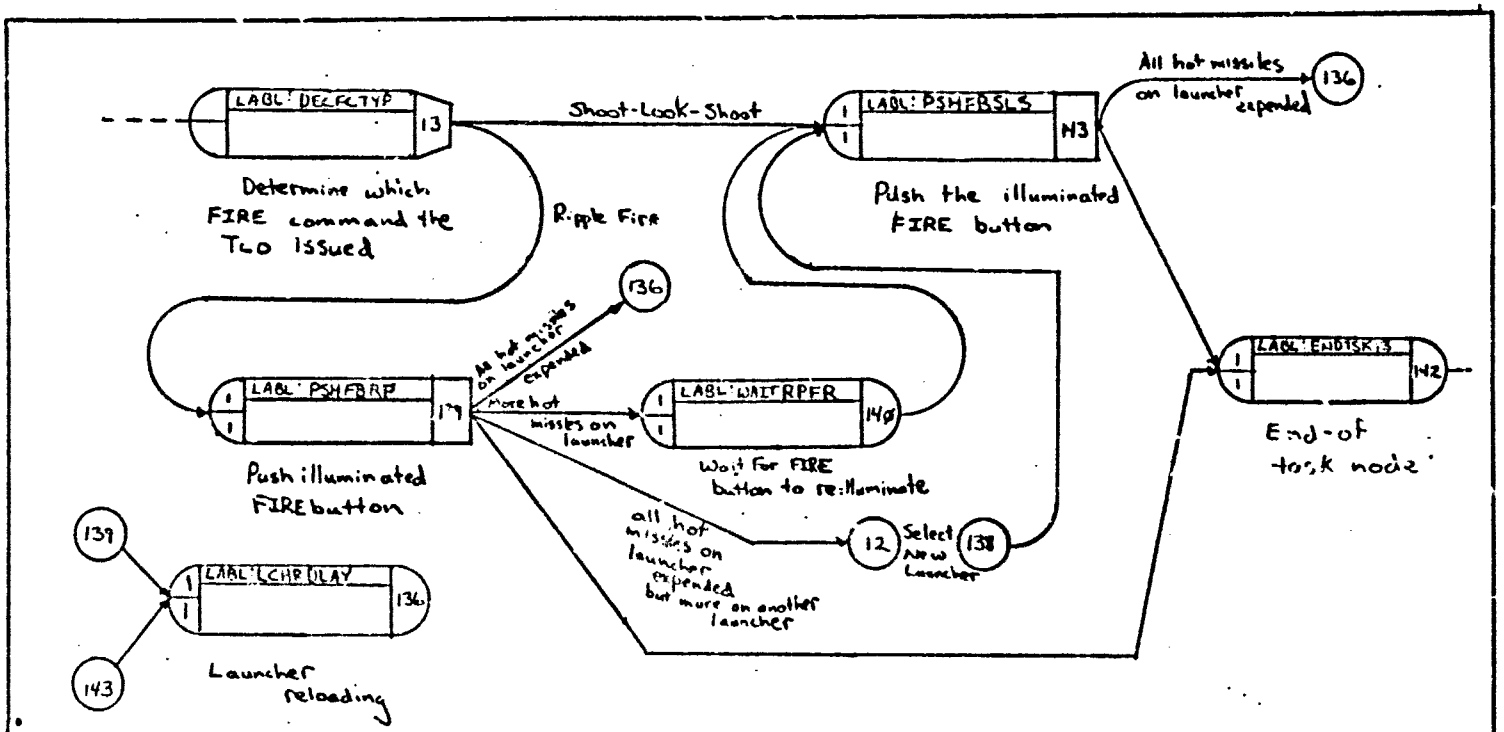
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
138	ENDTS12	12	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

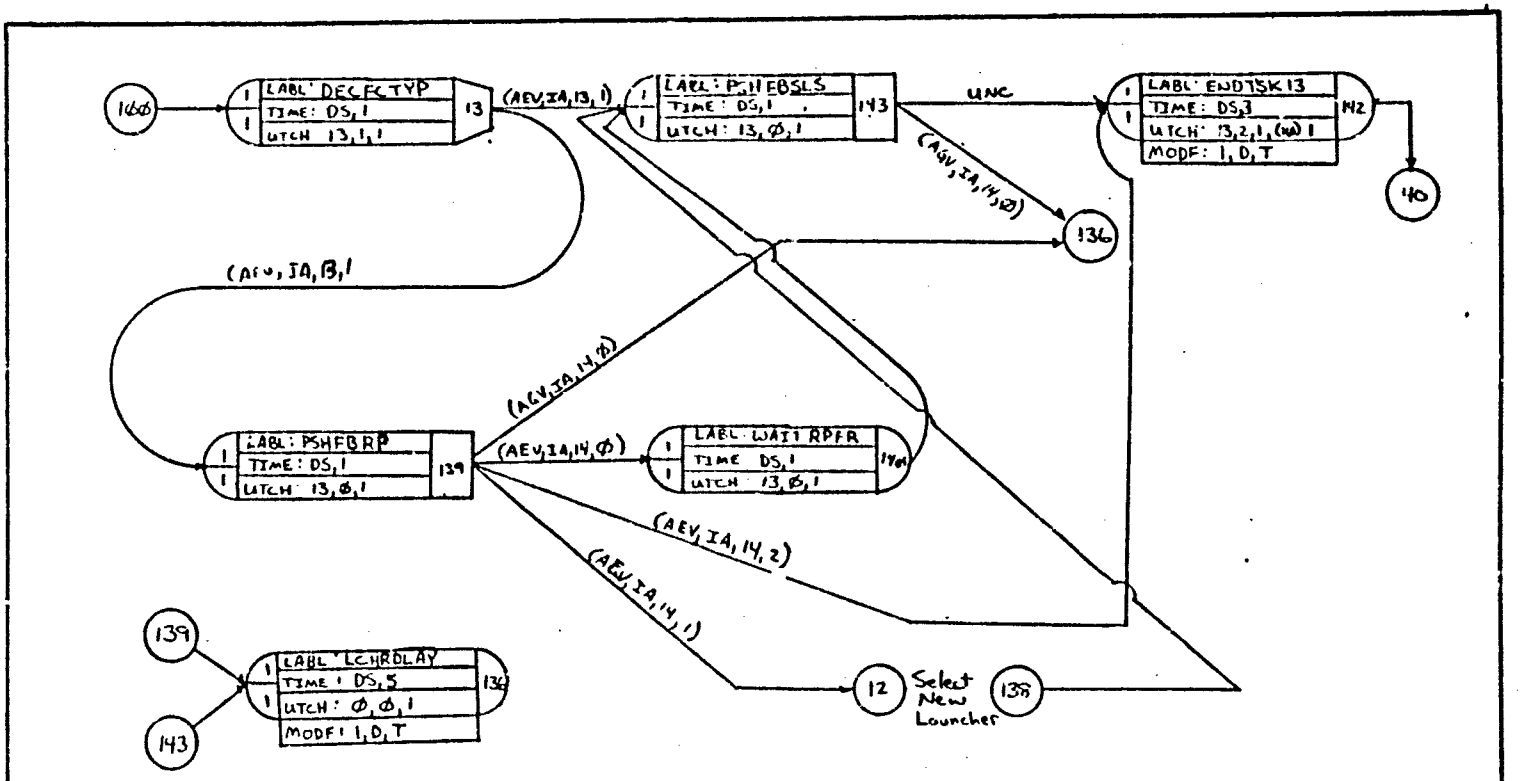


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
13) FIRE MISSILES	FCO A/B	TM 9-1430-1526-12-1,Table 2-25

NAME: Riley Goodin
DATE: 10/23/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
13) Fire Missiles	FCO A/B	

MSAINT TASK NETWORKS

TASK NODE: 13

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	1.000000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
13	DECFCTYP	13	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Control Entity: No Skills

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
136	LCHRDLAY	13	15.0	0	0	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 139

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-SIM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

201-X

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
139	PSHFBRP	13	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 140

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
140	WAITRPFR	13	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 142

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
142	ENDTSK13	13	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 143

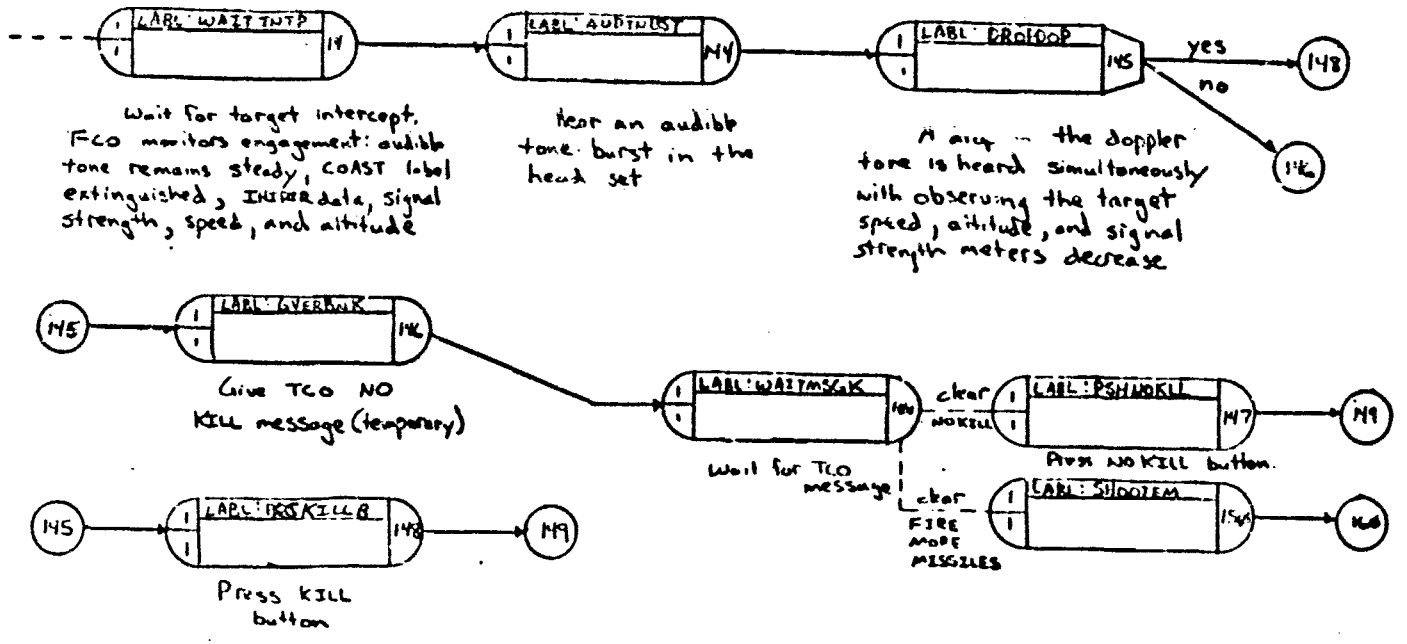
DISTRIBUTION-TYPE	8.000000
MEAN	6.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
143	PSHFBSLS	13	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

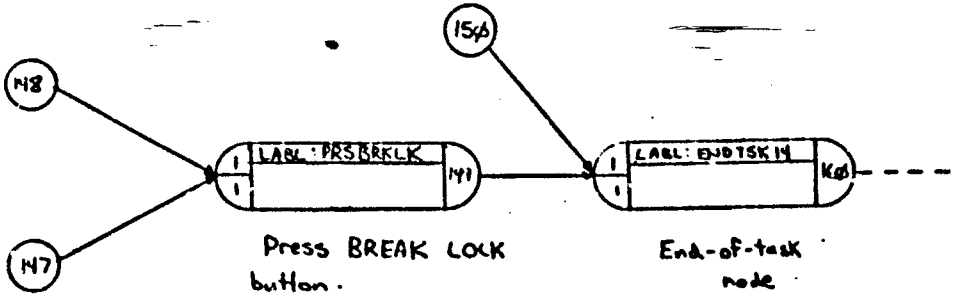


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
14) Evaluate Target Intercept	FCO A/B	TM 9-1430-1526-12-1,Table 2-25

NAME: Riley Goodin
DATE: 10/27/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

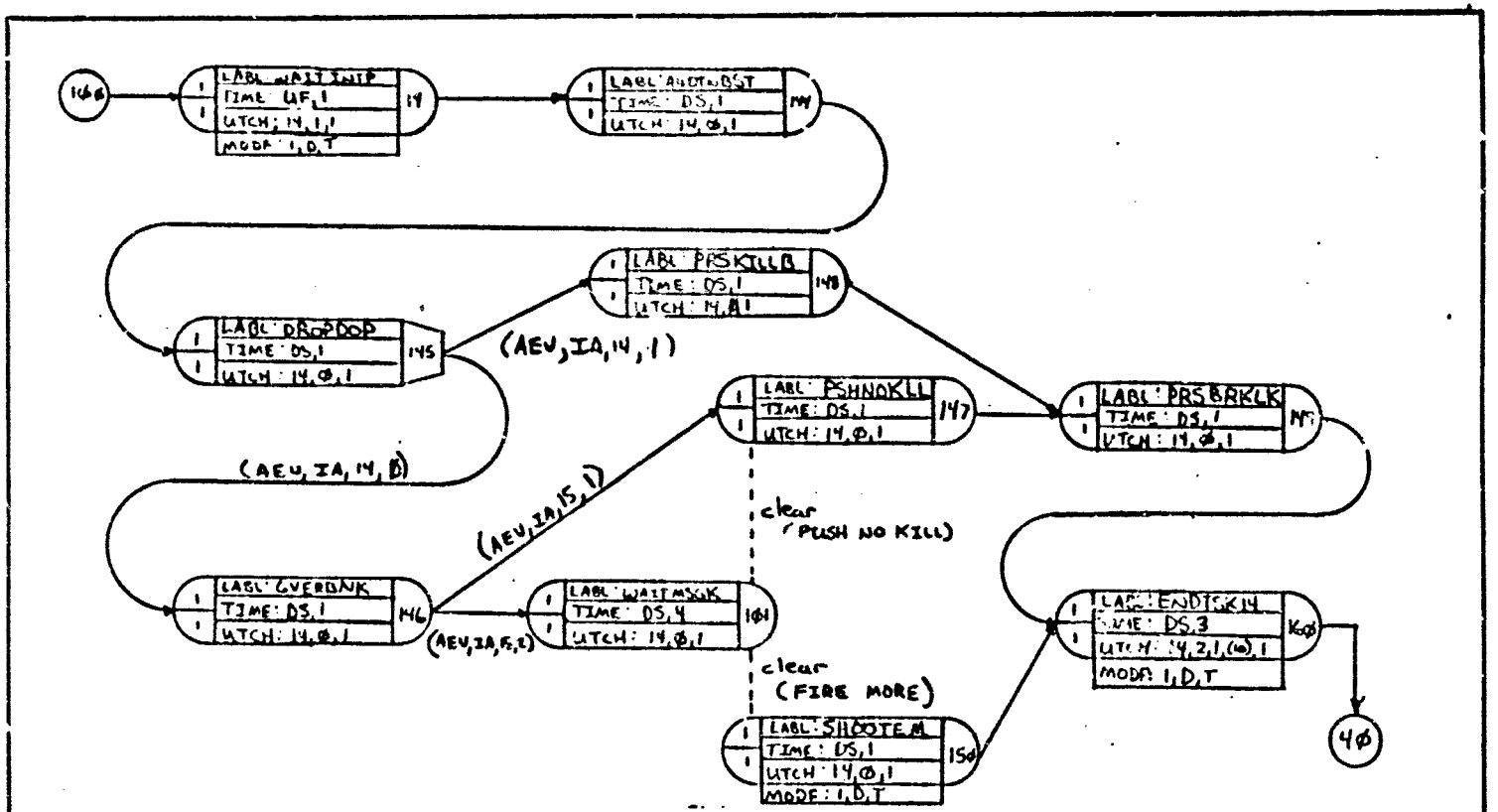
MSAINT TASK MODELS



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
14) Evaluate Target Intercept	FCO A/B	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 10/27/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
14) Evaluate Target Intercept	FCO A/B	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 14

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
14	WAITINTP	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 101

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
101	WAITMSGK	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 144

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	1.000000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
144	AUDTNBST	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 145

DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
145	DROPDOP	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 146

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSERVE-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
146	OVERBNK	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 147

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	80.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
147	PSHNOKLL	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 148

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
148	PRSKILLB	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 149

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
149	PRSBRKLK	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 150

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
150	SHOOTEM	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 160

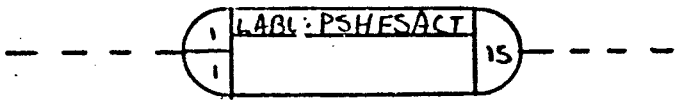
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
160	ENDTSK14	14	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-1HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



Push the FIRE UNIT. Switch to ACTIVE

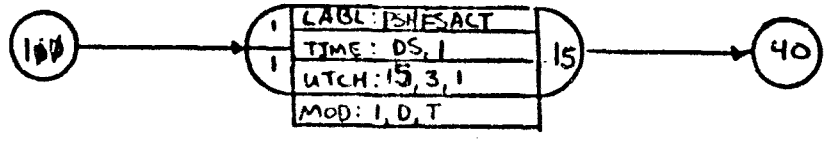
A-119

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
15) Put Fire Section Back In Action	FCO A/B	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 10/26/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
15) Put Fire Section Back in Action	FCO A/B	PROJECT: MOPADS

MSAINT :ASK NETWORKS

TASK NODE: 15

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
15	PSHFSACT	15	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Observe CHANGE TARGETS button flash or hear "break lock" command from TCO, DESTROY label illuminates for 3 seconds (if missiles have been filed).

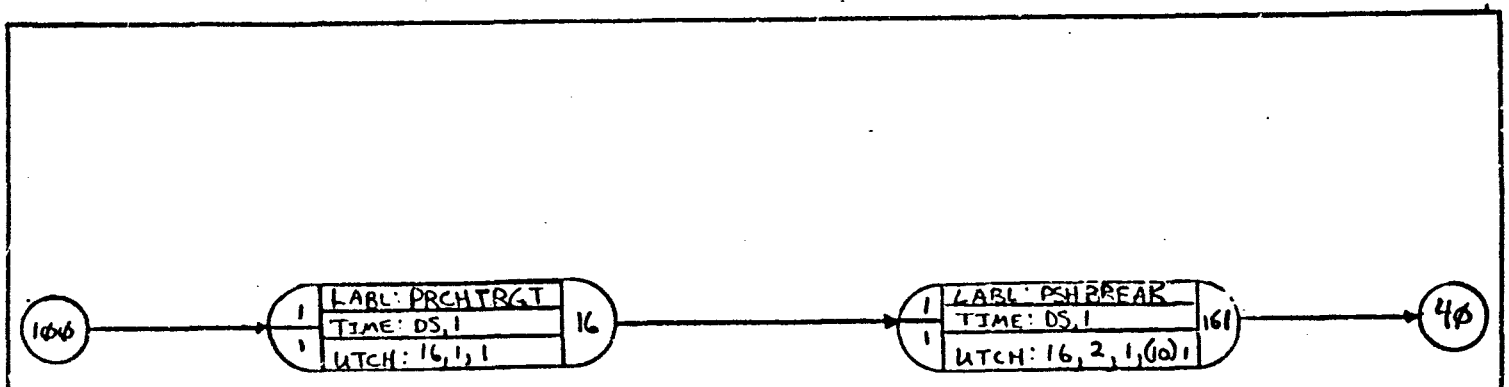
Push BREAK LOCK, observe LOCK lamp extinguish

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
16) Process Change Targets Command	FCO A/B	TM 9-1430-1526-12-1, Table 2-25

NAME: Riley Goodin
DATE: 8/26/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
16) Process Change Targets Command	FCO A/B	

MSAINT TASK NETWORKS

TASK NODE: 16

DISTRIBUTION-TYPE	8.000000
MEAN	0.6670000E-01
STANDARD-DEVIATION	0.2330000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	11.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
16	PRCHTRGT	16	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 161

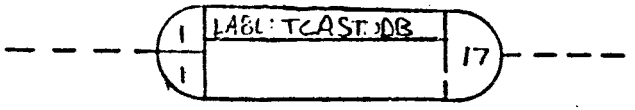
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
161	PSHBREAK	16	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



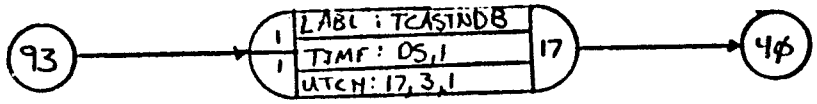
ASO monitors IFFN equipment, CRT, and indicators.

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
17) TCA Monitor CRT, Controls, Indicators	TCA	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
17) TCA Monitor CRT, Controls Indicators	TCA	

MSAINT TASK NETWORKS

TASK NODE: 17

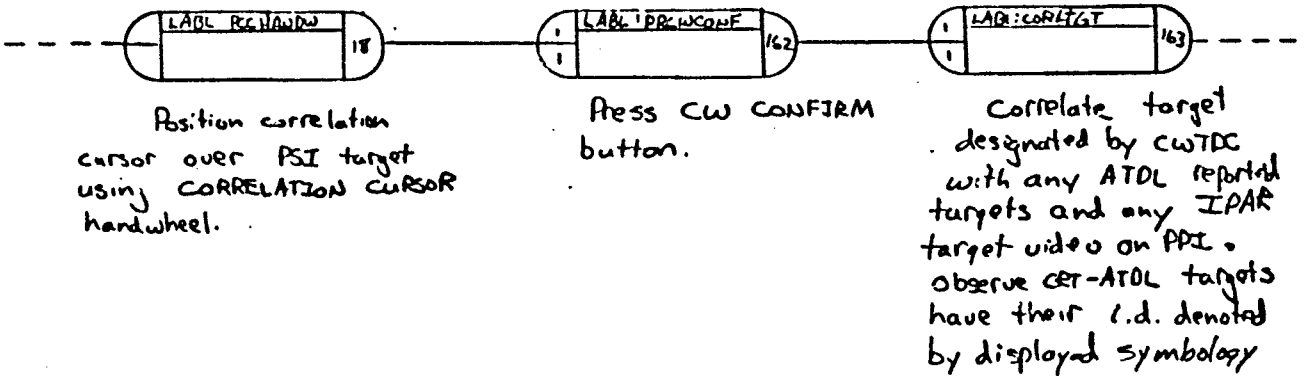
DISTRIBUTION-TYPE	9.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
17	TCASINDB	17	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

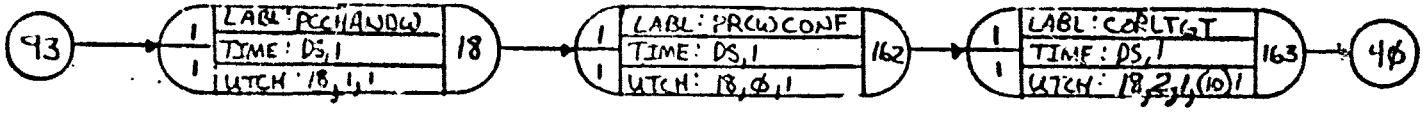


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
18) Accept CWTDC Target From AGO	TCA	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin
DATE: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
18) Accept CWTDC Target From ASQ	TCA	

NSAINT TASK NETWORKS

TASK NODE: 18

DISTRIBUTION-TYPE	8.000000
MEAN	0.5830000E-01
STANDARD-DEVIATION	0.2040000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
18	PCCHANDW	18	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 162

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
162	PRCWCONF	18	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IIHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

TASK NODE: 163

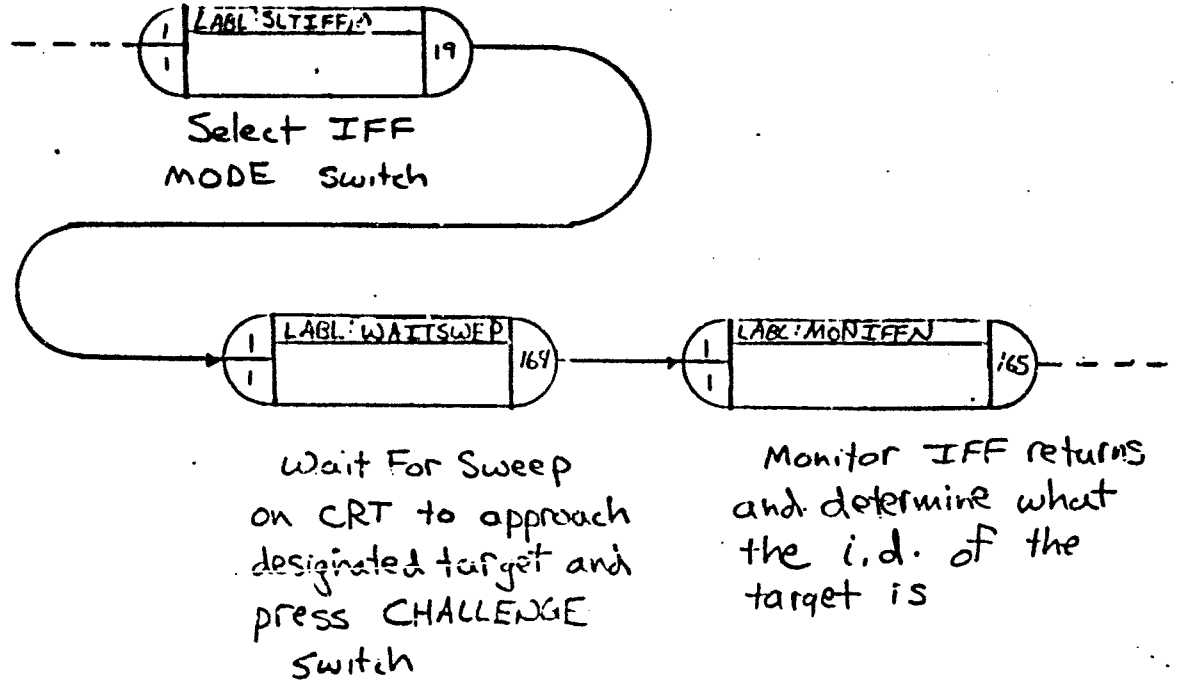
DISTRIBUTION-TYPE	8.000000
MEAN	0.8330000E-01
STANDARD-DEVIATION	0.2920000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	6.000000
SKILL-WEIGHT	30.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
163	CORLTGT	18	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

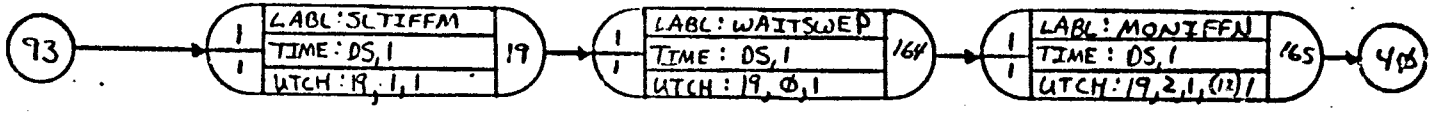


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
19) IFF CHALLENGE	TCA	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin
DATE: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
19) IFF CHALLENGE	TCA	

MSAINT TASK NETWORKS

TASK NODE: 19

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
19	SLTIFFM	19	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 164

DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	70.00000
SKILL-INDEX	13.00000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
164	WAITSWEP	19	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 165

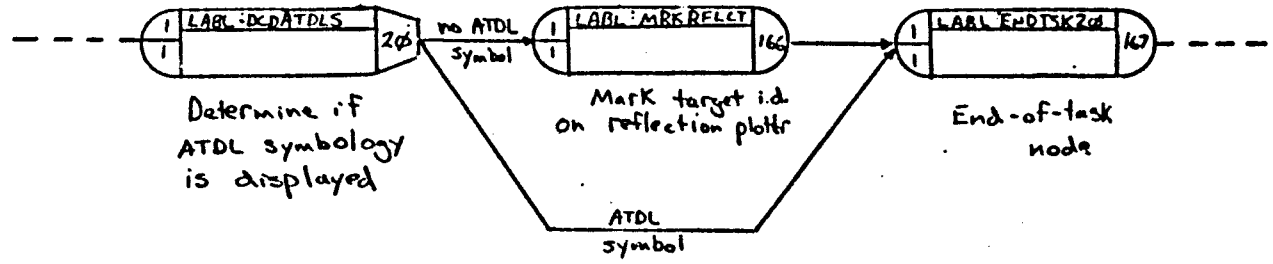
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	90.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	10.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
165	MONIFFN	19	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

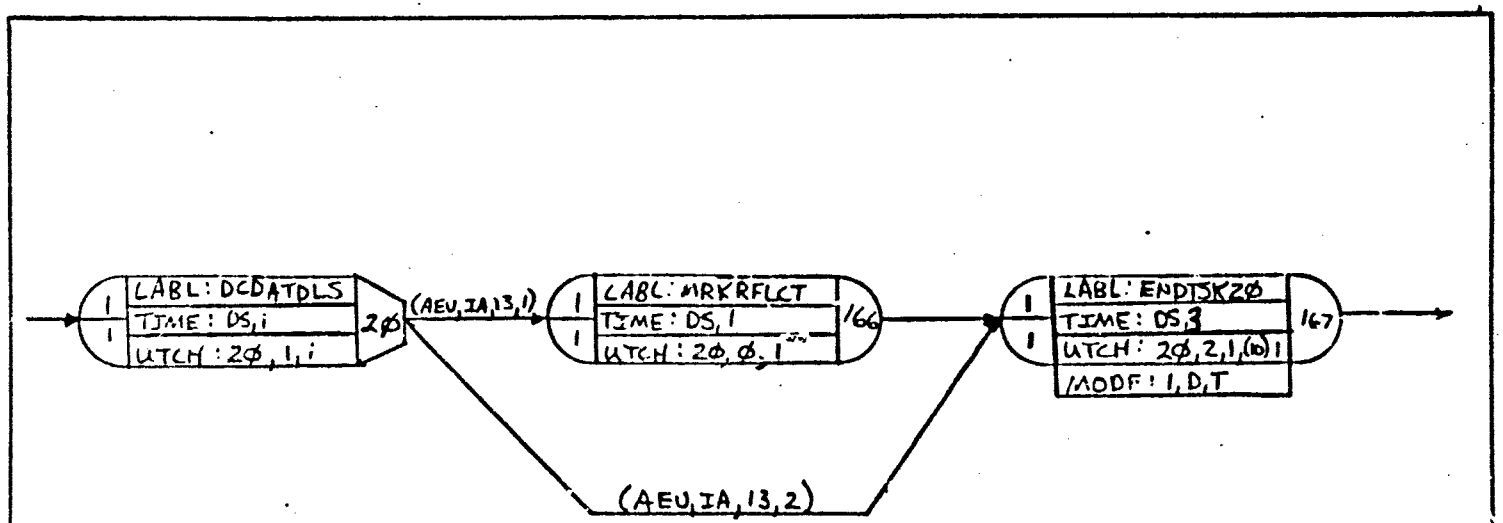


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
20) Mark On Reflection Plotter (Friend,Hostile,Unknown)	TCA	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin
DATE: 11/3/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
20) Mark Target On Reflection Plotter	TCA	

MSAINT TASK NETWORKS

TASK NODE: 20

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	6.000000
SKILL-WEIGHT	70.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
20	DCDATDLS	20	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 166

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
166	MRKRFLCT	20	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 167

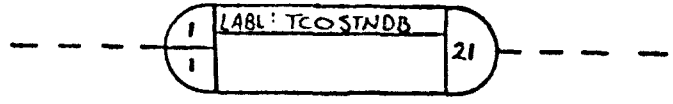
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
167	ENDTSK20	20	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

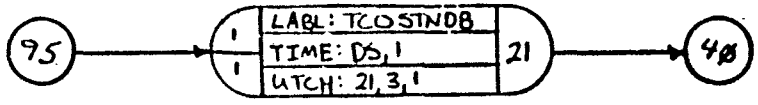


Monitor CRT,
IBCC Status Indicator,
and Communications Network

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
21) TCO Monitor CRT,Controls, Indicators	TCO	TM 9-1430-1526-12-1

NAME: Riley Goodin DATE: 11/4/83	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS

NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
21 TCO Monitor CTR, Controls,....	TCO	

MSAINT TASK NETWORKS

TASK NODE: 21

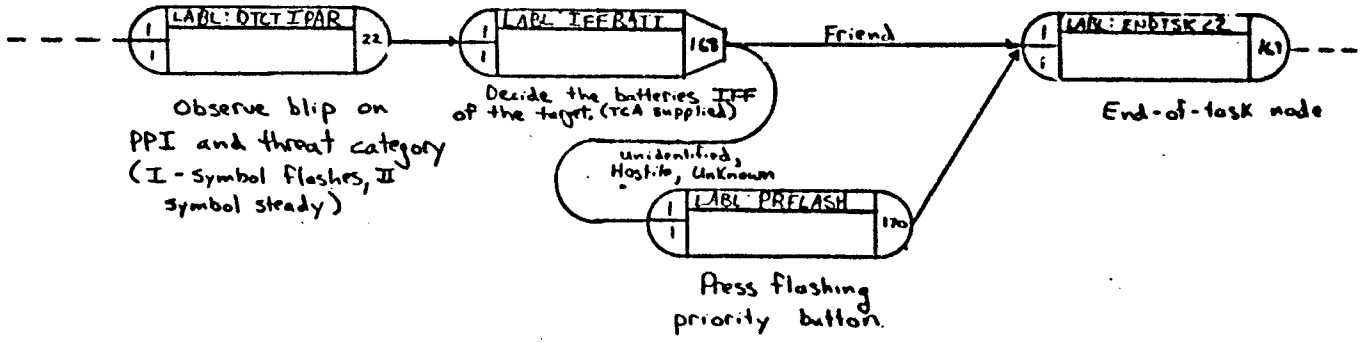
DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
21	TCOSTNDB	21	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

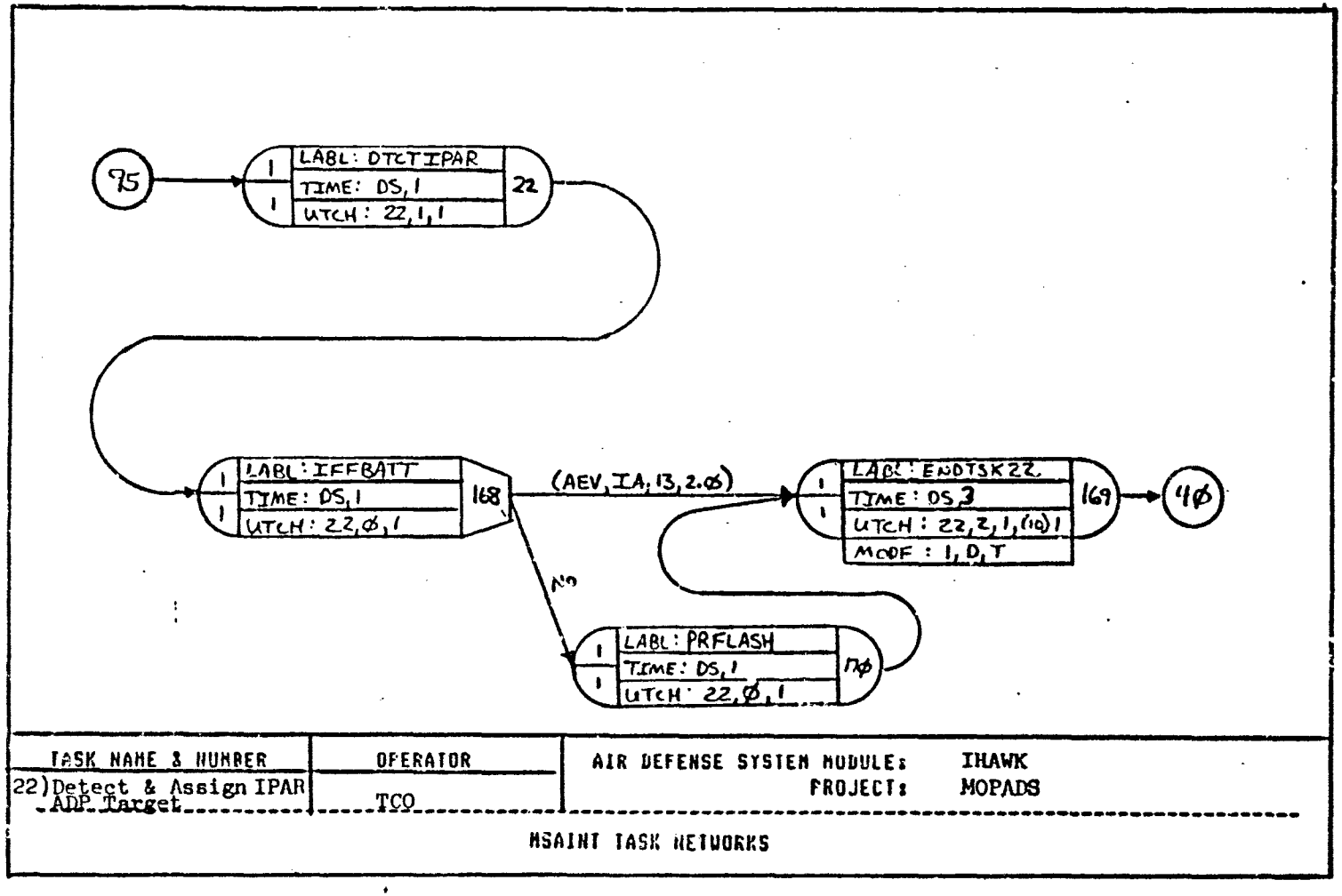


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
22) Detect & Assign IPAR ADP Targets	TCO	TM 9-1430-1526-12-1, Table 2-24

NAME: Riley Goodin
DATE: 11/4/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS


75
1
1
LABL: DTCTIPAR
TIME: DS,1
UTCH: 22,1,1
22
1
1
LABL: IFFBATT
TIME: DS,1
UTCH: 22,0,1
168
(AEV,IA,13,2.0)
1
1
LABL: ENDTSK22.
TIME: DS,3
UTCH: 22,2,1,(10)1
MODF: 1,D,T
169
40
NS
1
1
LABL: PRFLASH
TIME: DS,1
UTCH: 22,0,1
170
TASK NAME & NUMBER
OPERATOR
22) Detect & Assign IPAR ADP Target
TCO
AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS
MSAINT TASK NETWORKS

TASK NODE: 22

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	70.00000
SKILL-INDEX	4.000000
SKILL-WEIGHT	10.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
22	DTCTIPAR	22	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 168

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NON-SIM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	6.000000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
168	IFFBATT	22	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 169

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
169	ENDTSK??	22	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 170

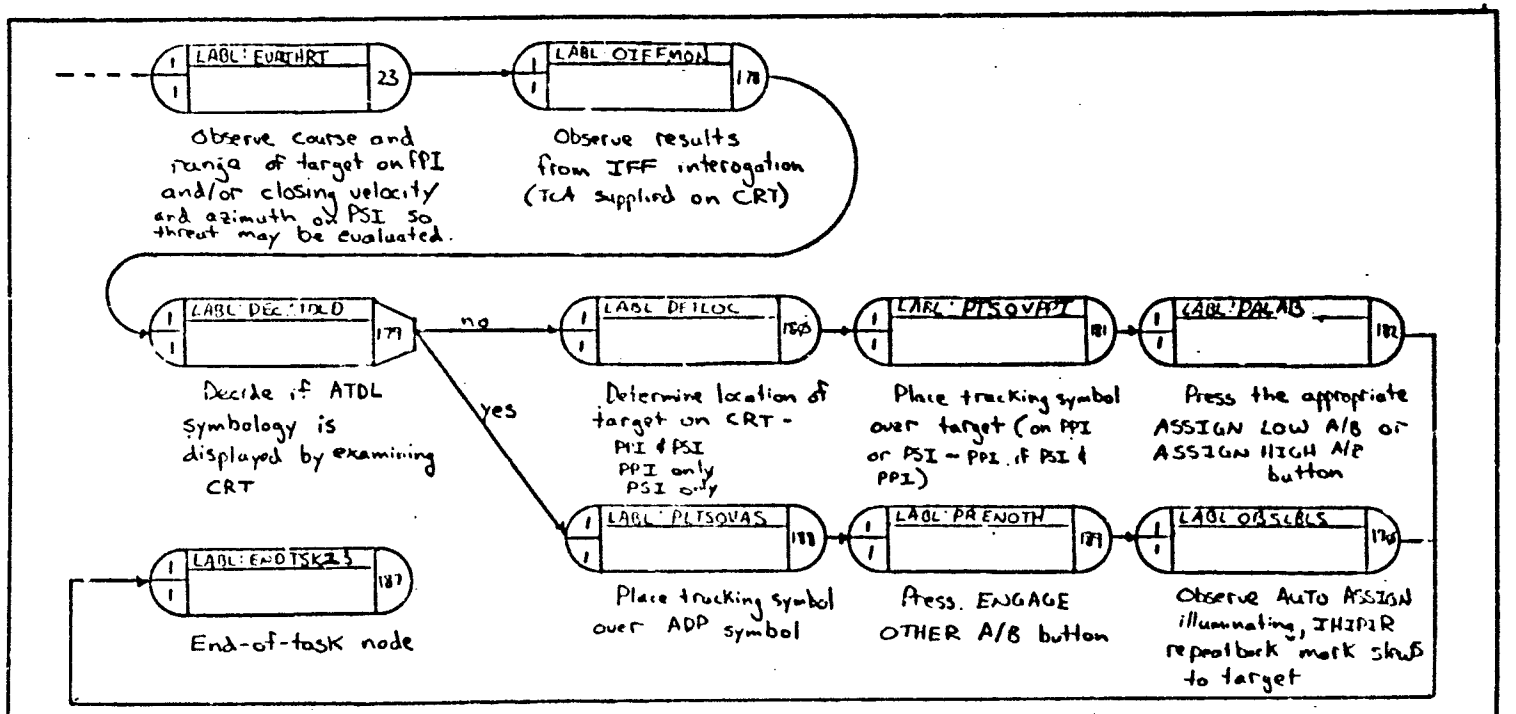
DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
170	PREFLASH	22	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

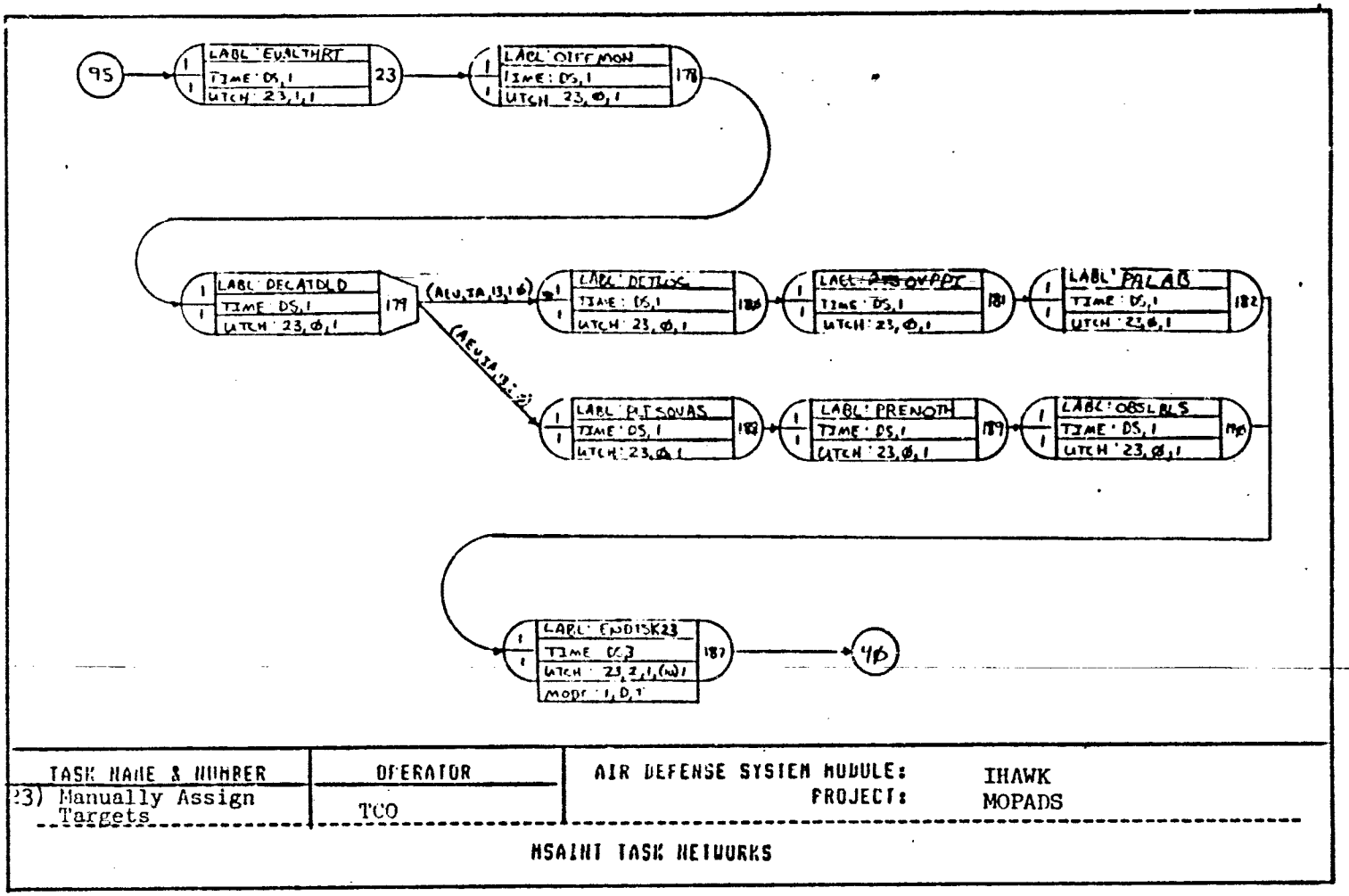


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
23) Manually Assign Targets	TCO	TM 9-1430-1526-12-1, Table 2-24

NAME: Riley Goodin
DATE: 11/7/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
23) Manually Assign Targets	TCO	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 23

DISTRIBUTION-TYPE	8.000000
MEAN	0.8330000E-01
STANDARD-DEVIATION	0.2920000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	18.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
23	EVALTHRT	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 178

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	20.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	80.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
178	DIFFMON	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: M-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 179

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-SIM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
179	DECATDLI	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 180

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	70.00000
SKILL-INDEX	18.00000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
180	DETLOC	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 181

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
181	PTSOVPPI	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 182

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
182	PALAB	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 187

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
187	ENDTSK23	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 188

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE 2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
188	PUTSOVAS	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 189

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STN-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
189	PRENOTH	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 190

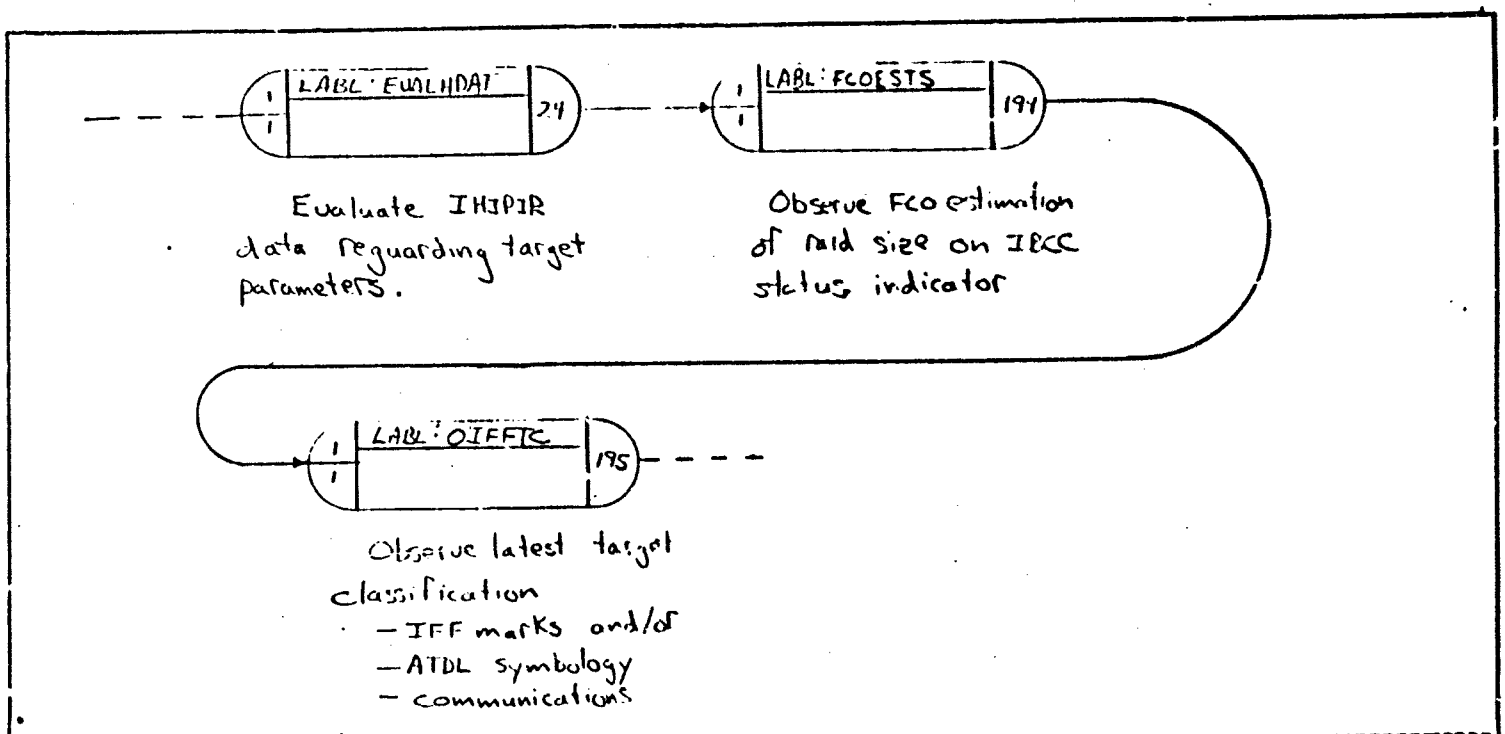
DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	80.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	HOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
190	OBSLBLS	23	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: HOPADS

TASK NODE SPECIFIC DATA

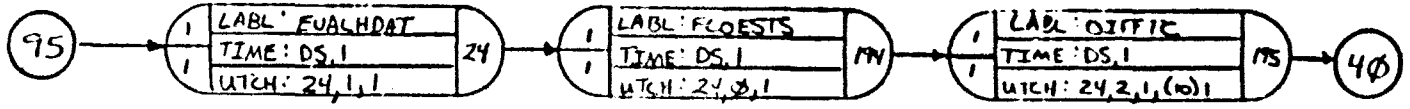


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
24) IHIPIR Tracking	TCO	TM 9-1430-1526-12-1, Table 2-24

NAME: Riley Goodin
DATE: 11/7/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
24) IHIPIR Tracking	TCO	

MSAINT TASK NETWORKS

TASK NODE: 24

DISTRIBUTION-TYPE	8.000000
MEAN	0.5830000E-01
STANDARD-DEVIATION	0.2040000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-S,M-ITEMS	1.000000
SKILL-INDEX	1.000000
SKILL-WEIGHT	20.00000
SKILL-INDEX	6.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	30.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
24	EVALHDAT	24	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 194

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
194	FCOESTS	24	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 195

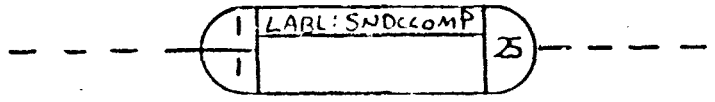
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	12.00000
SKILL-WEIGHT	60.00000
SKILL-INDEX	8.000000
SKILL-WEIGHT	40.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
195	OIFFTC	24	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



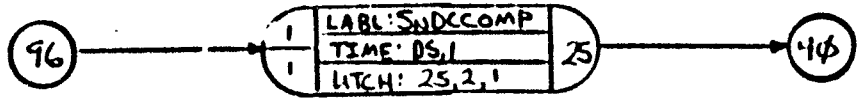
Push REJECT
button on TCO panel.

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
25) Send Cannot Comply Message to Q-73	TCO	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 12/26/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
25) Send Cannot Comply Message	TCO	PROJECT: MOPADS

NSAIMI TASK NETWORKS

TASK NODE: 25

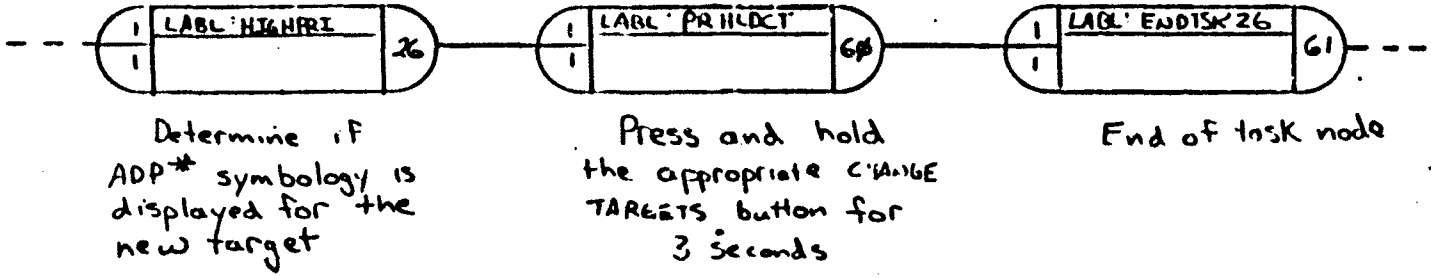
DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
25	SNDCCOMD	25	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



Determine if ADP* symbology is displayed for the new target

Press and hold the appropriate CHANGE TARGETS button for 3 seconds

End of task node

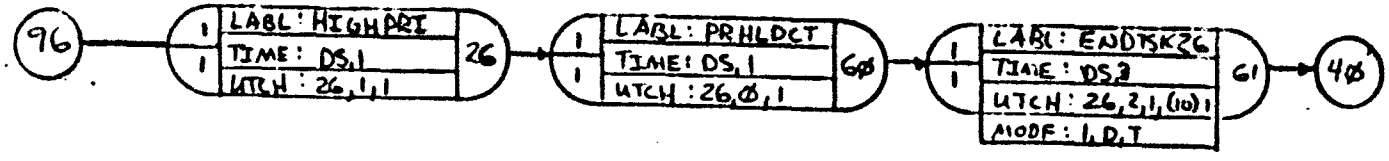
* Note: All CHANGE TARGETS processing will be done manually; however, node 26 and its associated time delay will remain with this task.

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
26) Higher Priority Target to be Assigned to Firing Section	TCO	TM 9-1430-1526-12-1, Table 2-24

NAME: Riley Goodin
DATE: 11/10/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE:	IHAWK
26)Higher Priority Target To Be...	TCO	PROJECT:	MOPADS

MSAINT TASK NETWORKS

TASK NODE: 26

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
26	HIGHPRI	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 60

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
60	PRHLDCT	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 61

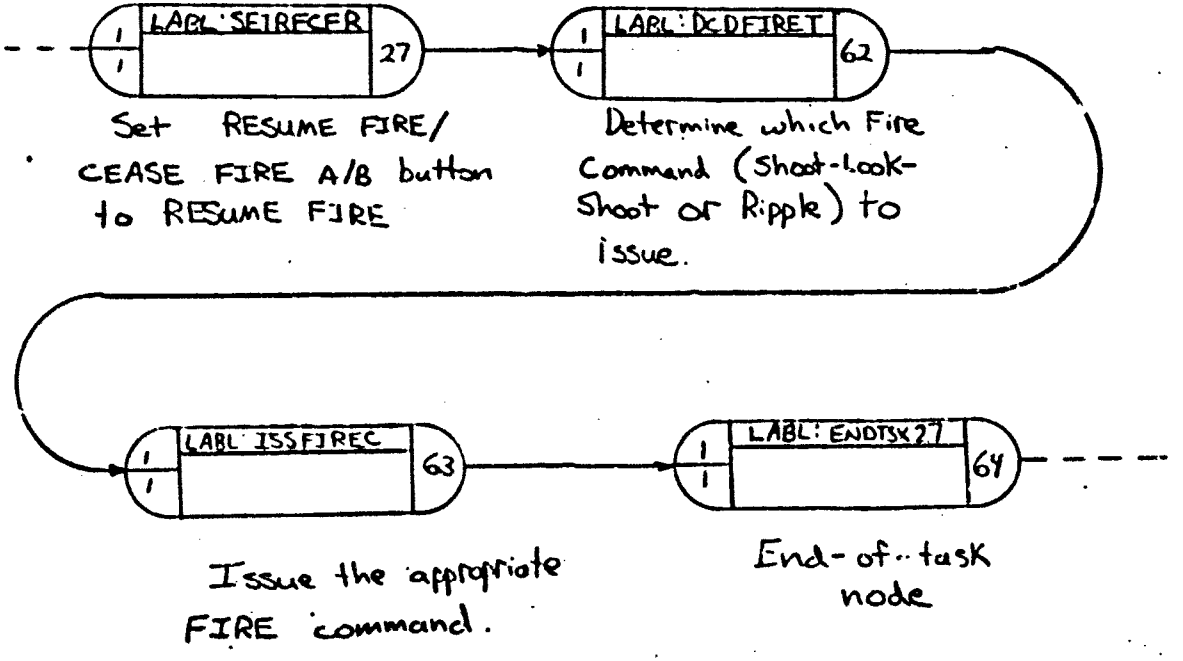
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
61	ENDTSK26	26	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

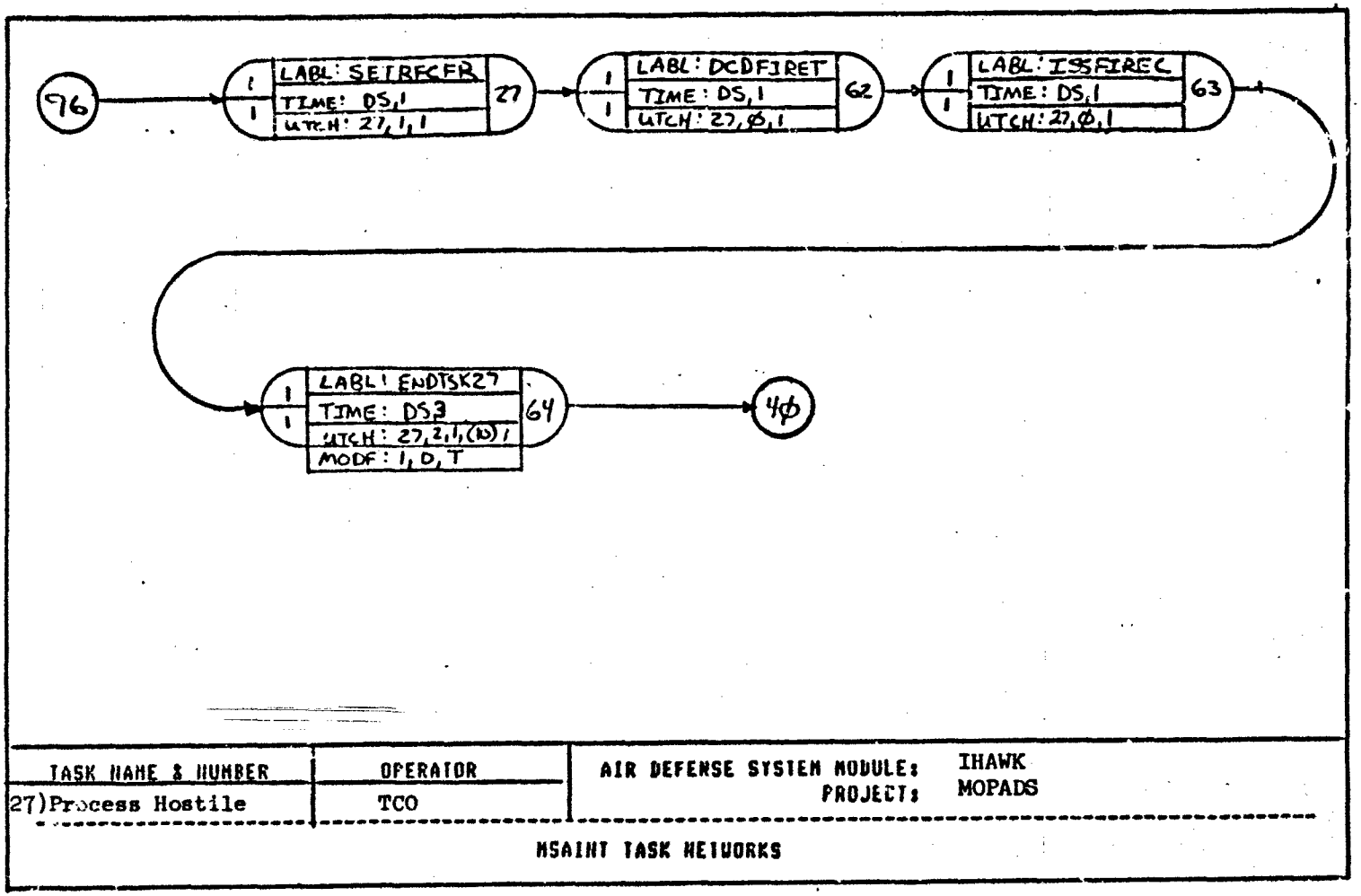


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
27) Process Hostile Target	TCO	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin
DATE: 11/8/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
27)Process Hostile	TCO	

MSAINT TASK NETWORKS

TASK NODE: 27

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
27	SETRFCFR	27	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 62

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	7.000000
SKILL-WEIGHT	60.00000
SKILL-INDEX	1.000000
SKILL-WEIGHT	40.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
62	DCDFIRET	27	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 63

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
63	ISSFIREC	27	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

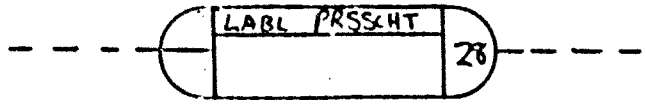
Zero Task Time - No Skills

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTUR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
64	ENDTSK64	27	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: M-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

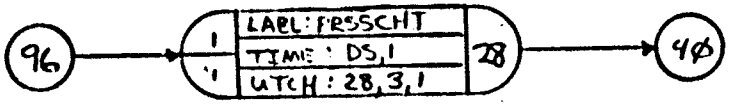


Press and hold CHANGE TARGETS button for 3 seconds

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
28) Assigned Target Determined Friendly	TCO	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin DATE: 11/10/83	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
28) Assigned Target Determined Friendly	TCO	

MSAINT TASK NETWORKS

TASK NODE: 28

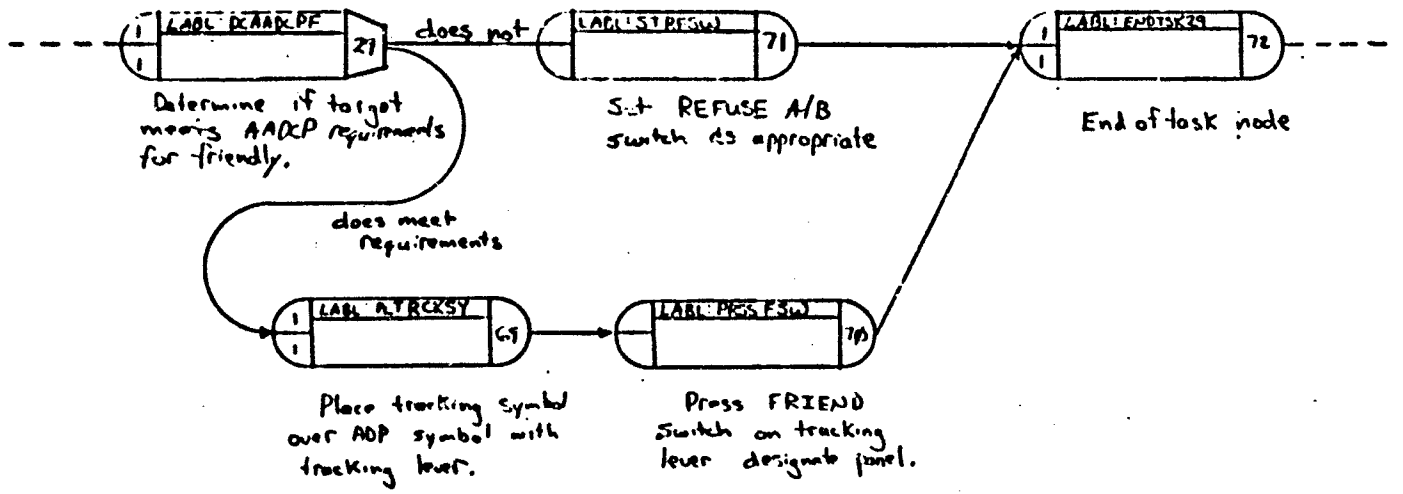
DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
28	PRSSCHT	28	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

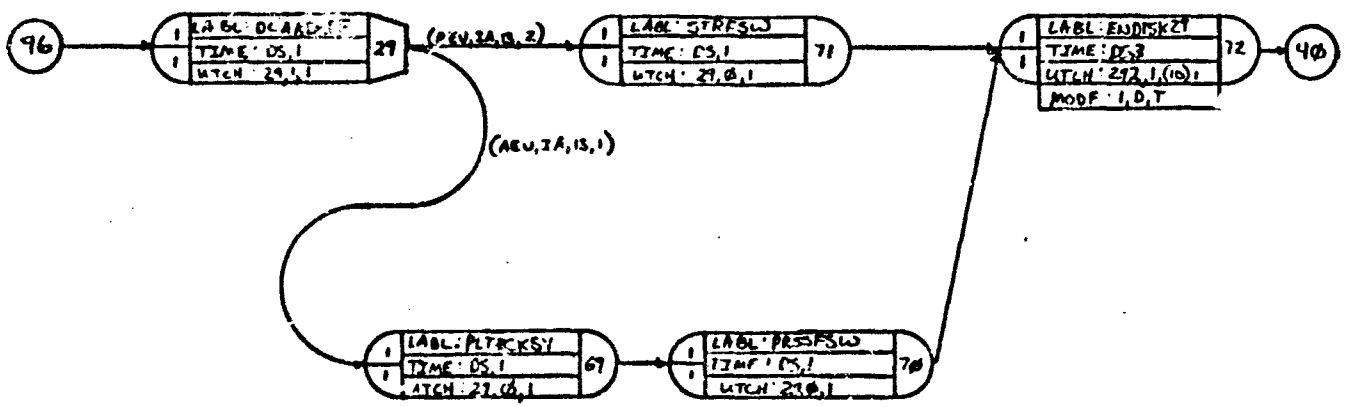


TASK NAME & DESCRIPTION:	OPERATOR.	REFERENCE:
29) Process Friend	TCO	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin
DATE: 11/8/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
(29) Process Friend	TCO	PROJECT: MOPADS

HSAINT TASK NETWORKS

TASK NODE: 29

DISTRIBUTION-TYPE	8.000000
MEAN	0.5000000E-01
STANDARD-DEVIATION	0.1750000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	3.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
29	DCAADCPF	29	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-1HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 69

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
69	PLTRCKSY	29	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 70

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-SIM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
70	PRSSFSW	29	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 71

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
71	STRFSW	29	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

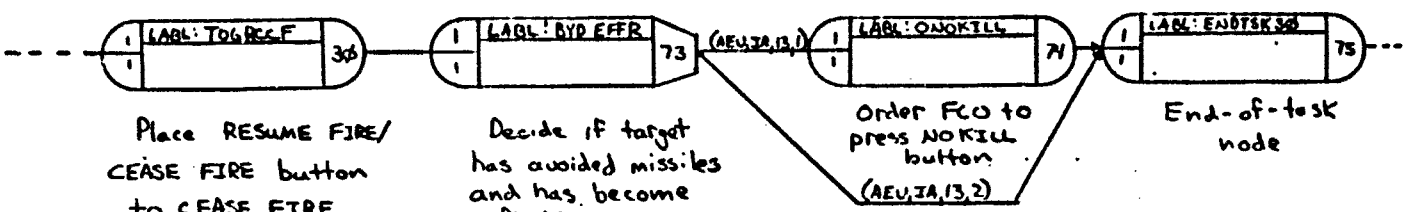
Zero time-no skills

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
72	ENDTSK2	29	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

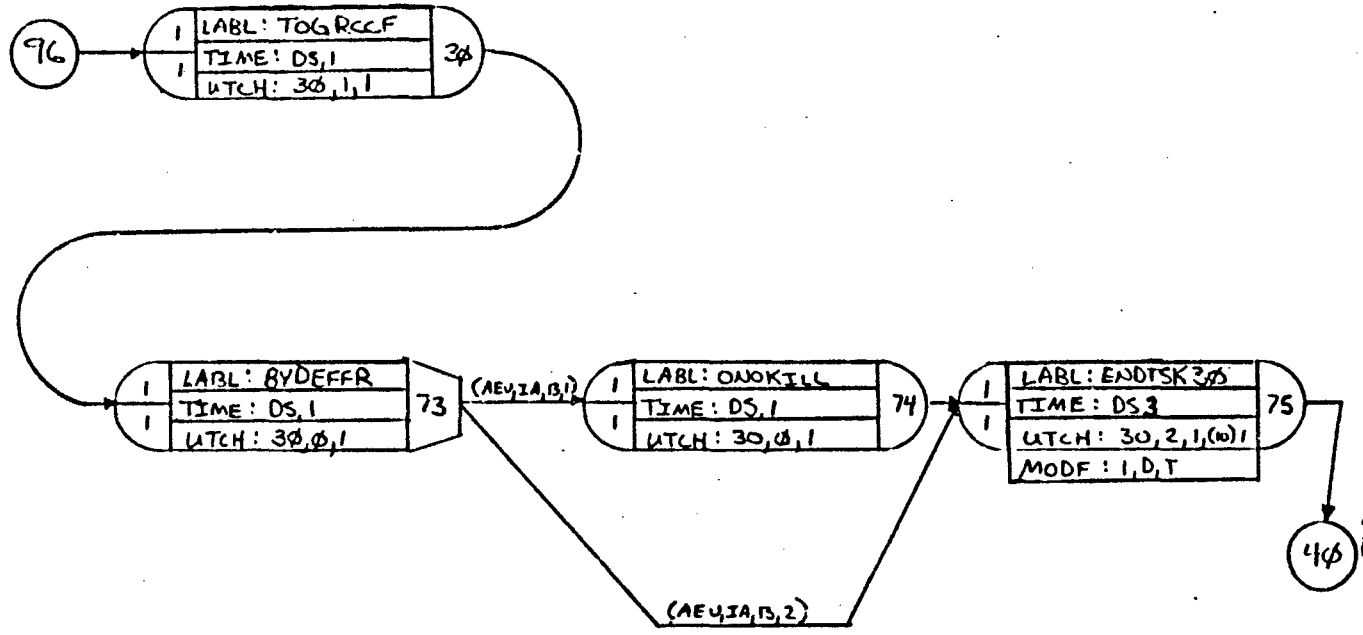
TASK NODE SPECIFIC DATA



TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
30) Evaluate Whetherh More Missiles Are to be Fired	TCO	TM 9-1430-1526-12-1,Table 2-24

NAME: Riley Goodin DATE: 11/9/83	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
30)Evaluate Whether More Missiles To Be Fired	TCO	

MSAINT TASK NETWORKS

TASK NODE: 30

DISTRIBUTION-TYPE	8.000000
MEAN	0.8300000E-02
STANDARD-DEVIATION	0.2900000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
30	TOGRCCF	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 73

DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	0.0000000E+00
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	3.000000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
73	BYDEFFR	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 74

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000009
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
74	ONOKILL	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: M-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 75

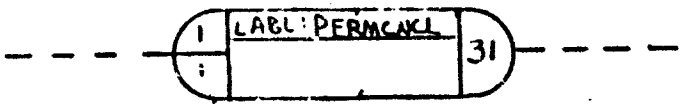
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	3.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
75	ENDTSK30	30	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



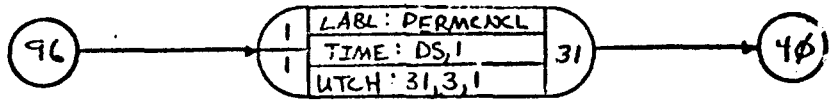
Give ASO verbal permission to cancel alert

TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
31) Give ASO Permission to Cancel Alert	TCO	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 11/4/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
31) Give ASO Permission to Cancel Alert	TCO	

MSAINT TASK NETWORKS

TASK NODE: 31

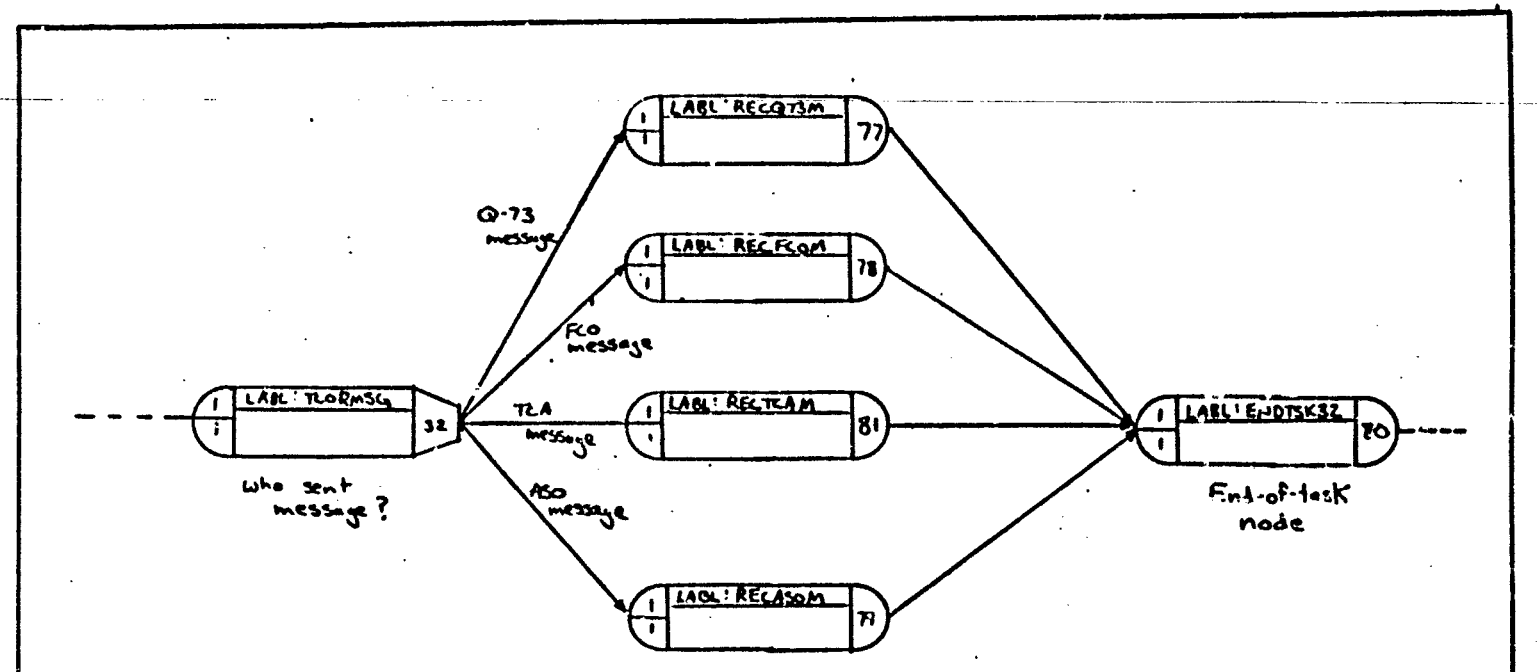
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
31	PERMCNCL	31	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

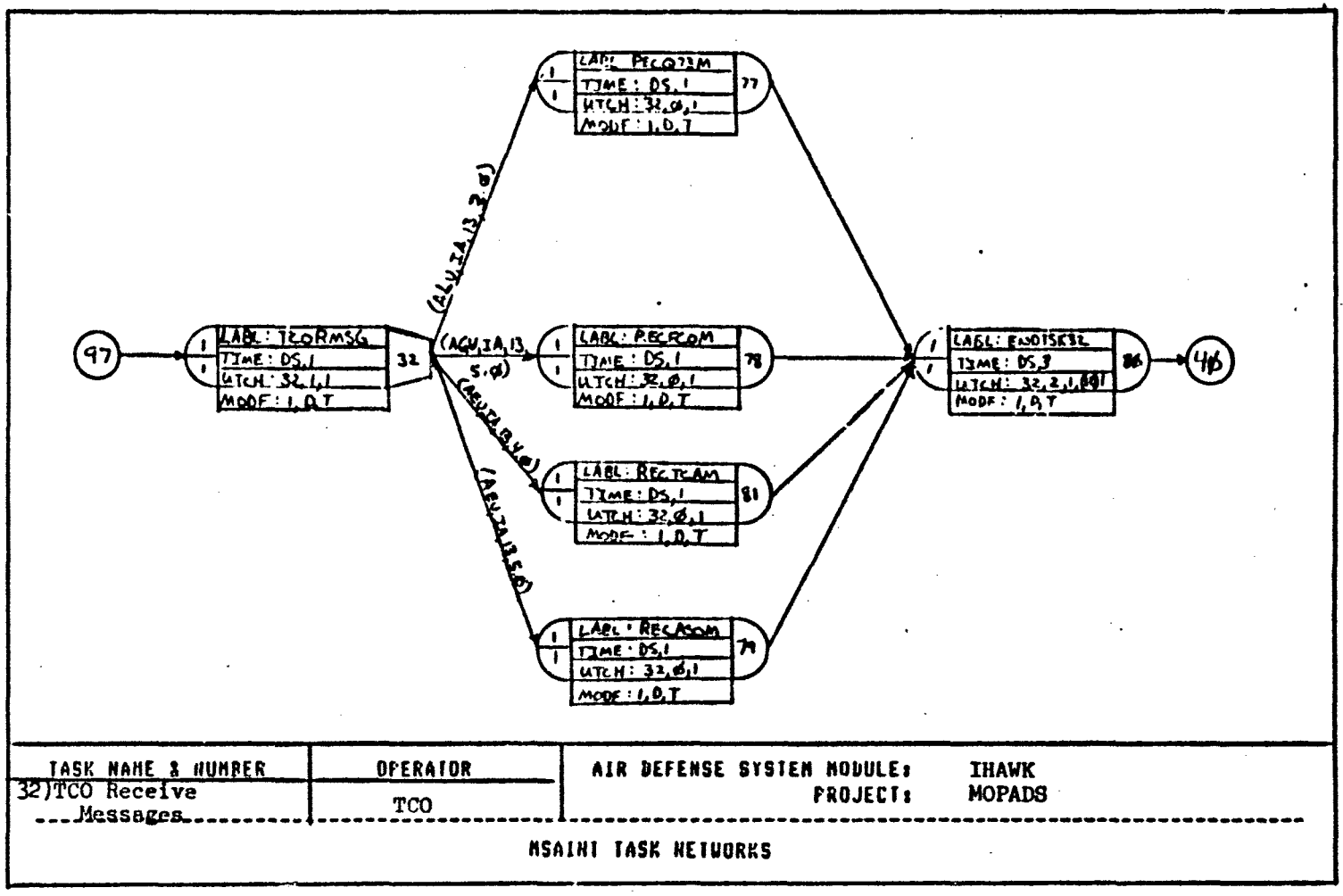


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
32) TCO Receive Message	TCO	TM 9-1430-1526-12-1, Table 2-24, 25, 26

NAME: Riley Goodin
DATE: 11/8/83

AIR DEFENSE SYSTEM MODULE: I!HAWK
PROJECT: MOPADS

NSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
32)TCO Receive Messages	TCO	

MSAINT TASK NETWORKS

TASK NODE: 32

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
32	TCORMSG	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 77

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
77	RECQ73M	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 78

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
78	RECFCOM	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 79

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
79	RECASOM	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 80

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
80	ENDTSK32	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 81

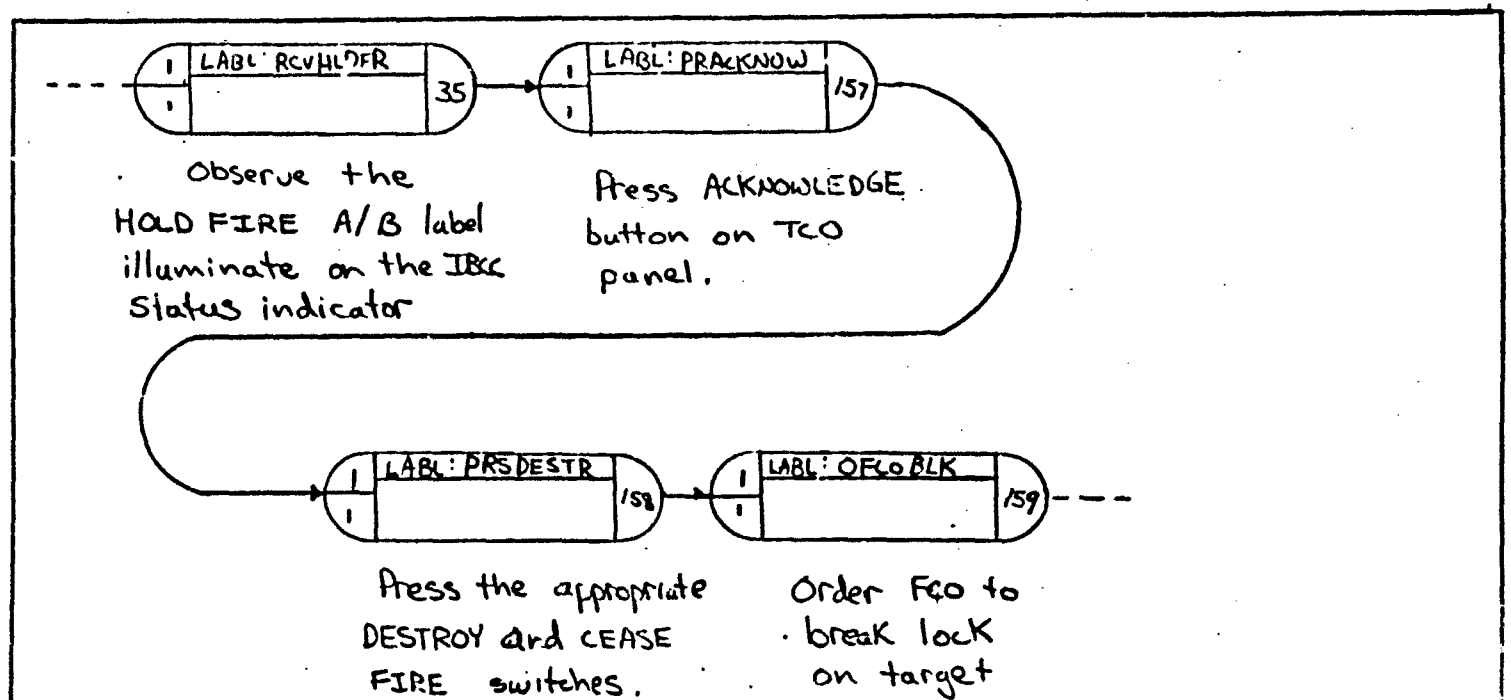
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
81	RECTCAM	32	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

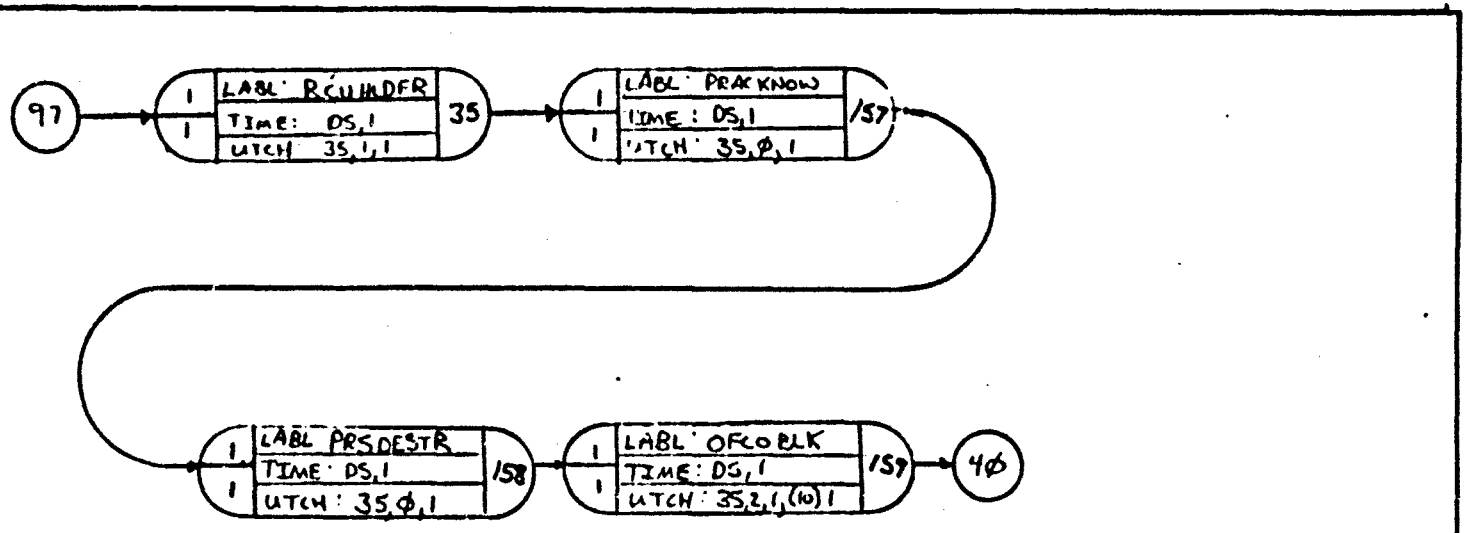


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
35) Process "HOLD FIRE" Command	TCO	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 11/9/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
35) Process "HOLD FIRE" Command	TCO	PROJECT: MOPADS

NSAINT TASK NETWORKS

TASK NODE: 35

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
35	RCVHLDFR	35	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 157

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
157	PRACKNOW	35	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 158

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
158	PRSDESTR	35	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 159

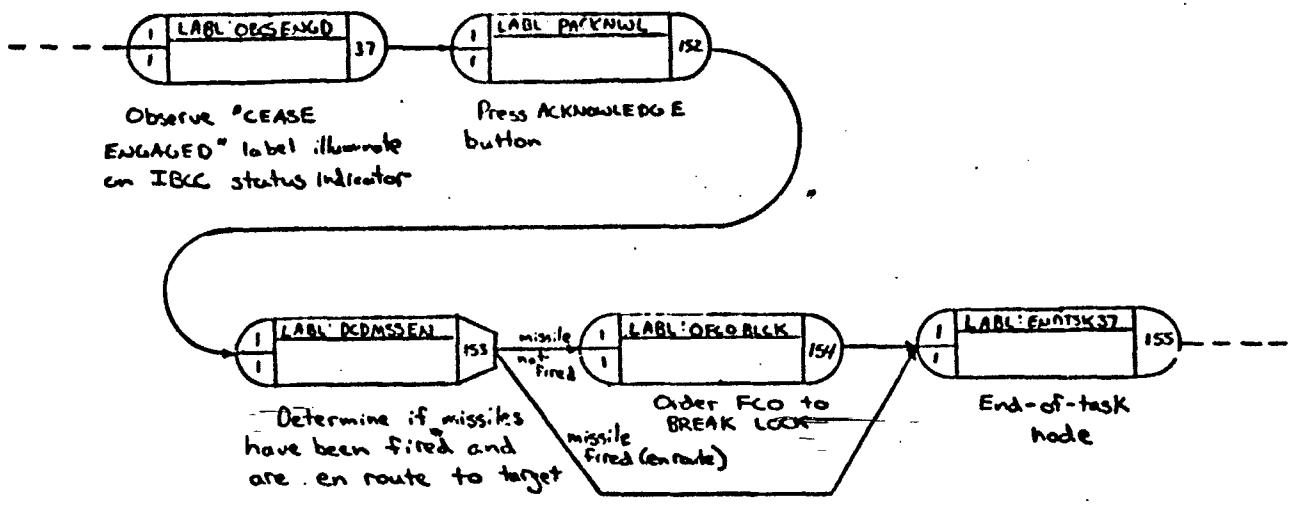
DISTRIBUTION-TYPE	8.000000
MEAN	0.3300000E-01
STANDARD-DEVIATION	0.1167000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
159	OFCOBLK	35	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-HAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

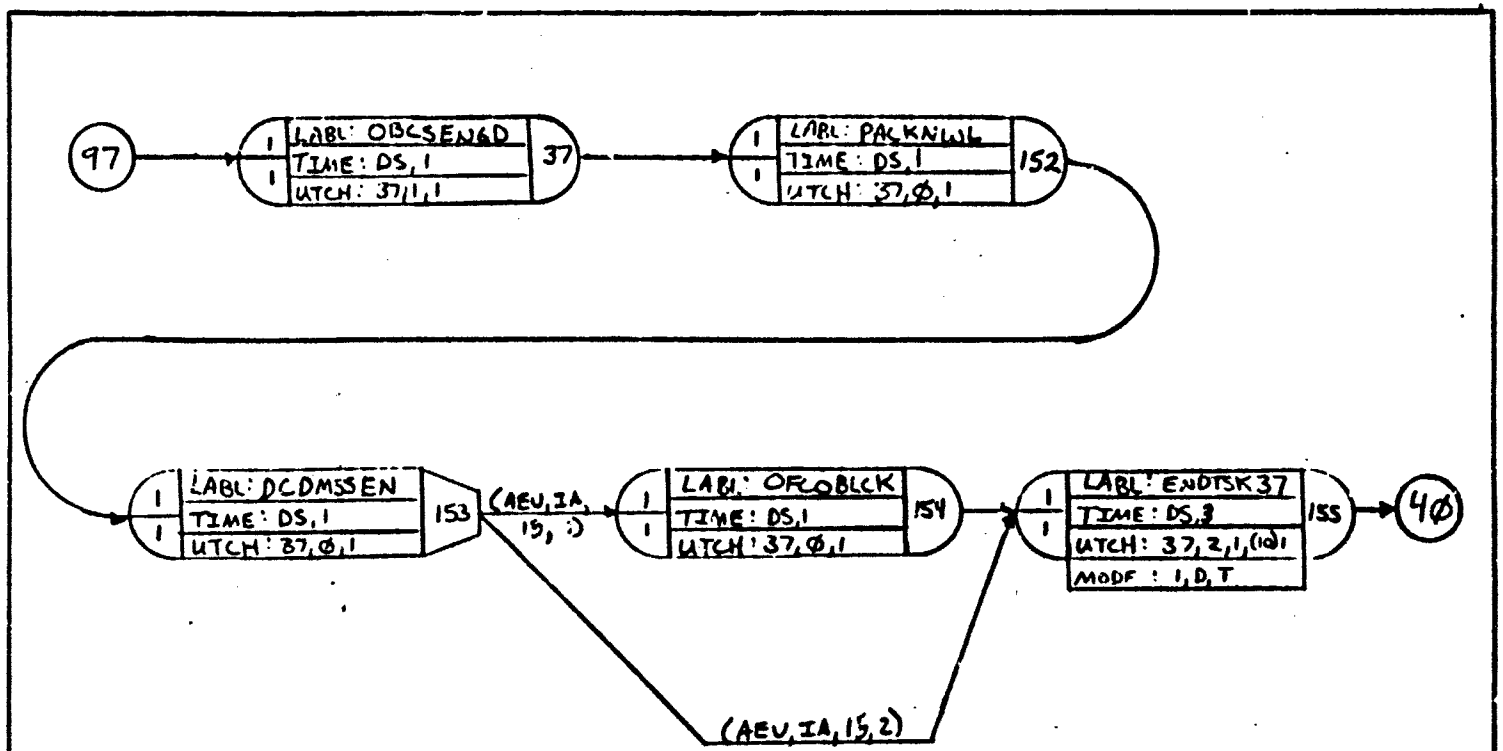


TASK NAME & DESCRIPTION:	OPERATOR:	REFERENCE:
37) Process "CEASE ENGAGED" Command	TCO	TM 9-1430-1526-12-1

NAME: Riley Goodin
DATE: 11/9/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

MSAINT TASK MODELS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
37) Process "CEASE ENGAGED" Command	TCO	

HSAINT TASK NETWORKS

TASK NODE: 153

DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	3.000000
SKILL-WEIGHT	50.00000
SKILL-INDEX	7.000000
SKILL-WEIGHT	50.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
153	DCDMSSEN	37	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 154

DISTRIBUTION-TYPE	8.000000
MEAN	0.4170000E-01
STANDARD-DEVIATION	0.1460000E-01
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	10.00000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	19.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
154	OFCOBLCK	37	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 155

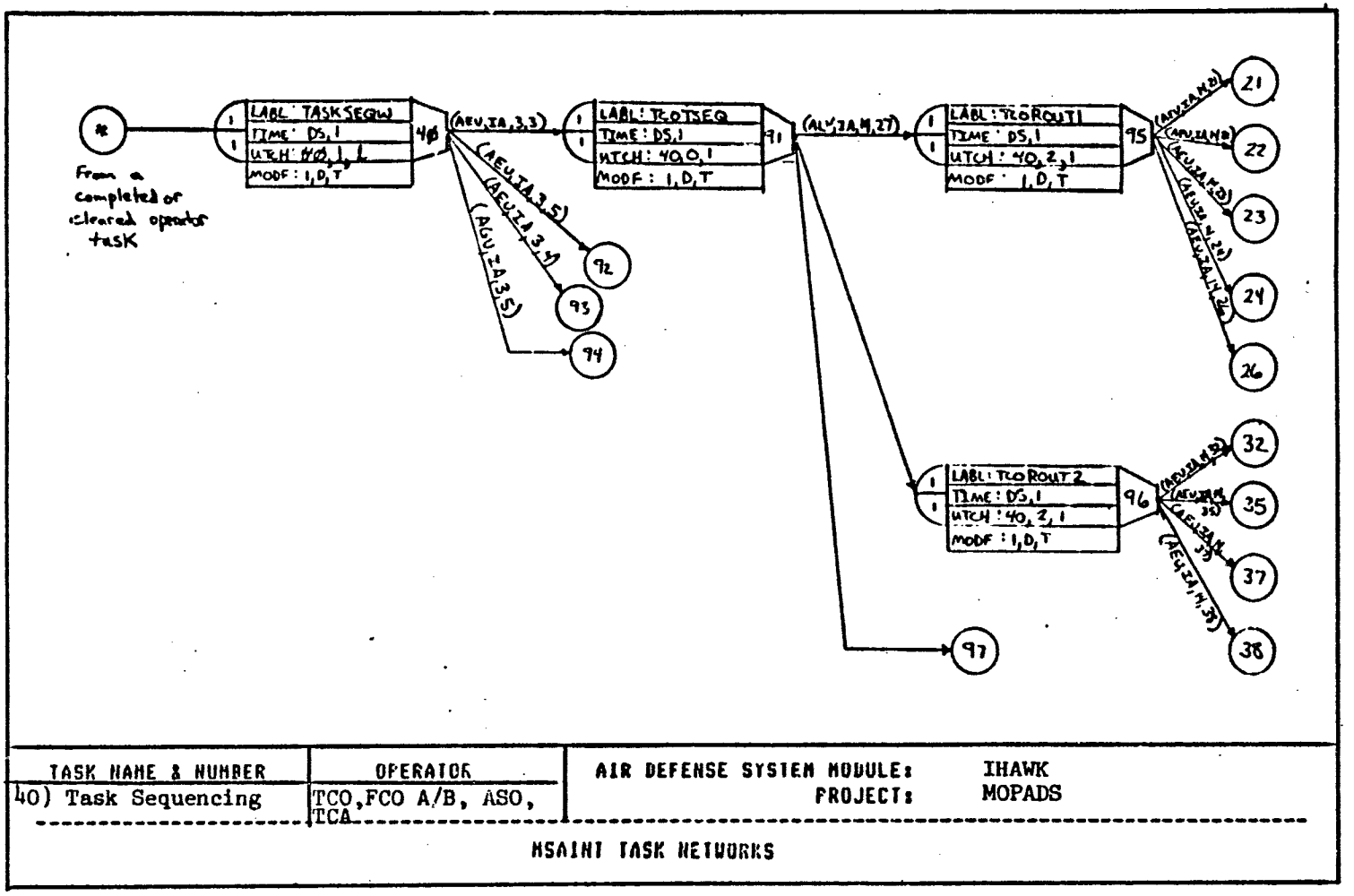
DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
155	ENDTSK37	37	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

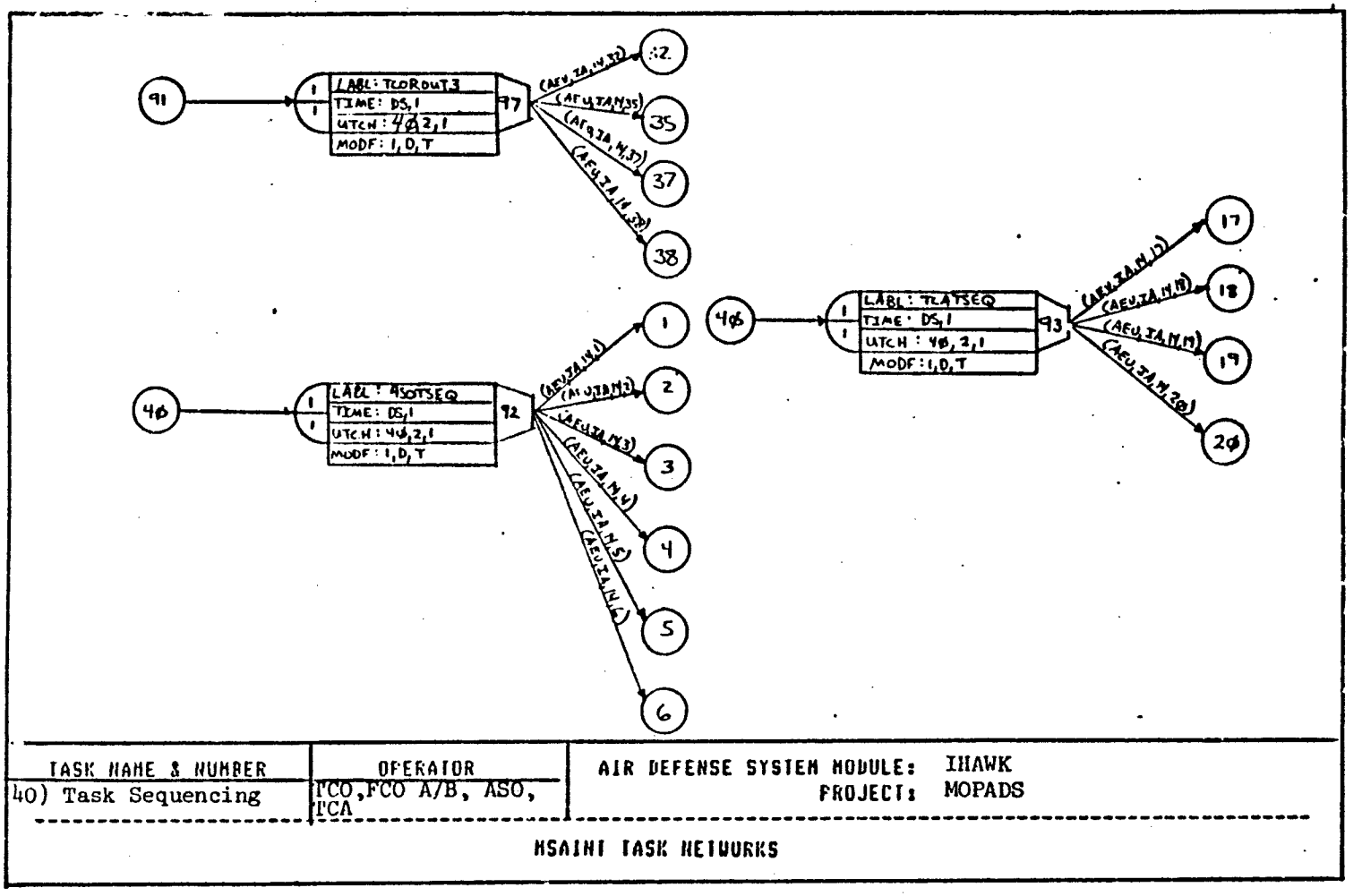
AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA



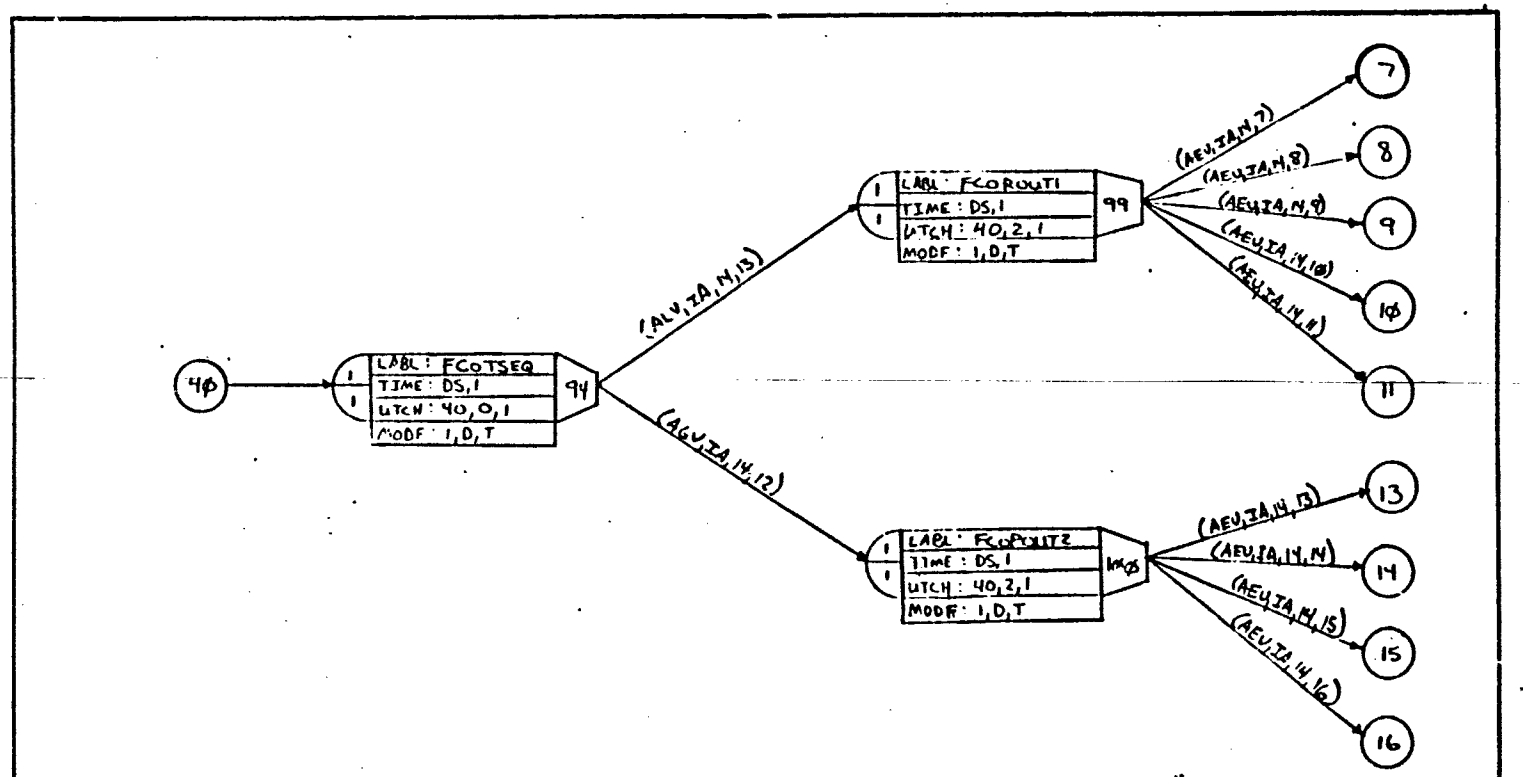
TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE:	IHAWK
40) Task Sequencing	TCO,FCO A/B, ASO, TCA	PROJECT:	MOPADS

MSAINT TASK NETWORKS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK PROJECT: MOPADS
40) Task Sequencing	FCO,FCO A/B, ASO, TCA	

HSAIHI TASK NETWORKS



TASK NAME & NUMBER	OPERATOR	AIR DEFENSE SYSTEM MODULE: IHAWK
40) Task Sequencing	TCO,FCO A/B, ASO, TCA	PROJECT: MOPADS

MSAINT TASK NETWORKS

TASK NODE: 40

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
40	TASKSEQW	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 91

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
91	TCOTSEQ	91	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1982

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 92

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
92	ASOTSEQ	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 93

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
93	TCATSEQ	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Page 1 of 1

TASK NODE: 94

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
94	FCOTSEQ	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 95

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
95	TCOROUT1	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 96

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
96	TCOROUT2	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 97

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
97	TCOROUT3	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 99

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
99	FCOROUT1	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 100

DISTRIBUTION-TYPE	1.000000
MEAN	0.0000000E+00
STANDARD-DEVIATION	0.0000000E+00
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
100	FCCROUT2	40	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA Page 1 of 1

Table II-2

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
IHAWK SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER	MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
1	ASOSTNB	1	41		
2	DTCTCWAR	2	42		
3	ESTTARPR	3	43		
4	REQCLRAL	4	44		
5	POSCWCRS	5	45	SOURCTCO	-
6	OBSVALEX	6	46	SOURCTCA	-
7	FCOSTNDB	7	47	SOURCASO	-
8	DROCADP	8	48	SOURCFCA	-
9	POSCURS	9	49	SOURCFCB	-
10	PSHFSOFF	10	50		
11	MONTDOPA	11	51		
12	OBSVLNCH	12	52		
13	DECFCTYP	13	53		
14	WAITINTP	14	54		
15	PSHFSACT	15	55		
16	PRCHTRGT	16	56		
17	TCASTNBD	17	57		
18	PCCHANDW	18	58		
19	SLTIFFM	19	59		
20	DCDATDLS	20	60	PRHLDCT	26
21	TCOSTNDB	21	61	ENDTSK26	26
22	DTCTIPAR	22	62	DCDFIRET	27
23	EVALTHRT	23	63	ISSFIRFC	27
24	EVALHDAT	24	64	ENDTSK27	27
25	SNDCCOMP	25	65		
26	HIGHPRI	26	66		
27	SETRFCFR	27	67		
28	PRSSCHT	28	68		
29	DCAADCPF	29	69	PLTRCKSY	29
30	TOGRCCF	30	70	PRSSFSW	29
31	PERMCNCL	31	71	STRFSW	29
32	TCORMSG	32	72	ENDTSK29	29
33			73	BYDEFFR	34
34			74	ONOKILL	30
35	RCVHLDFR	35	75	ENDTSK30	30
36			76		
37	OBCSENDGD	37	77	RECQ73M	32
38			78	RECFCOM	32
39			79	RECASOM	32
40	TASKSEQW	40	80	ENDTSK32	32

Table II-2 (continued)

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
IHAWK SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER	MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
81	RECTCAM	32	121	PRSFLSHE	9
82			122	DCDHIPLK	9
83			123	TARGALT	9
84			124	ENDTSK09	9
85			125		
86			126		
87			127		
88	ENDTSK10	10	128		
89	WAITRELD	10	129		
90	DEADNODE	-	130		
91	TCOTSEQ	40	131		
92	ASOTSEQ	40	132		
93	TCATSEQ	40	133	DCDRDSIZ	11
94	FCOTSEQ	40	134	PSHOFM	11
95	TCOROUT1	40	135		
96	TCOROUT2	40	136	LCHRDLAY	13
97	TCOROUT3	40	137	ENDTSK11	11
98			138	ENDTS12	12
99	FCOROUT1	40	139	PSHFBRP	13
100	FCOROUT2	40	140	WAITRPFR	13
101	WAITMSGK	14	141		
102			142	ENDTSK13	13
103			143	PSHFBSLS	13
104			144	AUDTNBST	14
105			145	DROPDOP	14
106	LOWPRCOF	3	146	GVERBNK	14
107	WAITTCOM	4	147	PSHNOKLL	14
108	PRSALCNL	4	148	PRSKILLB	14
109	PSWSLEW	5	149	PRSBRKLK	14
110	PRSTCCAB	5	150	SHOOTEM	14
111	MRKTARGT	6	151		
112	OBSRCHLM	8	152	PACKNWL	37
113	DHIPIRLK	8	153	DCDMSSEN	37
114			154	OFCOBLCK	37
115	OBVLOCKL	8	155	ENDTSK37	37
116			156		
117	ENDTSK08	8	157	PRACKNOW	35
118			158	PRSDESTR	35
119			159	OFCOBLK	35
120	OBASSLTP	8	160	ENDTSK14	14

TASK NODE: 37

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	8.000000
SKILL-WEIGHT	80.00000
SKILL-INDEX	12.00000
SKILL-WEIGHT	20.00000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
37	OBCSENGD	37	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

TASK NODE: 152

DISTRIBUTION-TYPE	8.000000
MEAN	0.1670000E-01
STANDARD-DEVIATION	0.5830000E-02
KILOCALORIES/MIN	1.000000
NUMBER-OF-BRANCHES-OUT	1.000000
STIMULUS-MODE-1	10.00000
STIMULUS-MODE-2	0.0000000E+00
RESPONSE-MODE	1.000000
OBSRVR-TARGET-POSITION	5.000000
CONTROL-DISTANCE	1.000000
CONTROL-WIDTH	0.1000000
NUMBER-OF-DISPLAYS	1.000000
NUMBER-OF-ALTERNATIVES	1.000000
NUM-STM-ITEMS	1.000000
SKILL-INDEX	13.00000
SKILL-WEIGHT	100.0000

TASK NODE NUMBER	TASK LABEL	MOPADS TASK NUMBER	TASK TIME INFORMATION			TASK ELEMENT ERROR FACTOR
			MEAN	STD.DEV.	DISTRIBUTION TYPE	
152	PACKNWL	37	(above)	(above)	(above)	1.0

NAME: J. Riley Goodin II
DATE: 21 December 1983

AIR DEFENSE SYSTEM MODULE: W-IHAWK
PROJECT: MOPADS

TASK NODE SPECIFIC DATA

Table II-2 (continued)

CROSS REFERENCE OF NODE NUMBERS TO OPERATOR TASK NUMBERS
FOR
IHAWK SYSTEM MODULE

MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER	MSAINT NODE NUMBER	TASK LABEL	OPERATOR TASK NUMBER
161	PSHBREAK	16			
162	PRCWCONF	18			
163	CORLTGT	18			
164	WAITSWEP	19			
165	MONIFFN	19			
166	MRKRFLCT	20			
167	ENDTSK20	20			
168	IFFBATT	22			
169	PRFLASH	22			
170	ENDTSK22	22			
171					
172					
173					
174					
175					
176					
177					
178	OIFFMON	23			
179	DECATDLD	23			
180	DETLOC	23			
181	PTSOVPPI	23			
182	PALAB	23			
183					
184					
185					
186					
187	ENDTSK23	23			
188	PLTSOVAS	23			
189	PRENOTH	23			
190	OBVLBLS	23			
191					
192					
193					
194	FCOESTS	24			
195	OIFFTC	24			
196					
197					
198					
199					
200					

5-0 USER FUNCTIONS

This section contains the forms used to describe the user functions called in the simulation.

USER FUNCTION NUMBER	TASK NUMBER(S) WHERE CALLED (USERF ONLY)	DESCRIPTION
1	14	Used to call FLTY, the subroutine to calculate the intercept time for missiles.
2	113, 122	Used to determine the delay time between attempting and achieving lock

WHICH USER FUNCTION? USERIN: USERF: X

NAME: Riley Goodin
DATE: 10/27/83

AIR DEFENSE SYSTEM MODULE: IHAWK
PROJECT: MOPADS

SAINT USER FUNCTIONS Page 1 of 1

6-0 MODERATOR FUNCTION

There is only one moderator function (referred to as Moderator Function #1) used by the IHAWK. This moderator function allows the user to gain access to the human factors codes for moderating task performance time.

Any task node that has moderator function one active must have skills and distribution data stored in the data base, and the task time on the Task data card must be specified as DS,1.

7-0 MESSAGES SENT FROM IHAWK SYSTEM MODULE

The following forms describe each of the messages sent from operators in the IHAWK system module to operators within the IHAWK and to operators in Battalion AN/TSQ-73.

MOPADS MESSAGE DESCRIPTION

Message No. 14	MESSAGE IB ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	--
2	Operator Type	1
3	Functional Type	4
4	Message Subtype	4
5	Message Priority	3

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	--
4	ATDL Code (Unused)	--
5	* Time Message Sent	--
6	* Message Number	--
7	* Sender CRN	--
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	--

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	Message Number Acknowledging

MESSAGE SUBTYPE DESCRIPTION

Acknowledge Message - "Will Comply"

Name: Riley Goodin Date: 12/20/83	System Module: IHAWK Project: MOPADS

From	To	From	To
IHAWK(35)	BN(22)		
IHAWK(37)	BN(22)		

MOPADS MESSAGE DESCRIPTION

Message No. 15 — MESSAGE ID ELEMENT — Page 1 of 1

Element		Description	Value
1	*	Receiver CRN	--
2		Operator Type	1
3		Functional Type	4
4		Message Subtype	5
5		Message Priority	4

MESSAGE DATA LINK ELEMENT

Element		Description	Value
1		Communication Network 1-Voice 2-ATDL	2
2		Acknouledgement Required 1 - Yes 2 - No	2
3		Unused	--
4		ATDL Code (Unused)	--
5	*	Time Message Sent	--
6	*	Message Number	--
7	*	Sender CRN	--
8		Sender Operator Type	3
9		Sender Systen Module Type	4
10		Task Node Number Sent From	--

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK column number of track that fire unit is unable to get a lock on

MESSAGE SUBTYPE DESCRIPTION

Acknowledge Message - "Can't Comply"

Name: Riley Goodin	System Module: IHAWK
Date: 12/20/83	Project: MOPADS

From	To	From	To
IHAWK(25)	BN(26)		

MOPADS MESSAGE DESCRIPTION

Message No. 16 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	--
2	Operator Type	1
3	Functional Type	3
4	Message Subtype	9
5	Message Priority	3

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	--
4	ATDL Code (Unused)	--
5	* Time Message Sent	--
6	* Message Number	--
7	* Sender CRN	--
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Mode Number Sent From	--

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 2
2	copy row number of fire section
3	FS A = 1 FS B = 2

MESSAGE SUBTYPE DESCRIPTION

FU back in action after missile reloading

Name: Riley Goodin
Date: 12/20/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
FCO(15)	BN(4)		

MOPADS MESSAGE DESCRIPTION

Message No. 17	MESSAGE IB ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	--
2	Operator Type	1,2,3,4,5,6,6,7,8,9
3	Functional Type	3
4	Message Subtype	1
5	Message Priority	-20

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	--
4	ATDL Code (Unused)	--
5	* Time Message Sent	--
6	* Message Number	--
7	* Sender CRN	--
8	Sender Operator Type	Aircraft
9	Sender System Module Type	1,4
10	Task Node Number Sent From	--

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 0

MESSAGE SUBTYPE DESCRIPTION

You are dead message - sent from CONTROL SYSTEM MODULE to all operators in the destroyed AD system module

Name: Riley Goodin Date: 12/20/83		System Module: CNTRL Project: MOPADS	
From	**To**	**From**	**To**
CNTRL	IHAWK	CNTRL	GROUP
CNTROL	BN		

MOPADS MESSAGE DESCRIPTION

Message No. 18	MESSAGE ID ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	-
2	Operator Type	1
3	Functional Type	3
4	Message Subtype	2
5	Message Priority	3

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 0

MESSAGE SUBTYPE DESCRIPTION

FU expended all hot missiles. This alert is followed by an inactive period during which reloading may occur.

Name:	Riley Goodin	System Module:	IHAWK
Date:	12/20/83	Project:	MOPADS

From	To	From	To
IHAWK(10)	BN(22)		

MOPADS MESSAGE DESCRIPTION

Message No. 19	MESSAGE ID ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	-
2	Operator Type	1
3	Functional Type	3
4	Message Subtype	3
5	Message Priority	+2

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 2
2	NTRACK column number of target
3	= 1 Kill = 2 No Kill

MESSAGE SUBTYPE DESCRIPTION

FU effective-tells the Q-73 whether the FU successfully engaged the indicated target

Name: Riley Goodin Date: 12/20/83	System Module: IHAWK Project: MOPADS

From	To	From	To
HAWK(14)	EN(4)		

MOPADS MESSAGE DESCRIPTION

Message No. 20 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	1
3	Functional Type	3
4	Message Subtype	4
5	Message Priority	+5

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK column number

MESSAGE SUBTYPE DESCRIPTION

FU self-initiated an Engagement (Alert 93)-Notifies the Q-73 that the FU is engaging a pop-up target.

Name: Jack Walker
Date: 8/4/73

System Module: IHAWK
Project: MOPADS

From	To	From	To
HAWK(38)	BN(22)		

MOPADS MESSAGE DESCRIPTION

Message No. 21	MESSAGE ID ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	-
2	Operator Type	3
3	Functional Type	2
4	Message Subtype	1
5	Message Priority	-5.5

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	1
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	5
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK Column Number (Track to be pre-empted)

MESSAGE SUBTYPE DESCRIPTION

Request permission to cancel TCC alert-ASO has found a higher priority target. (Coupled with Message 22)

Name: Riley Goodin	System Module: IHAWK
Date: 8/4/83	Project: MOPADS

From	To	From	To
ASO(4)	TCO(32)		

MOPADS MESSAGE DESCRIPTION

Message No. 22 — MESSAGE ID ELEMENT — Page 1 of 1

Element		Description	Value
1	*	Receiver CRN	-
2		Operator Type	5
3		Functional Type	1
4		Message Subtype	11
5		Message Priority	1

MESSAGE DATA LINK ELEMENT

Element		Description	Value
1		Communication Network 1-Voice 2-ATDL	1
2		Acknowledgement Required 1 - Yes 2 - No	2
3		Unused	-
4		ATDL Code (Unused)	-
5	*	Time Message Sent	-
6	*	Message Number	-
7	*	Sender CRN	-
8		Sender Operator Type	3
9		Sender System Module Type	4
10		Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK Column number

MESSAGE SUBTYPE DESCRIPTION

Cancel TCC alert order - TCO ok's the ASO forgetting about a pre-empted track (coupled with Message 21)

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
TCO(31)	ASO(4)		

MOPADS MESSAGE DESCRIPTION

Message No. 23 — MESSAGE ID ELEMENT — Page 1 of 1

Element		Description	Value
1	*	Receiver CRN	-
2		Operator Type	4
3		Functional Type	3
4		Message Subtype	5
5		Message Priority	-5

MESSAGE DATA LINK ELEMENT

Element		Description	Value
1		Communication Network 1-Voice 2-ATDL	1
2		Acknowledgement Required 1 - Yes 2 - No	1
3		Unused	-
4		ATDL Code (Unused)	-
5	*	Time Message Sent	-
6	*	Message Number	-
7	*	Sender CRN	-
8		Sender Operator Type	5
9		Sender System Module Type	4
10		Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK Column Number for New Track

MESSAGE SUBTYPE DESCRIPTION

Alert the TCA of a new ASO target (coupled with Message 24)

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
ASO(5)	TCA(18)		

MOPADS MESSAGE DESCRIPTION

Message No. 24 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	5
3	Functional Type	4
4	Message Subtype	6
5	Message Priority	+3

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	1
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	4
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 2
2	Message Number Acknowledging (Internally assigned no.)
3	NTRACK Column Number

MESSAGE SUBTYPE DESCRIPTION

Acknowledge new ASO target (coupled with Message 25)

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
TCA(18)	ASO(6)		

MOPADS MESSAGE DESCRIPTION

Message No. 25 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	3
3	Functional Type	3
4	Message Subtype	6
5	Message Priority	-7

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	1
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	6 or 7
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 3
2	NTRACK Column Number
3	IHIPIR Status:= 1 Lock, = 2 No Lock
4	Raid Size: 1-Few, 2-Few, 3-Many

MESSAGE SUBTYPE DESCRIPTION

Report IHIPIR Status to the TCO

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
FCO(8)	TCO(24)	FCO(9)	TCO(24)

MOPADS MESSAGE DESCRIPTION

Message No. 26 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	1
3	Functional Type	3
4	Message Subtype	7
5	Message Priority	-6

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK column number

MESSAGE SUBTYPE DESCRIPTION

Ready to Fire Message (HAWK to BN Only) (Coupled with Message 4 from Q-73)

Name: Riley Goodin
Date: 11/8/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
TCO(24)	BN(22)		

MOPADS MESSAGE DESCRIPTION

Message No. 27 — MESSAGE ID ELEMENT — Page 1 of 1

Element		Description	Value
1	*	Receiver CRN	-
2		Operator Type	3
3		Functional Type	3
4		Message Subtype	8
5		Message Priority	-6

MESSAGE DATA LINK ELEMENT

Element		Description	Value
1		Communication Network 1-Voice 2-ATDL	1
2		Acknowledgement Required 1 - Yes 2 - No	2
3		Unused	-
4		ATDL Code (Unused)	-
5	*	Time Message Sent	-
6	*	Message Number	-
7	*	Sender CRN	-
8		Sender Operator Type	6 or 7
9		Sender System Module Type	4
10		Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK Column Number

MESSAGE SUBTYPE DESCRIPTION

Temporary No Kill Message - FCO then waits for a Method of Fire Command (Subtype 14) or an order No Kill Command (subtype 15)

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
FCO(14)	TCO(32)		

MOPADS MESSAGE DESCRIPTION

Message No. 28 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	6 or 7
3	Functional Type	1
4	Message Subtype	12
5	Message Priority	4

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 2
2	NTRACK Column Number
3	Manual or Automatic Lock = 1 Manual, 2-Automatic

MESSAGE SUBTYPE DESCRIPTION

Assign new target - TCO task nodes 187 or 171 cause a sending of this message

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
TCO(23)	FCO(8)		
TCO(22)	FCO(8)		

MOPADS MESSAGE DESCRIPTION

Message No. 30 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	6 or 7
3	Functional Type	1
4	Message Subtype	14
5	Message Priority	6

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 2
2	NTRACK Column Number
3	Method of Fire = 1 Shoot-Look-Shoot, 2 Ripple Fire

MESSAGE SUBTYPE DESCRIPTION

Method of Fire Command - Shoot-Look-Shoot means shoot one then evaluate before shotting another. Ripple means shoot two missiles

Name: Riley Goodin
Date: 8/4/83

System Module: IHAWK
Project: MOPADS

From	To	From	To
TCO(27)	FCO(13)		

MOPADS MESSAGE DESCRIPTION

Message No. 32 — MESSAGE ID ELEMENT — Page 1 of 1

Element	Description	Value
1	* Receiver CRN	-
2	Operator Type	6 or 7
3	Functional Type	1
4	Message Subtype	16
5	Message Priority	-5

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	1
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	3
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	NTRACK Column Number

MESSAGE SUBTYPE DESCRIPTION

Order FCO to Break Lock

Name: Riley Goodin	System Module: IHAWK
Date: 8/4/83	Project: MOPADS

From	To	From	To
TCO(35)	FCO(16)		
TCO(26)	FCO(16)		

MOPADS MESSAGE DESCRIPTION

Message No. 34	MESSAGE ID ELEMENT	Page 1 of 1
Element	**Description**	**Value**
1	* Receiver CRN	-
2	Operator Type	3
3	Functional Type	4
4	Message Subtype	10
5	Message Priority	3

MESSAGE DATA LINK ELEMENT

Element	Description	Value
1	Communication Network 1-Voice 2-ATDL	2
2	Acknowledgement Required 1 - Yes 2 - No	2
3	Unused	-
4	ATDL Code (Unused)	-
5	* Time Message Sent	-
6	* Message Number	-
7	* Sender CRN	-
8	Sender Operator Type	4
9	Sender System Module Type	4
10	Task Node Number Sent From	-

* Must be set at the time the message is sent

VARIABLE MESSAGE FORMAT

Element	Description
1	#WORDS = 1
2	Row number of Track data of Track determined to be friendly

MESSAGE SUBTYPE DESCRIPTION

Track identified as friendly

Name: Joe Polito	System Module: IHAWK
Date: 10/4/83	Project: MOPADS

From	To	From	To
TCA(20)	TCO(32)		

III. USER WRITTEN PROGRAMS

1-0 OVERVIEW OF THE FLOW OF CONTROL

The user-written code for the IHAWK system module is accessed via subroutine UTHWKW (called by subroutine UTASK). These subroutines are divided into code to be used at first predecessor completion time, release time, start time, complete time or clear time. There are various types of processing that will go on at these different task times. When a node is being processed, MSAINT calls subroutine UTASK. UTASK determines if it is an IHAWK node. If so, UTASK calls UTHWKW. UTHWKW then calls the subroutine that performs processing for that node.

The IHAWK MSAINT network regulates the flow of the entities through the task nodes. Branching is accomplished either deterministically where the operator follows a logical sequence of actions or conditionally where the operator may do one node verses another node depending on the state of the system.

2-0 EXTERNAL FILE USAGE

There is no direct external file access in the IHAWK system module.

3-0 SUBPROGRAM DESCRIPTIONS

This section contains a copy of the subprogram descriptions for each of the IHAWK user code subprograms. These subprograms are on the following pages.

```
      SUBROUTINE CLASOU (IOPP)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE IS USED TO COMPUTE THE SCREEN CLUTTER FOR
C            THE ASO.II ALSO UPDATES THE OSV FOR SCREEN CLUTTER.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C
```

```
      SUBROUTINE CLFCOU (IOPP)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE IS USED TO COMPUTE THE SCREEN CLUTTER FOR
C            THE FCO AND UPDATES THE OSV.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C
```

```
      SUBROUTINE CLTCAU (IOPP)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE COMPUTES THE SCREEN CLUTTER FOR TCA AND
C            UPDATES HIS OSV.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C
```

```
      SUBROUTINE CLTCOU (IOPP)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE IS USED TO COMPUTE THE SCREEN CLUTTER FOR
C            THE TCO AND UPDATES THE OSV.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C
```

```
      SUBROUTINE DCRMSU(IFLAG)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE IS USED TO DECREASE MISSILE COUNTS WHEN AN
C            IHAWK MISSILE IS FIRED FROM THE LAUNCHER.
C
C**OUTPUT PARAMETERS:  IFLAG=0 IF MORE HOT MISSILES REMAIN ON THE
C                              SELECTED LAUNCHER
C                            =1 IF SELECTED LAUNCHER IS OUT OF HOT
C                              MISSILES
C                            =2 IF ALL HOT MISSILES ARE EXPENDED
C
```

```
      SUBROUTINE DEADU(IOPT,IDOPP,NPLACE)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE IS USED TO CHECK IF THE HAWK HAS BEEN
C           DESTROYED OR ALL THE MISSILES HAVE BEEN EXPENDED FOR
C            BOTH FIRE SECTIONS.
C
C**INPUT PARAMETERS:   IOPT=DEAD MODE TYPE
C                          =1 IF HAWK IS DESTROYED
C                          =2 IF BOTH FIRE SECTIONS HAVE
C                             EXPENDED ALL MISSILES
C                      IDOPP=OPERATOR ID
C                      NPLACE=TASK OCCURRENCE TIME
C
```

```
      SUBROUTINE FNDOPW (IOPRTY,ICOPRN,NTASKN,ID)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS SUBROUTINE WILL FIND THE TASK NUMBER A PARTICULAR
C            HAWK OPERATOR IS AT AND WILL RETURN THE ID OF THAT OPERATOR
C
```

```
C**INPUT PARAMETERS:  IOPRTY=HAWK OPERATOR TYPE
C                           =3 TCO
C                           =4 TCA
C                           =5 ASO
C                           =6 FCO A
C                           =7 FCO B
C                     ICOPRN=COPY ROU NUMBER
C
C**OUTPUT PARAMETERS: NTASKN=TASK NUMBER OPERATOR IOPRTY IS AT
C                     ID=THAT OPERATOR'S ID
C
```

```
      SUBROUTINE GEVALW(IDOP,NADSM,NCOPI,NGS,IOPR,GS,NNG)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      GEVALW WILL EVALUATE AN OPERATOR'S GOAL STATE VECTOR
C      FOR THE IHAWK.
C
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROU NUMBER
C      NGS-ACTUAL LENGTH OF GS
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR STATE VECTOR
C--OUTPUT PARAMETERS:
C      GS(NGS)-GOAL STATE VECTOR
C      NNG-THE NUMBER OF GOALS THE OPERATORS OF THE SYSTEM HAVE.
C           (I.E. ONLY THE FIRST NNG ELEMENTS OF GS HAVE MEANINGFUL
C            INFORMATION). IF THE OPERATOR DOES NOT HAVE GOAL I,
C            THEN GS(I)=-1.E10.
```

```
      SUBROUTINE GEV1W(IDOP,NCOPI,IOPR,GS1)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      GEV1W WILL EVALUATE GOAL 1 FOR THE IHAWK. GOAL 1 IS ATTACK
C      ASSIGNED TRACKS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NCOPI-COPY ROU NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
```

```
C--OUTPUT PARAMETERS:
C       GS1-GOAL STATE FOR GOAL 1

------------------------------------------------------------------------

      SUBROUTINE GEV2W(IDOP,NCOPI,IOPR,GS2)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV2W WILL EVALUATE GOAL 2 FOR THE IHAWK. GOAL 2 IS SELF
C       DEFENSE.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS2-GOAL STATE FOR GOAL 2

------------------------------------------------------------------------

      SUBROUTINE GEV3W(IDOP,NCOPI,IOPR,GS3)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV3W WILL EVALUATE GOAL 3 FOR THE IHAWK. GOAL 3 IS
C       PROTECT CRITICAL ASSETS.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS3-GOAL STATE FOR GOAL 3

------------------------------------------------------------------------

      SUBROUTINE GEV4W(IDOP,NCOPI,IOPR,GS4)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV4W WILL EVALUATE GOAL 4 FOR THE IHAWK. GOAL 4 IS
C       IDENTIFY TRACKS
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS4-GOAL STATE FOR GOAL 4

------------------------------------------------------------------------
```

```
      SUBROUTINE GEV5W(IDOP,NCOPI,IOPR,GS5)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV5W WILL EVALUATE GOAL 5 FOR THE IHAWK. GOAL 5 IS
C       RECEIVE MESSAGES.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS5-GOAL STATE FOR GOAL 5

-------------------------------------------------------------------------

      SUBROUTINE GEV6W(IDOP,NCOPI,IOPR,GS6)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV6W WILL EVALUATE GOAL 6 FOR THE IHAWK. GOAL 6 IS
C       MAXIMIZE MISSILES.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS6-GOAL STATE FOR GOAL 6

-------------------------------------------------------------------------

      SUBROUTINE GEV7W(IDOP,NCOPI,IOPR,GS7)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       GEV7W WILL EVALUATE GOAL 7 FOR THE IHAWK. GOAL 7 IS
C       PROTECT FRIENDS
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS7-GOAL STATE FOR GOAL 7

-------------------------------------------------------------------------
```

```
      SUBROUTINE SEV8W(IDOP,NCOPI,IOPR,GS8)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       SEV8W WILL EVALUATE GOAL 8 FOR THE IHAWK. GOAL 8 IS
C       MINIMIZE CHAP ONLY TARGETS.
C--INPUT PARAMETERS:
C       IDOP-OPERATOR ID
C       NCOPI-COPY ROW NUMBER OF THE OPERATOR
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAR OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GS8-GOAL STATE FOR GOAL 8
```

```
      SUBROUTINE HAWKEW(KODE)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--INPUT PARAMETERS:
C       HAWKEW PERFORMS SPECIAL ERROR PROCESSING FOR MOPADS ERROR
C       CODES 6000-6999 WHICH ARE GENERATED IN THE IHAWK SYSTEM
C       MODULE CODE
C--INPUT PARAMETERS:
C       KODE-ERROR CODE VALUE
```

```
      FUNCTION IDLCKW (NTRKCL,ICOPRN)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS FUNCTION IS USED TO DETERMINE IF IHIPIR LOCK WAS
C            ACHEIVED.
C
C**INPUT PARAMETERS:   NTRKCL=NTRACK COLUMN NUMBER FOR TRACK DATA
C                      ICOPRN=OPERATOR COPY NUMBER TRYING TO ACHEIVE
C                             LOCK
C
C**OUTPUT PARAMETERS:  IDLCKW=1 IF LOCK WAS ACHEIVED
C                             =2 IF LOCK WAS NOT ACHEIVED
C
```

```
      FUNCTION IFFNU (IOPP,NTRKCL)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS FUNCTION IS USED TO DETERMINE THE IFF RETURN WHEN THE
C            TCA CHALLENGES A TARGET.
C
C**INPUT PARAMETERS:   IOPP=OPERATOR IF
C                      NTRKCL=TRACK COLUMN NUMBER IN NTRACK
C
C**OUTPUT PARAMETERS:  IFFNU=1 HOSTILE
C                           =2 FRIEND
C                           =3 UNKNOWN
C
```

```
      SUBROUTINE INITU(NRUN)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      INITU PERFORMS INITIALIZATION FOR THE IHAWK SYSTEM MODULE.
C--INPUT PARAMETERS:
C      NRUN-RUN NUMBER
C           0-DO ONE-TIME INITIALIZATION BEFORE ANY RUNS
C           N-INITIALIZE FOR RUN N
```

```
      FUNCTION IRDSZU(NTRKCL)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS FUNCTION IS USED TO DETERMINE THE FCO'S PERCEPTION
C            OF THE RAID SIZE.
C
C**INPUT PARAMETERS:   NTRKCL=NTRACK COLUMN NUMBER FOR THE TRACK
C
C**OUTPUT PARAMETERS:  IRDSZU=1 IF RAID SIZE IS 1
C                             =2 IF RAID SIZE IS 2 OR 3
C                             =3 IF RAID SIZE IS > 3
C
```

```
      FUNCTION KILLTU(IOPP,NTRKCL)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS FUNCTION IS USED TO DETERMINE IF AN IHAWK ENGAGEMENT
C            WAS EFFECTIVE
C
C**INPUT PARAMETERS:   IOPP=OPERATOR ID
C                      NTRKCL=NTRACK COLUMN NUMBER
C
C**OUTPUT PARAMETERS:  KILLTU=0 INEFFECTIVE
C                             =1 EFFECTIVE
C                             =2 PARTIALLY EFFECTIVE
C
```

```
      FUNCTION NFRMSU(IOPRCN,NTRKCL)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C**PURPOSE:  THIS FUNCTION EVALUATES WHETHER THE TCO SHOULD ORDER THE
C            FCO TO PRESS THE NO KILL BUTTON OR TO FIRE MORE MISSILES.
C
C**INPUT PARAMETERS:   IOPRCN=COPY ROW NUMBER OF THE TCO
C                      NTRKCL=NTRACK COLUMN POINTER FOR THE TRACK
C
C**OUTPUT PARAMETERS:  NFRMSU=1 IF TCO SHOULD ORDER FCO TO PRESS NO KILL
C                             =2 IF TCO SHOULD ORDER FCO TO FIRE MORE
C                               MISSILES
C
```

```
      SUBROUTINE OEASOU(IDOP,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      OEASOU WILL COMPUTE EXPECTED GOAL STATES FOR THE IHAWK ASO
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OEASOU WILL
C            EVALUATE THE EXPECTED GOALS GIVEN THAT THE OPERATOR
C            PERFORMS TASK JTASK
```

```
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM TASK JTASK.
C               ON INPUT, GSJ=GS
C      TJ-EXPECTED TIME(MINUTES) TO PERFORM TASK JTASK. IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C         THEN DO NOT CHANGE TJ.

------------------------------------------------------------------------

      SUBROUTINE OEFCOU(IDOP,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      OEFCOU WILL COMPUTE EXPECTED GOAL STATES FOR THE IHAWK FCO
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OEFCOU WILL
C            EVALUATE THE EXPECTED GOALS GIVEN THAT THE OPERATOR
C            PERFORMS TASK JTASK
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM TASK JTASK.
C               ON INPUT, GSJ=GS
C      TJ-EXPECTED TIME(MINUTES) TO PERFORM TASK JTASK. IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C         THEN DO NOT CHANGE TJ.

------------------------------------------------------------------------

      SUBROUTINE OETCAU(IDOP,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      OETCAU WILL COMPUTE EXPECTED GOAL STATES FOR THE IHAWK TCA
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OETCAU WILL
C            EVALUATE THE EXPECTED GOALS GIVEN THAT THE OPERATOR
C            PERFORMS TASK JTASK
```

```
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM TASK JTASK.
C               ON INPUT, GSJ=GS
C      TJ-EXPECTED TIME(MINUTES) TO PERFORM TASK JTASK. IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C         THEN DO NOT CHANGE TJ.

------------------------------------------------------------------------

      SUBROUTINE OETCOU(IDOP,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      OETCOU WILL COMPUTE EXPECTED GOAL STATES FOR THE IHAWK TCO
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OETCOU WILL
C            EVALUATE THE EXPECTED GOALS GIVEN THAT THE OPERATOR
C            PERFORMS TASK JTASK
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM TASK JTASK.
C               ON INPUT, GSJ=GS
C      TJ-EXPECTED TIME(MINUTES) TO PERFORM TASK JTASK. IF THE
C         OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C         THEN DO NOT CHANGE TJ.

------------------------------------------------------------------------

      SUBROUTINE OEVALU(IDOP,NADSM,NCOPI,GS,NNG,JTASK,IOPR,GSJ,TJ,*)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C      OEVALU WILL COMPUTE THE EXPECTED GOAL STATE VECTOR FOR
C      THE IHAWK OPERATORS.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID
C      NADSM-SYSTEM MODULE TYPE
C      NCOPI-COPY ROW NUMBER
C      GS(NNG)-CURRENT GOAL STATE VECTOR
C      JTASK-OPERATOR TASK NUMBER OF THE OPERATOR. OEVALU WILL
C            EVALUATE EXPECTED GOAL STATES GIVEN THAT THE
C            OPERATOR PERFORMS TASK JTASK.
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C       GSJ(NNG)-EXPECTED GOAL STATES RESULTING FROM PERFORMING
C                TASK JTASK. ON INPUT GSJ=GS.
C       TJ-EXPECTED TIME (MINUTES) TO PERFORM TASK JTASK.  IF THE
C          OPERATOR CANNOT PERFORM OR DOES NOT PERFORM TASK JTASK,
C          THEN TJ=-1.
C--ALTERNATE RETURNS:
C       1-JTASK IS GREATER THAN THE MAXIMUM TASK NUMBER THAT THE OPERATOR
C         PERFORMS.  OEVALU WILL BE CALLED REPEATEDLY WITH GREATER
C         VALUES OF JTASK UNTIL THIS CONDITION OCCURS.

------------------------------------------------------------------------

      SUBROUTINE T---U (NPLACE)
C
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C
C**PURPOSE: USER CODE FOR IHAWK TASK NODE ---
C
C**INPUT PARAMETER: NPLACE - TASK OCCURRENCE TIME
C

 -------- THIS TEMPLATE IS TYPICAL FOR ALL TASK NODE PROGRAMS-------

------------------------------------------------------------------------


      SUBROUTINE THR2U(IDOP,NCOPI,JTRK,XLB,TOA,NTRK,ISTAT)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       THR2U WILL LOCATE THE TRACK THAT IS MOST THREATENING TO
C       AN IHAWK. THIS SUBPROGRAM IS USED TO EVALUATE THE SELF
C       DEFENSE GOAL.  THR2U WILL ALSO EVALUATE A SINGLE TRACK,
C       BASED ON THE VALUE OF JTRK.
C--INPUT PARAMETERS:
C       IDOP-THE OPERATOR'S ID
C       NCOPI-COPY ROW NUMBER FOR THE OPERATOR
C       JTRK-TRACK COLUMN NUMBER IN NTRACK
C            0-EVALUATE ALL TRACKS
C            K-EVALUATE ONLY THE TRACK IN COLUMN K
C       XLB-LOWER BOUND ON TOA.  THE TRACK WITH THE MINIMUM TIME
C           BUT GREATER THAN XLB WILL BE RETURNED. XLB .GT. 0
C           IS USED TO EXCLUDE THE MOST THREATENING TRACK WHEN
C           ESTIMATING THE IMPACT ON GOALS OF VARIOUS TASKS.
```

```
C--OUTPUT PARAMETERS:
C       TOA-TIME OF ARRIVAL OF THE MOST THREATENING TRACK.
C       NTRK-NTRACK COLUMN OF THE MOST THREATENING TRACK.
C       ISTAT-TRACK STATUS(SEE THR1Y)

------------------------------------------------------------------------

      SUBROUTINE THR3W(IDOP,NCOPI,JTRK,XLB,TOA,NTRK,ISTAT)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       THR3W WILL LOCATE THE TRACK THAT IS MOST THREATENING TO
C       AN IHAWK'S PROTECTED SITES.
C       THR3W WILL ALSO EVALUATE A SINGLE TRACK,
C       BASED ON THE VALUE OF JTRK.
C--INPUT PARAMETERS:
C       IDOP-THE OPERATOR'S ID
C       NCOPI-COPY ROW NUMBER FOR THE OPERATOR
C       JTRK-TRACK COLUMN NUMBER IN NTRACK
C            0-EVALUATE ALL TRACKS
C            K-EVALUATE ONLY THE TRACK IN COLUMN K
C       XLB-LOWER BOUND ON TOA.  THE TRACK WITH THE MINIMUM TIME
C           BUT GREATER THAN XLB WILL BE RETURNED. XLB .GT. 0
C           IS USED TO EXCLUDE THE MOST THREATENING TRACK WHEN
C           ESTIMATING THE IMPACT ON GOALS OF VARIOUS TASKS.
C--OUTPUT PARAMETERS:
C       TOA-TIME OF ARRIVAL OF THE MOST THREATENING TRACK.
C       NTRK-NTRACK COLUMN OF THE MOST THREATENING TRACK.
C       ISTAT-TRACK STATUS(SEE THR1Y)

------------------------------------------------------------------------

      FUNCTION USERFW(ICODE)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C        USERFW  EVALUATES  THE USER FUNCTION FOR THE IHAWK
C--INPUT PARAMETERS:
C       ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C       USERFW-VALUE OF THE USER FUNCTION
```

```
      FUNCTION USERNW(ICODE)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       USERNW EVALUATES THE INPUT USER FUNCTION FOR THE IHAWK
C--INPUT PARAMETERS:
C       ICODE-USER FUNCTION CODE
C--OUTPUT PARAMETERS:
C       USERNW-VALUE OF THE USER FUNCTION
```

```
      SUBROUTINE UTHWKW(NT,NPLACE)
C--MODULE: MOPADS IHAWK SYSTEM MODULE
C--REFERENCE: MOPADS VOLUME 5.16
C--PURPOSE:
C       UTHWKW PROCESSES CALLS FROM UTASK FOR THE IHAWK
C       SYSTEM MODULE.
C--INPUT PARAMETERS:
C       NT-TASK NODE NUMBER
C       NPLACE-TASK NODE OCCURRANCE TIME(SEE UTASK)
```

4-0 ERROR PROCESSING

When an error involving the IHAWK user code is detected a call is made to subroutine SERR. The only parameter this subroutine has is the error code. The numbers 6000 to 6999 are reserved for errors detected in the IHAWK user code. Table III-1 lists all the error codes and their definitions.

Table III-1. IHAWK Error Codes and Messages

Error Code	Description	Programs
6000	No track was found in data base corresponding to the current track for the IHAWK fire section	T8W
6001	The current track engaged by the HAWK does not match the track referred to in the message.	T13W
6002	Self-clearing operation has failed	T136W
6003	User clearing operation has failed	T136W
6004	Fire section status does not require the FCO to place his fire section out of action.	T10W
6005	FCO has received a message from the TCO to break lock on a target other than his current target.	T16W
6006	No hot missiles available for launchers.	T48W
6007	Effect of current fire section status on FCO clutter has not been accounted for.	CLFCOW
6008	The track (changing from) was not found in the data base.	T16W
6009	Too many tracks found in data base.	T16W
6010	Row does not exist in data base or it does not contain track data.	T31W, T110W
6011	Not used	
6012	Invalid message subtype for a Q-73 message to the TCO.	TT1W
6013	Invalid message subtype for a FCO message to the TCO.	T78W
6014	Invalid message subtype for an ASO message to the TCO.	T79W
6015	Cannot find FCO.	

Table III-1. (continued)

Error Code	Description	Programs
6016	Unable to clear FCO	T28W,T60W, T63W,T153W, T158W
6017	Invalid message subtype for a TCA message to the TCO.	T81W
6030	Could not find operator ID in NCOPY array.	FNDOPW
6031	Could not find operator at a task.	FNDOPW
6032	Operator not found at appropriate task	T31W,T73W
6033	Invalid operator type.	FNDOPW
6500	Invalid user function code	USERFW, USERNW
6501	No message was found for the ASO when one was expected.	OETCAW
6502	Incorrect message type received by TCA.	OETCAW
6503	Incorrect message type received by ASO.	OEASOW
6504	Incorrect node number.	UTHWKW
6505	An attempt was made to make a primary assignment to a track that already has a primary assignment.	TT1W

IV. MSAINT NETWORK DATA

The following is a listing of the MSAINT network data.

```
GEN,IHAWK,9,7,83,4*
POP,6,5,15,10,10*
DIS,1,CO,0.0*
DIS,2,CO,0.1*
DIS,3,CO,0.001*
DIS,4,CO,9999.0*
DIS,5,CO,15.0*
UTI,1,NUMIPAR*
UTI,2,NUMCHAR*
UBO,1,THREAT*
IRA,1,3,SC,1,4,SC,0*
IRA,2,3,SC,1,4,SC,0*
IRA,3,3,SC,1,4,SC,0*
IRA,4,3,SC,2,4,SC,0*
IRA,5,3,SC,2,4,SC,0*
IRA,6,3,SC,2,4,SC,0*
LRE,1,ILCHR1A,2,ILCHR2A,3,ILCHR3A,4,ILCHR1B,5,ILCHR2B,6,ILCHR3B*
IMO,1,A*
*
*   SOURCE TASKS FOR THE IHAWK SYSTEM MODULE
*
TAS,45,SOURCTCO,,,DS,1,,,SO*
UTC,45,0.0,0.0,1.0*
MOD,45,1,D,T*
DET,45,40*
*
TAS,46,SOURCTCA,,,DS,1,,,SO*
UTC,46,0.0,0.0,1.0*
MOD,46,1,D,T*
DET,46,40*
*
TAS,47,SOURCASC,,,DS,1,,,SO*
UTC,47,0.0,0.0,1.0*
MOD,47,1,D,T*
DET,47,40*
*
TAS,48,SOURCFCA,,,DS,1,,,SO*
UTC,48,0.0,0.0,1.0*
MOD,48,1,D,T*
DET,48,40*
```

```
*
TAS,49,SOURCFCR,,,DS,1,,,SO*
UTC,49,0.0,0.0,1.0*
MOD,49,1,D,T*
DET,49,40*
*
*   THIS CONTROL NETWORK IS FOR SCHEDULING LAUNCHER DOWNTIME
*   DUE TO RELOADING HOT MISSILES
*
TAS,136,LCHRDLAY,1,1,DS,5*
UTC,136,0.0,0.0,1.0*
MOD,136,1,D,T*
*  NO BRANCH, TREATED LIKE A SINK NODE
*
*   THIS NODE IS THE DEAD NODE, USED FOR REMOVING OPERATORS
*   THAT ARE NO LONGER PART OF THE SIMULATION (E.G. HAWK IS
*   DESTROYED (REMOVE ALL OPERATORS),FIRE UNIT HAS EXPENDED
*   ALL MISSILES (REMOVE THE CORRESPONDING FCO)
*
TAS,90,DEADNODE,1,1,DS,1*
UTC,90,0.0,3.0,1.0*
MOD,90,1,D,T*
*   NO BRANCH, ENTITY IS DESTROYED
*
*   OPERATOR TASK 1, ASO MONITOR CRT,CONTROLS,INDICATORS
*
TAS,1,ASOSTNDB,1,1,DS,1*
UTC,1,1.0,3.0,1.0*
DET,1,40*
*
*  OPERATOR TASK 2, DETECT TARGET AT CWIDC
*
TAS,2,DTCTCWAR,1,1,DS,1*
UTC,2,2.0,3.0,1.0*
DET,2,40*
*
*   OPERATOR TASK 3, ESTABLISH TARGET PRIORITY (ASO)
*
TAS,3,ESTTARPR,1,1,DS,1*
UTC,3,3.0,1.0,1.0*
DET,3,106*
*
TAS,106,LOWPRCOF,1,1,DS,1*
UTC,106,3.0,2.0,1.0,(12)1.0*
DET,106,40*
```

```
*
*   OPERATOR TASK 4, PREEMPT LOWER PRIORITY TARGET (ASO)
*
TAS,4,REQCLRAL,1,1,DS,1*
UTC,4,4.0,1.0,1.0*
MOD,4,1,D,T*
DET,4,107*
*
TAS,107,WAITTCOM,1,1,DS,4*
UTC,107,4.0,0.0,1.0*
MOD,107,1,D,T*
*   NO BRANCH,OPERATOR MUST BE CLEARED TO TASK 108
*
TAS,108,PRSALCNL,1,1,DS,1*
UTC,108,4.0,2.0,1.0,(12)1.0*
DET,108,40*
*
*   OPERATOR TASK 5, TCC ALERT (ASO)
*
TAS,5,POSCUCRS,1,1,DS,1*
UTC,5,5.0,1.0,1.0*
DET,5,109*
*
TAS,109,PHUSLEU,1,1,DS,1*
UTC,109,5.0,0.0,1.0*
DET,109,110*
*
TAS,110,PRSTCCAB,1,1,DS,1*
UTC,110,5.0,2.0,1.0,(12)1.0*
DET,110,40*
*
*   OPERATOR TASK 6, MARK TARGET AS ACCEPTED BY TCC (ASO)
*
TAS,6,OBSVALEX,1,1,DS,1*
UTC,6,6.0,1.0,1.0*
DET,6,111*
*
TAS,111,MRKTARGT,1,1,DS,1*
UTC,111,6.0,2.0,1.0,(12)1.0*
DET,111,40*
*
*   OPERATOR TASK 7, FCO MONITOR CRT,CONTROLS,INDICATORS
*
TAS,7,FCOSTNDB,1,1,DS,4*
UTC,7,7.0,3.0,1.0*
MOD,7,1,D,T*
DET,7,40*
```

```
*
*   OPERATOR TASK 3, TRACK TARGET (FCO)
*
TAS,8,DPROCADP,1,1,DS,1*
UTC,8,8.0,1.0,1.0*
CFI,8,120,AEV,2.0,13,IA,,112,AEV,1.0,13,IA*
*
TAS,120,DBADSLTP,1,1,DS,1*
UTC,120,8.0,0.0,1.0*
DET,120,117*
*
TAS,112,OBSRCHLM,1,1,DS,1*
UTC,112,8.0,0.0,1.0*
DET,112,113*
*
TAS,113,DHIPIRLK,1,1,UF,2*
UTC,113,8.0,0.0,1.0*
MOD,113,1,D,T*
CFI,113,117,AEV,0.0,14,IA,,115,AEV,1.0,14,IA*
*
TAS,115,OBVLOCKL,1,1,DS,1*
UTC,115,8.0,0.0,1.0*
DET,115,117*
*
TAS,117,ENDTSK08,1,1,DS,3*
UTC,117,8.0,2.0,1.0,(12)1.0*
MOD,117,1,D,T*
DET,117,40*
*
*  OPERATOR TASK 9, OBTAIN MANUAL HIPIR LOCK (FCO)
*
TAS,9,POSCURS,1,1,DS,1*
UTC,9,9.0,1.0,1.0*
DET,9,121*
*
TAS,121,PRSFLSHE,1,1,DS,1*
UTC,121,9.0,0.0,1.0*
DET,121,122*
*
TAS,122,DCDHIPLK,1,1,UF,2*
UTC,122,9.0,0.0,1.0*
MOD,122,1,D,T*
CFI,122,123,AEV,1.0,14,IA,,124,AEV,2.0,14,IA*
*
TAS,123,TARGALT,1,1,DS,1*
UTC,123,9.0,0.0,1.0*
DET,123,124*
```

```
*
TAS,124,ENDTSK09,1,1,DS,3*
UTC,124,9.0,2.0,1.0*
MOD,124,1,D,T*
DET,124,40*
*
*   OPERATOR TASK 10, PUT FIRE SECTION OUT OF ACTION (FCO)
*
TAS,10,PSHFSOFF,1,1,DS,1*
UTC,10,10.0,1.0,1.0*
CFI,10,88,AEV,0.0,15,IA,,89,AEV,1.0,15,IA*
*
TAS,88,ENDTSK10,1,1,DS,3*
UTC,88,10.0,2.0,1.0,(12)1.0*
MOD,88,1,D,T*
DET,88,40*
*
TAS,89,WAITRELD,1,1,DS,4*
UTC,89,10.0,0.0,1.0*
MODF,89,1,D,T*
*  NO BRANCH MUST BE CLEARED TO TASK NODE 88
*
*   OPERATOR TASK 11, ESTIMATE RAID SIZE (FCO)
*
TAS,11,MONTDOPA,1,1,DS,1*
UTC,11,11.0,1.0,1.0*
DET,11,133*
*
TAS,133,DCDRDSIZ,1,1,DS,1*
UTC,133,11.0,0.0,1.0*
DET,133,134*
*
TAS,134,PSHOFM,1,1,DS,1*
UTC,134,11.0,0.0,1.0*
DET,134,137*
*
TAS,137,ENDTSK11,1,1,DS,3*
UTC,137,11.0,2.0,1.0,(12)1.0*
MOD,137,1,D,T*
DET,137,40*
*
*   OPERATOR TASK 12, SELECT LAUNCHER (FCO)
*
TAS,12,OBSVLNCH,1,1,DS,1*
UTC,12,12.0,1.0,1.0*
DET,12,138*
```

```
*
TAS,138,ENDTS12,1,1,DS,1*
UTC,138,12.0,2.0,1.0,(12)1.0*
CAL,138, 40,AEV,0,1:,IA,,
        143,AEV,1.0,11,IA*
*
*   OPERATOP TASK 13, FIRE MISSILES (FCO)
*
TAS,13,DECFCTYP,1,1,DS,1*
UTC,13,13.0,1.0,1.0*
CFI,13,143,AEV,1.0,13,IA,,139,AEV,2.0,13,IA*
*
TAS,143,PSHFBSLS,1,1,DS,1*
UTC,143,13.0,0.0,1.0*
CAL,143,142,UNC,,,,,136,AGV,0.0,14,IA*
*
TAS,139,PSHFBRP1,1,1,DS,1*
UTC,139,13.0,0.0,1.0*
CAL,139,140,AEV,0.0,14,IA,,
         12,AEV,1.0,14,IA,,
        142,AEV,2.0,14,IA,,
        136,AGV,0.0,14,IA*
*
TAS,140,WAITRPFR,1,1,DS,1*
UTC,140,13.0,0.0,1.0*
DET,140,143*
*
TAS,142,ENDTSK13,1,1,DS,3*
UTC,142,13.0,2.0,1.0*
MOD,142,1,D,T*
DET,142,40*
*
*   OPERATOR TASK 14, EVALUATE TARGET INTERCEPT (FCO)
*
TAS,14,WAITINTP,1,1,UF,1*
UTC,14,14.0,1.0,1.0*
MOD,14,1,D,T*
DET,14,144*
*
TAS,144,AUDTNBST,1,1,DS,1*
UTC,144,14.0,0.0,1.0*
DET,144,145*
*
TAS,145,DROPDOP,1,1,DS,1*
UTC,145,14.0,0.0,1.0*
CFI,145,148,AEV,1.0,14,IA,,146,AEV,0.0,14,IA*
```

```
*
TAS,148,PRSKILLB,1,1,DS,1*
UTC,148,14.0,0.0,1.0*
DET,148,149*
*
TAS,146,GVERBNK,1,1,DS,1*
UTC,146,14.0,0.0,1.0*
CFI,146,147,AEV,1.0,15,IA,,101,AEV,2.0,14,IA*
*
TAS,101,WAITMSGK,1,1,DS,4*
UTC,101,14.0,0.0,1.0*
XOD,101,1,B,T*
*  NO BRANCH, MUST BE CLEARED TO TASK 147 OR TASK 150
*
TAS,147,PRSNOKLL,1,1,DS,1*
UTC,147,14.0,0.0,1.0*
DET,147,149*
*
TAS,150,SHOOTEM,1,1,DS,1*
UTC,150,14.0,0.0,1.0*
MOD,150,1,B,T*
DET,150,160*
*
TAS,149,PRSBRKLX,1,1,DS,1*
UTC,149,14.0,0.0,1.0*
DET,149,160*
*
TAS,160,ENDTSK14,1,1,DS,3*
UTC,160,14.0,2.0,1.0,(12)1.0*
MOD,160,1,B,T*
DET,160,40*
*
*   OPERATOR TASK 15, PUT FIRE SECTION BACK IN ACTION
*
TAS,15,PSHFSACT,1,1,DS,1*
UTC,15,15.0,3.0,1.0*
MOD,15,1,B,T*
DET,15,40*
*
*   OPERATOR TASK 16, PROCESS CHANGE TARGETS COMMAND (FCO)
*
TAS,16,PRCHTRGT,1,1,DS,1*
UTC,16,16.0,1.0,1.0*
DET,16,161*
*
TAS,161,PSHBREAK,1,1,DS,1*
UTC,161,16.0,2.0,1.0,(12)1.0*
DET,161,40*
```

```
*
*   OPERATOR TASK 17, TCA MONITOR CRT,CONTOLS,INDICATORS
*
TAS,17,TCASTNDB,1,1,DS,1*
UTC,17,17.0,3.0,1.0*
DET,17,40*
*
*   OPERATOR TASK 18, ACCEPT OUTDC TARGET FROM ASD (TCA)
*
TAS,18,PCCHANDU,1,1,DS,1*
UTC,18,18.0,1.0,1.0*
DET,18,162*
*
TAS,162,PRCWCONF,1,1,DS,1*
UTC,162,18.0,0.0,1.0*
DET,162,163*
*
TAS,163,CORLTGT,1,1,DS,1*
UTC,163,18.0,2.0,1.0,(12)1.0*
DET,163,40*
*
*   OPERATOR TASK 19, IFF CHALLENGE (TCA)
*
TAS,19,SLTIFFM,1,1,DS,1*
UTC,19,19.0,1.0,1.0*
DET,19,164*
*
TAS,164,WAITSWEP,1,1,DS,1*
UTC,164,19.0,0.0,1.0*
DET,164,165*
*
TAS,165,MONTIFFM,1,1,DS,1*
UTC,165,19.0,2.0,1.0,(12)1.0*
DET,165,40*
*
*   OPERATOR TASK 20,  MARK ON REFLECTION PLOTTER (TCA)
*
TAS,20,DCDATDLS,1,1,DS,1*
UTC,20,20.0,1.0,1.0*
CFI,20,166,AEV,1.0,13,IA,,167,AEV,2.0,13,IA*
*
TAS,166,MRKRFLCT,1,1,DS,1*
UTC,166,20.0,0.0,1.0*
DET,166,167*
*
TAS,167,ENDTSK20,1,1,DS,3*
UTC,167,20.0,2.0,1.0,(12)1.0*
MOD,167,1,D,T*
DET,167,40*
```

```
*
*   OPERATOR TASK 21, TCO MONITOR CRT,CONTROLS,INDICATORS
*
TAS,21,TCOSTNDB,1,1,DS,1*
UTC,21,21.0,3.0,1.0*
DET,21,40*
*
*   OPERATOR TASK 22, DETECT AN ASSIGN IPAR ADP TARGET (TCO)
*
TAS,22,DTCTIPAR,1,1,DS,1*
UTC,22,22.0,1.0,1.0*
DET,22,168*
*
TAS,168,IFFBATT,1,1,DS,1*
UTC,168,22.0,0.0,1.0*
CFI,168,169,AEV,2.0,13,IA,,170,UNC*
*
TAS,169,ENDTSK22,1,1,DS,3*
UTC,169,22.0,2.0,1.0,(12)1.0*
MOD,169,1,D,T*
DET,169,40*
*
TAS,170,PRFLASH,1,1,DS,1*
UTC,170,22.0,0.0,1.0*
DET,170,169*
*
*   OPERATOR TASK 23, MANUALLY ASSIGN TARGET(TCO)
*
TAS,23,EVALTHRT,1,1,DS,1*
UTC,23,23.0,1.0,1.0*
DET,23,178*
*
TAS,178,DIFFMON,1,1,DS,1*
UTC,178,23.0,0.0,1.0*
DET,178,179*
*
TAS,179,DECATDLD,1,1,DS,1*
UTC,179,23.0,0.0,1.0*
CFI,179,180,AEV,1.0,13,IA,,188,AEV,2.0,13,IA*
*
TAS,188,PLTSOVAS,1,1,DS,1*
UTC,188,23.0,0.0,1.0*
DET,188,189*
*
TAS,189,PRENOTH,1,1,DS,1*
UTC,189,23.0,0.0,1.0*
DET,189,190*
```

```
*
TAS,190,OBSLBLS,1,1,DS,1*
UTC,190,23.0,0.0,1.0*
DET,190,187*
*
TAS,180,DETLOC,1,1,DS,1*
UTC,180,23.0,0.0,1.0*
DET,180,181*
*
TAS,181,PTSOVPPI,1,1,DS,1*
UTC,181,23.0,0.0,1.0*
DET,181,182*
*
TAS,182,PALAB,1,1,DS,1*
UTC,182,23.0,0.0,1.0*
DET,182,187*
*
TAS,187,ENDTSK23,1,1,DS,3*
UTC,187,23.0,2.0,1.0,(12)1.0*
MOD,187,1,D,T*
DET,187,40*
*
*   OPERATOR TASK 24, IHIPIR TRACKING (TCO)
*
TAS,24,EVALHDAT,1,1,DS,1*
UTC,24,24.0,1.0,1.0*
DET,24,194*
*
TAS,194,FCOESTS,1,1,DS,1*
UTC,194,24.0,0.0,1.0*
DET,194,195*
*
TAS,195,DIFFTC,1,1,DS,1*
UTC,195,24.0,2.0,1.0,(12)1.0*
DET,195,40*
*
*   OPERATOR TASK 25, SEND CANNOT COMPLY MESSAGE TO Q-73 (TCO)
*
TAS,25,SNDCCOMP,1,1,DS,1*
UTC,25,25.0,3.0,1.0*
DET,25,40*
*
*   OPERATOR TASK 26, HIGHER PRIORITY TARGET TO BE ASSIGNED TO FS (TCO)
*
TAS,26,HIGHPRI,1,1,DS,1*
UTC,26,26.0,1.0,1.0*
DET,26,60*
```

```
*
TAS,60,PRHLDCT,1,1,DS,1*
UTC,60,26.0,0.0,1.0*
DET,60,61*
*
TAS,61,ENDTSK26,1,1,DS,3*
UTC,61,26.0,2.0,1.0,(12)1.0*
MOD,61,1,D,T*
DET,61,40*
*
*    OPERATOR TASK 27, PROCESS HOSTILE TARGET (TCO)
*
TAS,27,SETRFCFR,1,1,DS,1*
UTC,27,27.0,1.0,1.0*
DET,27,62*
*
TAS,62,DCDFIRET,1,1,DS,1*
UTC,62,27.0,0.0,1.0*
DET,62,63*
*
TAS,63,ISSFIREC,1,1,DS,1*
UTC,63,27.0,0.0,1.0*
DET,63,64*
*
TAS,64,ENDTSK27,1,1,DS,3*
UTC,64,27.0,2.0,1.0,(12)1.0*
MOD,64,1,D,T*
DET,64,40*
*
*   OPERATOR TASK 28,ASSIGNED TARGET DETERMINED FRIEND (TCO)
*
TAS,28,PRSSCHT,1,1,DS,1*
UTC,28,28.0,3.0,1.0*
DET,28,40*
*
*   OPERATOR TASK 29, PROCESS FRIEND (TCO)
*
TAS,29,DCAADCPF,1,1,DS,1*
UTC,29,29.0,1.0,1.0*
CFI,29,71,AEV,2.0,13,IA,,69,AEV,1.0,13,IA*
*
TAS,69,PLTRCKSY,1,1,DS,1*
UTC,69,29.0,0.0,1.0*
DET,69,70*
*
TAS,70,PRSSFSW,1,1,DS,1*
UTC,70,29.0,0.0,1.0*
DET,70,72*
```

```
*
TAS,71,STRFSW,1,1,DS,1*
UTC,71,29.0,0.0,1.0*
DET,71,72*
*
TAS,72,ENDTSK29,1,1,DS,3*
UTC,72,29.0,2.0,1.0,(12)1.0*
MOD,72,1,D,T*
DET,72,40*
*
*    OPERATOR TASK 30, EVALUATE MORE MISSILES (TCO)
*
TAS,30,TOGRCCF,1,1,DS,1*
UTC,30,30.0,1.0,1.0*
DET,30,73*
*
TAS,73,BYDEFER,1,1,DS,1*
UTC,73,30.0,0.0,1.0*
CFI,73,74,AEV,1.0,13,IA,,75,AEV,2.0,13,IA*
*
TAS,74,ONOKILL,1,1,DS,1*
UTC,74,30.0,0.0,1.0*
DET,74,75*
*
TAS,75,ENDTSK30,1,1,DS,3*
UTC,75,30.0,2.0,1.0*
MOD,75,1,D,T*
DET,75,40*
*
*   OPERATOR TASK 31, GIVE ASO PERMISSION TO CANCEL ALERT (TCO)
*
TAS,31,PERMCNCL,1,1,DS,1*
UTC,31,31.0,3.0,1.0*
DET,31,40*
*
*    OPERATOR TASK 32, TCO RECEIVE MESSAGES
*
TAS,32,TCORMSG,1,1,DS,1*
UTC,32,32.0,1.0,1.0*
MOD,32,1,D,T*
CFI,32,77,ALV,3.0,13,IA,,
       78,AGV,5.0,13,IA,,
       79,AEV,5.0,13,IA,,
       81,AEV,4.0,13,IA*
*
TAS,77,RECQ73M,1,1,DS,1*
UTC,77,32.0,0.0,1.0*
MOD,77,1,D,T*
DET,77,80*
```

```
*
TAS,78,RECFCOM,1,1,DS,1*
UTC,78,32.0,0.0,1.0*
MOD,78,1,D,T*
DET,78,80*
*
TAS,79,RECASOM,1,1,DS,1*
UTC,79,32.0,0.0,1.0*
MOD,79,1,D,T*
DET,79,80*
*
TAS,81,RECTCAM,1,1,DS,1*
UTC,81,32.0,0.0,1.0*
MOD,81,1,D,T*
DET,81,80*
*
TAS,80,ENDTS32,1,1,DS,3*
UTC,80,32.0,2.0,1.0,(12)1.0*
MOD,80.1,D,T*
DET,80,40*
*
*   OPERATOR TASK 35, PROCESS "HOLD FIRE" COMMAND (TCO)
*
TAS,35,RCVHLDFR,1,1,DS,1*
UTC,35,35.0,1.0,1.0*
DET,35,157*
*
TAS,157,PRACKNOW,1,1,DS,1*
UTC,157,35.0,0.0,1.0*
DET,157,158*
*
TAS,158,PRSDESTR,1,1,DS,1*
UTC,158,35.0,0.0,1.0*
DET,158,159*
*
TAS,159,OFCOBLK,1,1,DS,1*
UTC,159,35.0,2.0,1.0,(12)1.0*
DET,159,40*
*
*   OPERATOR TASK 37, PROCESS "CEASE ENGAGED" COMMAND (TCO)
*
TAS,37,OBCSENGD,1,1,DS,1*
UTC,37,37.0,1.0,1.0*
DET,37,152*
*
TAS,152,PACKNWL,1,1,DS,1*
UTC,152,37.0,0.0,1.0*
DET,152,153*
```

```
*
TAS,153,DCDMSSEN,1,1,DS,1*
UTC,153,37.0,0.0,1.0*
CFI,153,154,AEV,1.0,15,IA,,155,AEV,2.0,15,IA*
*
TAS,154,OFCOBLCK,1,1,DS,1*
UTC,154,37.0,0.0,1.0*
DET,154,155*
*
TAS,155,ENDTSK37,1,1,DS,3*
UTC,155,37.0,2.0,1.0,(12)1.0*
MOD,155,1,D,T*
DET,155,40*
*
*    OPERATOR TASK 40, TASK SEQUENCING
*
TAS,40,TASKSEQU,1,1,DS,1*
UTC,40,40.0,1.0,1.0*
MOD,40,1,D,T*
CFI,40,91,AEV,3.0,3,IA,,
       92,AEV,5.0,3,IA,,
       93,AEV,4.0,3,IA,,
       94,AGV,5.0,3,IA*
*
TAS,91,TCOTSEQ,1,1,DS,1*
UTC,91,40.0,0.0,1.0*
MOD,91,1,D,T*
CFI,91,95,ALV,27.0,14,IA,,
       96,ALV,32.0,14,IA,,
       97,ALV,38.0,14,IA*
*
TAS,95,TCOROUT1,1,1,DS,1*
UTC,95,40.0,2.0,1.0*
MOD,95,1,D,T*
CFI,95,21,AEV,21.0,14,IA,,
       22,AEV,22.0,14,IA,,
       23,AEV,23.0,14,IA,,
       24,AEV,24.0,14,IA,,
       26,AEV,26.0,14,IA*
*
TAS,96,TCOROUT2,1,1,DS,1*
UTC,96,40.0,2.0,1.0*
MOD,96,1,D,T*
CFI,96,27,AEV,27.0,14,IA,,
       28,AEV,28.0,14,IA,,
       29,AEV,29.0,14,IA,,
       30,AEV,30.0,14,IA,,
       31,AEV,31.0,14,IA*
```

```
*
TAS,97,FCOROUT3,1,1,DS,1*
UTC,97,40.0,2.0,1.0*
MOD,97,1,D,T*
CFI,97,32,AEV,32.0,14,IA,,
       33,AEV,33.0,14,IA,,
       37,AEV,37.0,14,IA*
*
TAS,92,ASOTSEQ,1,1,DS,1*
UTC,92,40.0,2.0,1.0*
MOD,92,1,D,T*
CFI,92,1,AEV,1.0,14,IA,,
       2,AEV,2.0,14,IA,,
       3,AEV,3.0,14,IA,,
       4,AEV,4.0,14,IA,,
       5,AEV,5.0,14,IA,,
       6,AEV,6.0,14,IA*
*
TAS,93,TCATSEQ,1,1,DS,1*
UTC,93,40.0,2.0,1.0*
MOD,93,1,D,T*
CFI,93,17,AEV,17.0,14,IA,,
       18,AEV,18.0,14,IA,,
       19,AEV,19.0,14,IA,,
       20,AEV,20.0,14,IA*
*
TAS,94,FCOTSEQ,1,1,DS,1*
UTC,94,40.0,2.0,1.0*
MOD,94,1,D,T*
CFI,94,99,AEV,13.0,14,IA,,100,AGV,12.0,14,IA*
*
TAS,99,FCOROUT1,1,1,DS,1*
UTC,99,40.0,2.0,1.0*
MOD,99,1,D,T*
CFI,99, 7,AEV, 7.0,14,IA,,
        8,AEV, 8.0,14,IA,,
        9,AEV, 9.0,14,IA,,
       10,AEV,10.0,14,IA,,
       11,AEV,11.0,14,IA,,
       12,AEV,12.0,14,IA*
*
TAS,100,FCOROUT2,1,1,DS,1*
UTC,100,40.0,2.0,1.0*
MOD,100,1,D,T*
CFI,100,13,AEV,13.0,14,IA,,
       14,AEV,14.0,14,IA,,
       15,AEV,15.0,14,IA,,
       16,AEV,16.0,14,IA*
```

V. REFERENCES

Goodin II, J. R. & Polito, J. User's guide for the IHAWK system module (MOPADS Vol. 3.2). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

VI. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VII. CHANGE NOTICES

LIST OF TABLES

MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE (ADSM)	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

Additional Terminology and Abbreviations

ACN	ADSM Component Number.
ADSM	Air Defense System Module
Reference System Module	A code 11 directory in a MOPADS data base which contains default values for operator and system parameters. It is the "baseline" model for an air defense system.
DB	Data base.

ID	An identifier. ID's of MOPADS data base entries are lists of integer numbers.
ACC	ADSM character code. A single character value uniquely assigned to a reference system module.
DLCC	Data List Character Code. A one, two, or three character mnemonic assigned to certain types of data lists. For example, operator state vectors have a DLCC of "OP."
NECC	Number Equivalent of a Character Code. An integer number that corresponds to a short character string. The capital letters, A-Z, correspond to the integers 1-26 and the digits 0-9 correspond to the numbers 27-36. Thus the string P2B has an NECC of 162902 ("P" corresponds to 16, "2" to 29, and "B" to 02).
DL	Data List. A list of values.
DR	Directory. An index of other directories and/or data lists.

I. INTRODUCTION

This document describes the particular data base created and used by the MOPADS software. MOPADS data base software is described in Polito (1983a). Here we are concerned with the structure and contents of the data base file used to implement the MOPADS simulations. User instructions for exercising MOPADS software are contained in Polito (1983b).

The MOPADS data base contains virtually all of the information required to set up, run, and review MOPADS simulations. During initial phases of development, a MOPADS modeler uses the User Interface (Polito, 1983c) to create reference air defense system modules and air scenarios (Polito, 1983d,e). This information is stored in the MOPADS data base.

With the above building blocks, a MOPADS user will create simulation data sets that contain a specific command and control structure involving (perhaps) multiple copies of the reference system modules and specifying a particular air-scenario. When this simulation data set is executed, the resulting statistics are stored in the data base. The MOPADS user can later examine these statistics and select those he/she desires to be printed.

The data base may contain one or more simulation data sets, so multiple analyses can be performed with one data base file. Also, the data base files provide ideal vehicles for archiving MOPADS data, because the data base contains all of the required data to duplicate a simulation.

II. THE MOPADS DATA BASE

1-0 STRUCTURE OF THE DATA BASE.

Figure II-1 shows the structure of the directories in the MOPADS data base. The "DR" in the boxes indicates a directory as opposed to a data list (data lists are not shown on this figure). Directory labels are shown in capital letters in the larger portions of the boxes. If the label section has an entry in lower case letters, it indicates that an arbitrary number of these directories may exist with user assigned labels. The numbers over the upper right corners are directory codes which are discussed in Section 2-0 below.

The REFERENCE-ADSM directory (code 2) owns all reference system modules. The reference system modules are copied to appropriate locations in a particular command and control structure where they become code 12 directories. Multiple copies of each reference system module may be copied and each code 12 directory may be edited individually.

Multiple battlefield scenarios can be created. They are all owned by the SCENARIOS directory (code 4). Multiple CRITICAL-ASSET-CONFIGURATION (code 13) directories can be created, each of which may contain one or more air scenarios (code 8).

From the menu of reference system modules and scenarios available, the user may construct one or more SIMULATION-DATA-SET (code 1) directories that contain all information necessary to perform MOPADS simulations. Each of the directories is explained in detail in Section III.

2-0 DIRECTORY CODES AND CONTENTS.

The MOPADS data base contains 14 directory types identified by directory codes numbered from zero to 13. More than one instance of some directory types may exist in a single data base. Table II-1 explains the contents of each directory type.

3-0 IDENTIFIERS AND LABELS FOR CODE 11 AND 12 DIRECTORIES.

Code 11 and 12 directories have systematic conventions for assigning labels and ID's. Labels of reference ADSM's (code 11) directories consist of a one-character ADSM code character (ACC) followed by a hyphen (-) and a text string of up to 23 characters.

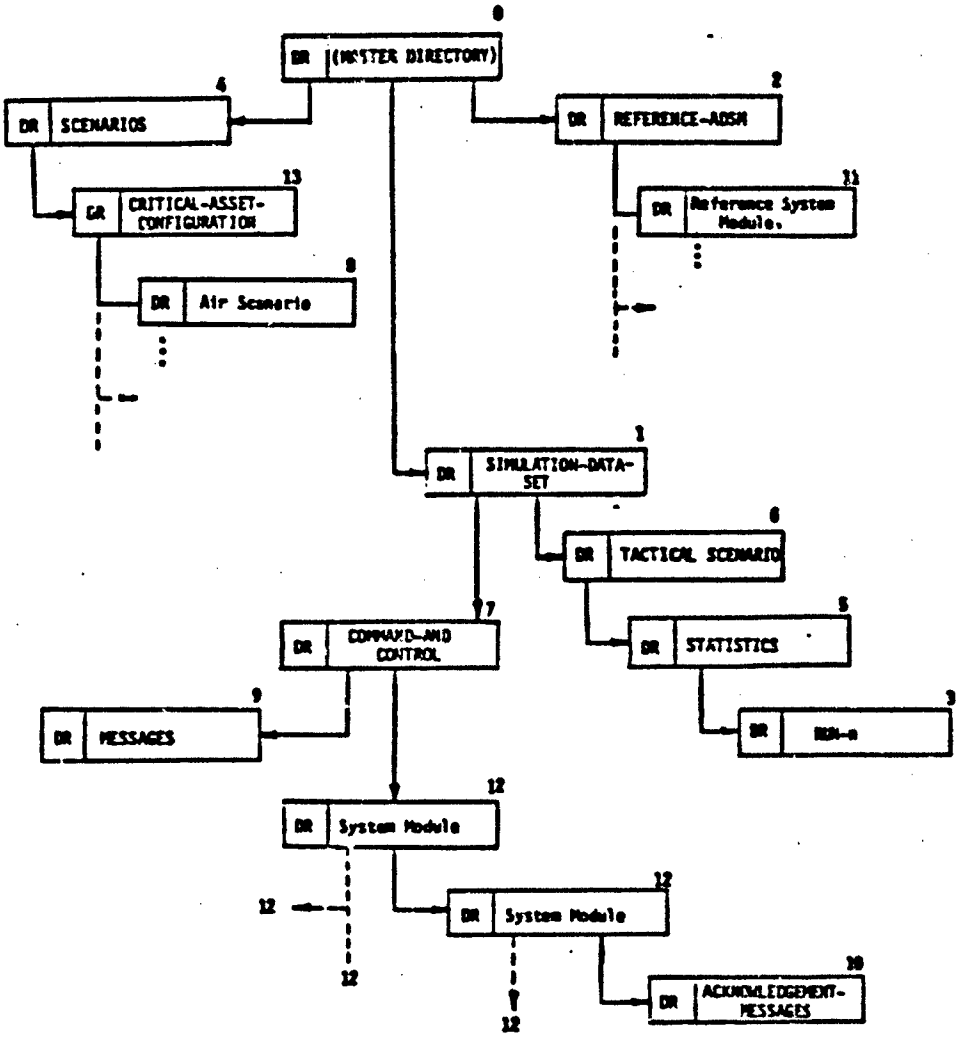


Figure II-1. Structure of MOPADS Data Base.

Table II-1. MOPADS Data Base Directories.

DIRECTORY CODE	LABEL	DESCRIPTON
0	(MASTER-DIRECTORY)	The master directory is the single entry point to the MOPADS DB.
1	SIMULATION-DATA-SET	This directory owns all of the information requird to perform a MOPADS simulation. It owns a command and control configuration, specification of the scenario, and output statistics. More than one SIMULATION-DATA-SET directory may exist.
2	REFERENCE-ADSM	This directory owns all of the reference system modules in the DB. Only one code 2 directory exists.
3	RUN-n*	This directory contains statistics for one run of the simulation (run n).
4	SCENARIOS	This directory owns all of the scenarios in the DB. It is analogous to a code 2 directory. Only one will exist.
5	STATISTICS	This directory owns output statistics from a MOPADS simulation resulting from execution of a code 3 directory specification. More than one of these directories may exist.
6	TACTICAL-SCENARIO	This directory owns all information necessary to run a MOPADS simulation except the command and control configuration. A simulation data set may contain more than one code 6 directory.
7	COMMAND-AND-CONTROL	This directory owns an entire command and control configuration. All of the system modules in the configuration descend from this directory. Each code 1 directory contains only one code 7 directory.

Table II-1. (continued)

DIRECTORY CODE	LABEL	DESCRIPTION
8	air scenario*	These directories contain records of aircraft movements in air battles. As many of these as are needed may exist in each code 13 directory.
9	MESSAGES	This directory contains messages sent among system modules during simulations. Each code 7 directory will own one of these.
10	ACKNOWLEDGEMENT-MESSAGES	Each code 12 directory owns one of these. The directory contains messages that must be acknowledged.
11	reference system module*	Each reference system module is a type 11 directory.
12	system module*	When reference system modules are copied to their positions in a command and control configuration, they become code 12 directories. Their labels are internally assigned as in Section II,3-0.
13	CRITICAL-ASSET-CONFIGURATION	This directory contains the physical layout of the battle area. It defines the coordinate system and location of critical assets.

* Labels or portions of labels in lower case letters imply that the label may vary from instance to instance of these directories.

For example:

Q-BATTALION-Q-73

is an acceptable reference ADSM label. The ACC is "Q". Every reference ADSM must have a unique ACC. The ACC is used in developing mnemonic labels for system modules in the command and control configuration as will be seen soon.

Directory identifiers (DRID) for reference ADSM's have four values whose definitions are shown below.

DRID(1) = 11

DRID(2) = NECC(ACC)

DRID(3) = 0

DRID(4) = basic ACN

The function NECC is the Number Equivalent of Character Code. It converts, in this case, the ACC character to an integer. NECC maps the letters A-Z to the numbers 1-26 and the digits 0-9 to the numbers 27-36.

In addition to the ACC, each reference ADSM is assigned a basic ADSM Component Number (ACN) by the user. The basic ACN is a multiple of 1000 and must be unique for the ADSM. When reference ADSM's are copied as code 12 directories, the basic ACN is incremented to identify each copy (e.g., code 12 directories might have ACN's of 1001, 1002, 1003, etc. for a reference ADSM with a basic ACN of 1000).

The DRID's for code 12 directories are:

DRID(1) = 12

DRID(2) = NECC(ACC)

DRID(3) = sequence number

DRID(4) = ACN

The ACN is assigned as discussed above so every code 12 system module directory has a unique value in its fourth ID element. The sequence number is assigned within the owning directory. For example, suppose there are two battalion Q-73's each with three HAWK batteries. The HAWK batteries for each Q-73 will be numbered 1, 2, and 3 (their sequence numbers). Thus HAWKs belonging to the different Q-73's may have identical values for DRID(3) (but of course their ACN's will be different).

Labels of code 12 directories reflect their location in the command and control configuration. This is done by taking the label of the owning directory and appending the ACC and the DRID(3) value to it. In other words, the "component suffix" is appended to the owning directory's label. For example,

Q2H4

might be the label of the fourth HAWK belonging to the second battalion Q-73. The "4" in the label above is the sequence number stored in DRID(3) of the HAWK ID.

The labels for code 12 directories are assigned automatically by the MOPADS software using information from the reference ADSM's and the command and control configuration.

III. CONTENTS OF THE DIRECTORIES AND DATA LISTS

1-0 CONTENTS OF THIS SECTION

Section III contains lists of the information contained in each directory type of the MOPADS data base. Since this is a reference document, no extensive definitions or discussion of the contents are provided. Expanded discussions are found in user documents referred to in Section I. Also, it is assumed that the reader is familiar with the concepts in Section I and II of MOPADS Volume 5.13 (Polito, 1983a).

The function NECC is used in several instances. NECC can translate strings greater in length than one character (which was the only case presented in Section II, 3-0). In fact, each character is mapped to a two-digit number (which may have a leading zero if the number is less than 10) and these two-digit numbers are concatenated if the input string has more than one character. As examples,

String	NECC(string)
A	1
BA	201
ZZ	2626
ZY	2625

Note that NECC( ) is a number, not a string. Therefore, the length of the input string that can be converted by NECC is limited by the number of digits which the computer can represent as an integer number.

Also, some data list labels have code characters that identify them. These "Data List Code Characters" or DLCC's are one, two, or three characters. A data list label with a DLCC is composed of the DLCC followed by a dash (-) followed by the rest of the label; e.g., OP-OPERATOR-STATE-VECTOR. Frequently, the ID of such a data list contains the NECC of the DLCC. For example, NECC('OP') is 1516.

Each subsequent section contains a description of a directory. A schematic shows the directories and data lists owned by the directory, and the contents of all owned data lists is given in detail.

1-1. Forms Used in This Document.

Three forms are used to describe the contents of the MOPADS data base. The first form is shown in Figure III-1. Each directory is described by one of these forms. The directory code and label are given on the first line. The directory level specified on the second line is the depth of the directory from the master directory. For example, the depth, in Figure III-2 of the Code 2 directory (REFERENCE-ADSM) is one because it is owned directly by the master directory. Similarly, the directory level of Code 11 directories is two.

The owner directory is specified next in Figure III-1. Next is the identifier of this directory. Recall that this identifier is stored physically in the directory that owns this directory. Thus the title "ID IN OWNER DIRECTORY."

The next three items concern the ID's of directories and data lists that are owned by this directory. First the number of words in the ID's is specified. The ranking codes for owned directories and data lists is given next, see Polito (1983a).

The second form shown for each directory is the Directory Contents Form shown in Figure III-2. All owned directories and data lists are shown on this form. In the "TYPE" column will appear "DL" or "DR" to indicate a data list or directory, respectively. For directories, the directory code will be specified in the "CODE" column. The label is given in the center column, and the number of such entities which may be owned by the directory will be given in the "NUMBER" column. This column will be either "1" to indicate that only one of the specified data list or directories may belong to the directory or "as needed" to indicate that more than one may be owned. For example, the master directory may own more than one "SIMULATION-DATA-SET" (Code 1) directory. It's NUMBER would be specified as "as needed" in the directory contents form of the (MASTER DIRECTORY).

The last form is the Data List Description Form shown in Figure III-3. This form is used to describe the detailed contents of all data lists. As discussed in Polito (1983a), each data list may be either real (RDL), integer (IDL), or character (CDL). RDL's and IDL's may be two-dimensional, and all elements of a data list may have a 25-character label. All of this information may be specified on the Data List Description form.

Space exists at the top of the form to specify the data list label, the owner directory, and the data list ID. The data list type is also specified. For example, to specify a two-dimensional, row-oriented, real data list whose maximum column dimension is 21, the following would be filled in on the form:

X RDL/(R) C DIM = 21

MOPADS/DATA BASE DIRECTORY DESCRIPTION

DIRECTORY LABEL(CODE/LABEL):_____

DIRECTORY LEVEL:_____

OWNER DIRECTORY(CODE):_____

ID IN OWNER DIRECTORY:_____

NUMBER OF WORDS IN ID'S OF
OWNED DIRECTORIES & DATA LISTS:_____

RANKING CODE FOR OWNED DIRECTORIES:_____

RANKING CODE FOR OWNED DATA LISTS:_____

DESCRIPTION:

Figure III-1. Directory Description Form.

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE:_____

DIRECTORY LABEL:_____
_____

TYPE	CODE	LABEL	NUMBER

Figure III-2. Directory Contents Form.

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD -

Data List Label (25 char.max):_____ Page ___of ____
Owner Directory Code:_____ DL Id:_____ Rev.Date:_____
DL Type:____IDL/R C DIM =_____ ____RDL/R C DIM =____ ____CDL/max char=_____ Author:_____

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE

Figure III-3. Data List Description Form.

In the case of character data lists, only the maximum number of characters per element may be specified.

The body of the form is filled in to describe the individual data list elements. For one-dimensional data lists, the elements are numbered in the left-hand column, and the label and element descriptions are given in the indicated columns. If initial or default values are specified for elements, they are given with the following notation: "(I)=" implies an initial value and "(D)=" implies a default; e.g., (I) = 16 and (D) = 21.4.

For two-dimensional data lists, one dimension must be fixed. If it is a row-oriented DL, then the column dimension is fixed. In this case, each column generally has a fixed meaning, and as many rows as needed can be included. The column definitions are specified by putting the column numbers in the first column as follows: C1, C2, C3, etc. If a label is provided for that column (actually in row 1 of the data list), it is put in the label column. The column definition is given in the last column as before. For column oriented DL's, the situation is reversed, and row definitions are given with the notation R1, R2, R3, etc. used in the first column of the form.

2-0 THE MASTER DIRECTORY (CODE 0)

MOPADS/DATA BASE — DIRECTORY DESCRIPTION

DIRECTORY LABEL(CODE/LABEL): 0/(MASTER-DIRECTORY)

DIRECTORY LEVEL: 0

OWNER DIRECTORY(CODE): (none)

ID IN OWNER DIRECTORY: (none)

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 2

RANKING CODE FOR OWNED DIRECTORIES: 0

RANKING CODE FOR OWNED DATA LISTS: 0

DESCRIPTION:

MCPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 0

DIRECTORY LABEL: (MASTER-DIRECTORY)

TYPE	CODE	LABEL	NUMBER
DR	1	SIMULATION-DATA-SET	as needed
DR	2	REFERENCE-ADSM	1
DR	4	SCENARIOS	1
DL	-	DB-TITLE	1

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 15
Data List Label (25 char.max): DB-TITLE Owner Directory Code: 0 MASTER DIRECTORY DL Id:010 DL Type:____IDL/R C DIM =_____ ____RDL/R C DIM =____ X CDL/max char =40		Page 1 of 1 Rev.Date: 01 FEB 83 Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1		1st line of title information
2		2nd line of title information etc.

3-0 THE "SIMULATION-DATA-SET" DIRECTORY (CODE 1)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	1|SIMULATION-DATA-SET
DIRECTORY LEVEL:	1
OWNER DIRECTORY(CODE):	0|(MASTER-DIRECTORY)
ID IN OWNER DIRECTORY:	1|(sequence number)
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	2
RANKING CODE FOR OWNED DIRECTORIES:	1
RANKING CODE FOR OWNED DATA LISTS:	1

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 1

DIRECTORY LABEL: SIMULATION-DATA-SET

TYPE	CODE	LABEL	NUMBER
DR	6	TACTICAL-SCENARIO	as needed
DR	7	COMMAND-AND-CONTROL	1
DL	-	COPY-COUNTER	1

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 16

Data List Label (25 char.max): COPY-COUNTER
Owner Directory Code: 1 SIMULATION-DATA-SET DL Id: 010
DL Type: X IDL (R) C DIM = 2 RDL/R C DIM = ____ CDL/max char= _____
Page 1 of 1
Rev.Date: 9 JUN 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C1		The last ACN used for the system module whose basic ACN is (row number) *1000
C2		The number of system modules for this basic ACN currently existing in this simulation data set.
		EXAMPLE: Let the basic ACN be 6000. Then values for row 6 might be C1=6008 and C2=5. This could occur if 8 copies of the 6000 system module were INSERTED and then three of them were subsequently REMOVED.

4-0 THE "REFERENCE-ADSM" DIRECTORY (CODE 2)

MOPADS/DATA BASE **DIRECTORY DESCRIPTION**

DIRECTORY LABEL(CODE/LABEL): 2|REFERENCE-ADSM

DIRECTORY LEVEL: 1

OWNER DIRECTORY(CODE): 0|(MASTER-DIRECTORY)

ID IN OWNER DIRECTORY: 2|1

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 4

RANKING CODE FOR OWNED DIRECTORIES: 2

RANKING CODE FOR OWNED DATA LISTS: 2

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 2

DIRECTORY LABEL: REFERENCE-ADSM

TYPE	CODE	LABEL	NUMBER
DL	-	SKILL-CATEGORIES	1
DR	11	reference ADSM'S	as needed

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 17

Data List Label (25 char.max): SKILL-CATEGORIES Page 1 of 1
Owner Directory Code: 2 | REFERENCE-ADSM DL Id: 1|0|0|0 Rev.Date: 07 NOV 83
DL Type: ___IDL/R C DIM = _____ ___RDL/R C DIM = ___ X CDL/max char = 30 Author: POLLIO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1		PROBABILITY-ESTIMATION
2		TIME-ESTIMATION
3		LTM-SENSORY
4		LTM-SYMBOLIC
5		STM-SENSORY
6		STM-SYMBOLIC
7		NUMERIC-MANIPULATION
8		RECOGNITION
9		UNUSED
10		UNUSED
11		TIME-SHARING
12		DETECTION
13		FINE-MANIPULATION
14		GROSS-MANIPULATION
15		UNUSED
16		GENERAL-PHYSICAL-EFFORT
17		REACTION-TIME
18		TRACKING
19		TEAM-COORDINATION

5-0 THE "RUN-n" DIRECTORY (CODE 3)

MOPABS/DATA BASE DIRECTORY DESCRIPTION

DIRECTORY LABEL(CODE/LABEL): 3|RUN-n

DIRECTORY LEVEL: 4

OWNER DIRECTORY(CODE): 5|STATISTICS

ID IN OWNER DIRECTORY: 3|n

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 2

RANKING CODE FOR OWNED DIRECTORIES: 0

RANKING CODE FOR OWNED DATA LISTS: 0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 3

DIRECTORY LABEL: RUN-n

TYPE	CODE	LABEL	NUMBER
DL	-	TRACE	1
DL	-	STATISTICS-ARRAYS	1

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 23

Data List Label (25 char.max): STATISTICS-ARRAYS
Owner Directory Code: 3|RUN-n DL Id: n|1
DL Type: IDL/R C DIM = X RDL/R C DIM = 1 CDL/max char=
Page 1of1
Rev.Date: 28 NOV 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		The following MSAINT statistics arrays are stored in this data list. For RUN-n, n is the run number. If n = 0, then the data list is for summary statistics. SUMAI JCELS RSTAT SUMAF USOBV USTPV JJCEL TTFNY TASKY COUNTY

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 24

Data List Label (25 char.max): TRACE
Owner Directory Code: 3|RUN-n DL Id: n12
DL Type: IDL/R C DIM = X RDL/R C DIM =25 CDL/max char=
Page 1 of1
Rev.Date:10 OCT 83
Author:POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		Each row of this data list represents one record of trace with up to 25 elements. The meaning of each element is dependent upon the type of the record.

6-0 THE "SCENARIOS" DIRECTORY (CODE 4)

MOPADS/DATA BASE **DIRECTORY DESCRIPTION**

DIRECTORY LABEL(CODE/LABEL): 4|SCENARIOS

DIRECTORY LEVEL: 1

OWNER DIRECTORY(CODE): 0|(MASTER-DIRECTORY)

ID IN OWNER DIRECTORY: 4|0

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 2

RANKING CODE FOR OWNED DIRECTORIES: 0

RANKING CODE FOR OWNED DATA LISTS: 0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 4

DIRECTORY LABEL: SCENARIOS

TYPE	CODE	LABEL	NUMBER
DR	13	CRITICAL-ASSET-CONFIGURATION	as needed

7-0 THE "STATISTICS" DIRECTORY (Code 5)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	5|STATISTICS
DIRECTORY LEVEL:	3
OWNER DIRECTORY(CODE):	6|TACTICAL SCENARIO
ID IN OWNER DIRECTORY:	5|1
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	2
RANKING CODE FOR OWNED DIRECTORIES:	2
RANKING CODE FOR OWNED DATA LISTS:	0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 5

DIRECTORY LABEL: STATISTICS

TYPE	CODE	LABEL	NUMBER
DR	3	RUN-n	multiple
DL	-	MSAINT-DATA	1
DL	-	LABELS	1

NUPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 25

Data List Label (25 char.max): MSAINT-DATA
Owner Directory Code: 5|STATISTICS DL Id: 011
DL Type: IDL/R C DIM = X RDL/R C DIM = 1 CDL/max char=
Page 1 of 3
Rev.Date: 28 NOV 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		The following MSAINT variables and arrays are stored in this data list. NNCLT NSIP NEIP NMCPY NMSYS NCNEXT NCCOL LFOP LFDB NMIRD NIRID NXTID NOPR NXSTA NOPNO IMN

NOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 25

Data List Label (25 char.max): MSAINI-DATA Page 2 of 3
Owner Directory Code: SISTATISTICS DL ID#: 011 Rev.Date: 28 NOV 83
DL Type: IDL/R C DIM = X RDL/R C DIM = 1 CDL/max char= Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		MNCLT
		MNSTP
		MXSTY
		MRUNS
		NSTTS
		NDOP
		TSTRTK
		MNHIS
		NNSTP
		NNHTS
		NCOPY
		NOPRI
		NTRID2
		LSINK
		NSINK
		KSTPE
		KSTTM
		NCELS
		NNCEL

MUPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 25

Data List Label (25 char.max): MSAINI-DATA
Owner Directory Code: 5|STATISTICS DL Id: 0|1
DL Type: ___IDL/R C DIM =_____ X RDL/R C DIM =1 ____CDL/max char=_____

Page 3 of 3
Rev.Date: 28 NOV 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		NMCOP XLOW WIDTH NHLOW NWID

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 26

Data List Label (25 char.max): LABELS
Owner Directory Code: 5 STATISTICS DL Id: 012
DL Type: ___IDL/R C DIM = _____ ___RDL/R C DIM = ___ X CDL/max char=4
Page 1 of 1
Rev.Date: 28 NOV 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		The following MSAINT label arrays are stored in this character data list. LLUGC LLUGT LLUGH LLRES LLTSK NTRID

8-0 THE "TACTICAL SCENARIO" DIRECTORY (CODE 6)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	6|TACTICAL-SCENARIO
DIRECTORY LEVEL:	2
OWNER DIRECTORY(CODE):	1-SIMULATION-DATA-SET
ID IN OWNER DIRECTORY:	5|(sequence number)
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	2
RANKING CODE FOR OWNED DIRECTORIES:	0
RANKING CODE FOR OWNED DATA LISTS:	0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 6

DIRECTORY LABEL: TACTICAL-SCENARIOS

TYPE	CODE	LABEL	NUMBER
DL	-	RUN-DATA	1
DR	5	STATISTICS	1
DL	-	TITLE	1

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 20

Data List Label (25 char.max): RUN-DATA
Page 1 of 1
Owner Directory Code: 6 | TACTICAL-SCENARIO DL Id: 1 | 0
Rev.Date: 7 JULY 1983
DL Type: ____IDL/R C DIM = _____ X RDL/R C DIM = 1 ____CDL/max char= ____
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE	
1	CRITICAL-ASSET-CONFIG	Critical Asset Configuration Number	(D) = 0
2	AIR-SCENARIO	Air Scenario Number	(D) = 0
3	NUMBER-OF-RUNS	Number of simulation runs to perform	(D) = 1
4	START-TIME	Simulation start time (military time)	(D) = 0
5	END-TIME	Simulation end time (military time)	(D) = 2359
6	TRACE-START-RUN	Run to begin event trace	(D) = 0
7	TRACE-END-RUN	Run to end event trace	(D) = 0
8	TRACK-UPDATE-INTERVAL	Maximum time between updates of track position (seconds)	(D) = 15
9	RANDOM-NUMBER-SEED	Seed for the random number generator	(D) = 0
10	NUM-ASSETS	Number of critical assets	(D) = 0
11	critical asset label	ACN of working ADSM that is protecting this site. Repeat 11 for each critical asset.	

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 21

Data List Label (25 char.max): TITLE
Owner Directory Code: 6 | TACTICAL-SCENARIO DL ID: 2 | 0
DL Type: IDL/R C DIM = RDL/R C DIM = Y CDL/max char= 40

Page 1 of 1
Rev.Date: 6 JULY 1983
Author: POLITO

ELEMENT NO.	ELEMENT LABEL (25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1		First row of Title information for the Tactical Scenario.
2		Second row of Title information for the Tactical Scenario.
3		Etc.

9-0 THE "COMMAND-AND-CONTROL" DIRECTORY (CODE 7)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	7|COMMAND-AND-CONTROL
DIRECTORY LEVEL:	2
OWNER DIRECTORY(CODE):	1|SIMULATION-DATA-SET
ID IN OWNER DIRECTORY:	7|1
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	4
RANKING CODE FOR OWNED DIRECTORIES:	4
RANKING CODE FOR OWNED DATA LISTS:	2

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 7

DIRECTORY LABEL: COMMAND-AND-CONTROL

TYPE	CODE	LABEL	NUMBER
DR	9	MESSAGES	1
DR	12	working ADSM'S (assigned by MOPADS)	as needed

10-0 THE AIR SCENARIO DIRECTORY (CODE 8)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	8|user specified (AIR SCENARIO)
DIRECTORY LEVEL:	3
OWNER DIRECTORY(CODE):	13|CRITICAL-ASSET-CONFIGURATION
ID IN OWNER DIRECTORY:	8|(user assigned number)
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	1
RANKING CODE FOR OWNED DIRECTORIES:	0
RANKING CODE FOR OWNED DATA LISTS:	1

DESCRIPTION:

MCPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 8

DIRECTORY LABEL: user specified (Air Scenario)

TYPE	CODE	LABEL	NUMBER
DL	-	AIRCRAFT-TRACKS-HOSTILE	1
DL	-	AIRCRAFT-TRACKS-FRIENDLY	1
DL	-	AIRCRAFT-TRACKS-OTHER	1

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	S-17/DLD - 13

Data List Label (25 char.max):AIRCRAFT-TRACKS-HOSTILE(FRIENDLY)(OTHER) — Page 1 of 2
Owner Directory Code: B (user determined) DL Id: 1(2)(3) — Rev.Date:10 DEC 82
DL Type: IDL/R C DIM = X RDL/(R) C DIM =9 CDL/max char= — Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C1		0-implies initiation row of a new track, 1-implies segment row of a track
	If C1 = 0	
C2		Identifying number for the aircraft
C3		Initiation time of the track from the simulation start time (minutes)
C4		Actual primary ID (1-hostile,2-friendly,3-other)
C5		Multiplirity
C6		Aircraft type (see attached)
C7		x - n.miles } initial point relative to the coordinate
C8		y - n.miles } initial point relative to the coordinate
C9		z - ft. } reference point
	If C1 = 1	
C2		x - n.miles } end point of flight segment relative to coordinate reference point
C3		B - n.miles } end point of flight segment relative to coordinate reference point
C4		z - ft. } end point of flight segment relative to coordinate reference point
C5		speed (knots) on this segment
C6		is the end point a target (0-no, 1-yes)(used only if hostile)
C7		is the aircraft jamming on this segment (0-no,1-yes)
C8		Probability of destroying target if C6=1
C9		0 not used

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 13
Data List Label (25 char.max):AIRCRAFT-TRACKS-HOSTILE(FRIENDLY)(OTHER)		Page 2 of 2
Owner Directory Code: B (user determined)	DL Id: 1(2)(3)	Rev.Date:10 DEC 82
DL Type:___IDL/R C DIM =_____ X RDL(R)C DIM = 9 ___CDL/max char=_____		Author:POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		NOTE: 1. All rows for a track are in chronological and geographical order. The first row is the "initiation row" (column 1 = 0). Each subsequent row is a "segment row" (column 1 = 1). 2. Tracks must be ordered in chronological order of their initiation times.

-0 THE "MESSAGES" DIRECTORY (CODE 9)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	9|MESSAGES
DIRECTORY LEVEL:	4
OWNER DIRECTORY(CODE):	7|COMMAND-AND-CONTROL
ID IN OWNER DIRECTORY:	9|0|0|0
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	5
RANKING CODE FOR OWNED DIRECTORIES:	0
RANKING CODE FOR OWNED DATA LISTS:	0

DESCRIPTION:

MOPARS/DIRECTORY CONTENTS

DIRECTORY CODE: 9

DIRECTORY LABEL: MESSAGES

TYPE	CODE	LABEL	NUMBER
DL	-	MESSAGES-DL	as needed

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 12

Data List Label (25 char.max): MESSAGES-DL
Owner Directory Code: 9 | MESSAGES Id: CRN/OI/FI/SB/MP
DL Type: IDL/R C DIM = X RDL/R C DIM =1 CDL/max char=
Page 1of 2
Rev.Date:2 JUN 83
Author: POLIIU

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		ID elements have the following meaning. CRN - Copy Row Number of receiver OI - Operator Type of receiver (zero if n/a) FI - Functional Type 1 - Commands 2 - Request for Information 3 - Reporting Status 4 - Acknowledgement 5 - Computer Generated SB - Message subtype - determined by application MP - 1000 times the message priority (negative implies interrupting message)
1		1 - Voice, 2 - ATDL
2		Acknowledgement Required, 1 - Yes, 2 - No
3		UNUSED
4		ATDL message code (currently unused - always zero)
5		Time message was sent.
6		Message number (sequentially assigned by MOPADS)
7		Sender Copy Row Number
8		Sender Operator Type
9		System module type of the sender
10		SAINT Task node number message is sent from

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 12

Data List Label (25 char.max): MESSAGES-DL
Owner Directory Code: 9 | MESSAGES Id: CRN/OI/FI/SB/MP
DL Type: IDL/R C DIM = X RDL/R C DIM = CDL/max char=
Page 2 of 2
Rev.Date: 2 JUNE 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
11 12 . . K+11		K = number of subsequent words Words 12 to K+11 are dependent upon the specific message

12-0 THE "ACKNOWLEDGEMENT-MESSAGES" DIRECTORY (CODE 10)

MOPADS/DATA BASE **DIRECTORY DESCRIPTION**

DIRECTORY LABEL(CODE/LABEL): 10|ACKNOWLEDGEMENT-MESSAGES

DIRECTORY LEVEL: various

OWNER DIRECTORY(CODE): 12

ID IN OWNER DIRECTORY: 10|1|0|ACN

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 5

RANKING CODE FOR OWNED DIRECTORIES: 0

RANKING CODE FOR OWNED DATA LISTS: 0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 10

DIRECTORY LABEL: ACKNOWLEDGEMENT-MESSAGES

TYPE	CODE	LABEL	NUMBER
DL	-	ACK-MESSAGES-DL	as needed

MOPADS/DATA BASE: **DATA LIST DESCRIPTION** 5-17/DLB - 19

Data List Label (25 char.max): ACK-MESSAGES-DL Page1 of 2
Owner Directory Code: 10 | ACKNOWLEDGEMENT-MESSAGES Id: CRN/OT/FT/SB/MP Rev.Date:25 JULY 83
DL Type:____IDL/R C DIM =_____ X RDL/R C DIM = 1 ____CDL/max char=_____Author:POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		ID elements have the following meaning CRN - Copy Row Number of receiver OT - Operator Type of Receiver (zero if n/a) FT - Functional Type 1 - Commands 2 - Request for Information 3 - Reporting Status 4 - Acknowledgement 5 - Computer Generated SB - Message Subtype - determined by application MP - 1000 times the Message Priority
1		1 - Voice, 2 - ATDL
2		Acknowledgement Required, 1 - Yes, 2- No
3		Time message was received
4		ATDL message code (currently unused - always zero
5		Time message was sent
6		Message number (sequentially assigned by MOPADS)
7		Sender Copy Row Number
8		Sender Operator Type
9		System module type of the sender
10		SAINT Task node number message is sent from

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/BLB - 19

Data List Label (25 char.max): ACK-MESSAGES-DL — Page 2 of 2
Owner Directory Code: 10 | ACKNOWLEDGEMENT-MESSAGES — Id: CRN/QT/FT/SB/MP — Rev.Date: 25 JULY 83
DL Type: ____IDL/R C DIM = _____ X RDL/R C DIM = 1 ____CDL/max char= _____ — Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
11 12 . . K+11		K = number of subsequent words Words 12 to K+11 are dependent upon the specific message

13-0 THE REFERENCE SYSTEM MODULE DIRECTORY (CODE 11)

MOPADS/DATA BASE DIRECTORY DESCRIPTION

DIRECTORY LABEL(CODE/LABEL): 11|*

DIRECTORY LEVEL: 2

OWNER DIRECTORY(CODE): 2|REFERENCE-ADSM

ID IN OWNER DIRECTORY: 11|NECC(ACC)|0|basic ACN

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 4

RANKING CODE FOR OWNED DIRECTORIES: 0

RANKING CODE FOR OWNED DATA LISTS: 2

DESCRIPTION:

* The label is a one-character ACC followed by a dash (-) followed by a user specified label; e.g., Q-BATTALION-Q73.

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 11

DIRECTORY LABEL: user specified

TYPE	CODE	LABEL	NUMBER
DL	-	TD-TASK-DATA	1
DL	-	R-ADS-RESOURCES	1
DL	-	RL-RESOURCE-LABELS	1
DL	-	OP-OPERATOR-STATE-VECTOR*	as needed
DL	-	EN-ENVIRONMENTAL-STATE-VEC*	1
DL	-	OT-OPERATOR-TYPE	1

* These DL's have 2 rows. The second row holds initial values. The second row is copied to the first row prior to each simulation run.

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 03

Data List Label (25 char.max): _ID-TASK-DATA_ Page 1 of 1
Owner Directory Code: 1i DL Id: NECC(ACC)|NECC('ID')|1|0 Rev.Date:25 MAR 83
DL Type:___IDL/R C DIM =_____ _X_RDL(R) C DIM=56 ___CDL/max char=_____ Author:Polito___

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		Each row corresponds to a MSAINT TASK node whose number is the same as the row number
C 1	DISTRIBUTION-TYPE	distribution type (D) = 8
C 2	MEAN	Mean of the nominal task time (D) = 0
C 3	STANDARD-DEVIATION	Standard Deviation of the nominal task time (D) = 0
C 4	POINTER-TO-SKILLS	First column number containing skills information (D) = 18
C 5	POINTER-TO-RESOURCES	First column no. containing ADSM resource rqmts. (D) = J
C 6	UNUSED	
C 7	KILOCALORIES/MIN	Kilocalories/minute (D) = 1
C 8	NUMBER-OF-BRANCHES-OUT	Number of Branches out (D) = 1
C 9	STIMULUS-MODE-1	Stimulus Mode 1 (D) = 10
C10	STIMULUS-MODE-2	Stimulus Mode 2 (D) = 0
C11	RESPONSE-MODE	Response Mode (D) = 1
C12	OBSRVR-TARGET-POSITION	Observer Target Position (D) = 3
C13	CONTROL-DISTANCE	Control Distance (ft) (D) = 1
C14	CONTROL-WIDTH	Control Width (ft) (D) =0.1
C15	NUMBER-OF-DISPLAYS	Number of Displays (D) = 1
C16	NUMBER-OF-ALTERNATIVES	Number of Alternatives (D) = 1
C17	NUM-STM-ITEMS	No. of items that must be kept in short term memory (D) = 1
C18	SKILL-INDEX	Code Number of a required skill category (see attached)
C19	SKILL-WEIGHT	Weight of the above skill (repeats of C18 & C19 for each required skill)
CJ	RESOURCE-INDEX	Index of a hardware resource required
CJ+1	RESOURCE-PARAMETER	Resource parameter for the above resource (repeats of CJ & CJ+1 till a resource index is zero)

Column	Description	Default
7	Kilocalories/minute	1
8	Number of branches out	1
9	Stimulus Mode 1 A two-digit number 10's digit - 0 - no visual 1 - visual 1's digit - 0 - no auditory 1 - auditory	10
10	Stimulus Mode 2 A three-digit number as above 100's digit- 0 - no olfactory 1 - olfactory 10's digit - 0 - no kinesthetic 1 - kinesthetic 1's digit - 0 - no tactile 1 - tactile	00
11	Response Mode A two-digit number 10's digit - 0 - no vocal 1 - vocal 1's digit - 0 - no tactile 1 - tactile	1
12	Observer - Target Position 1 - ground to ground 2 - air to ground 3 - ground to air 4 - air to air 5 - at a display	3
13	Control Distance (distance from operator to control device in feet)	1
14	Control Width (width of the control device in feet)	0.1
15	Number of Displays	1
16	Number of Alternatives (number of possible controls that may need to be actuated	1
17	Number of items that must be kept in short term memory	1

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 01

Data List Label (25 char.max) : R-ADS-RESOURCES Page 1 of 1
Owner Directory Code: 11 DL Id: NECC(ACC)|18|1|0 Rev.Date: 14 JUL 82
DL Type: IDL/R C DIM = X RDL (R) C DIM = 7 CDL/max char= Author: Polito

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C 1	NUMBER-OF-UNITS	The number of units of the resource (D) = 1
C 2	NUMBER-BUSY	The number of units busy (I) = 0
C 3	NUMBER-DOWN	The number of units inoperable (I) = 0
C 4	MTBF	Mean Time Between Failures (minutes) (D) = 1.E20
C 5	MTR	Mean Time To Repair (minutes) (D) = 0.
C 6	USE-CODE	0 - Non-exclusive use. If the resource is available, it can be used by all who need it
		1 - Use by operators makes units of the resource unavailable for use by others (D) = 1
C 7	EQUIPMENT-STATUS	1 - Green, 2 - amber, 3 - red (I) = 1
		NOTE: This data list is included for future growth of MOPADS. The current models do not use this information.

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 02

Data List Label (25 char.max): RL-RESOURCE-LABELS Page 1 of 1
Owner Directory Code: 11 DL Id: MECC(ACC)|1|9|2|1|1|9 Rev.Date: 14 JUL 82
DL Type: ___ IDL/R C DIM = _____ ___ RDL/R C DIM = ___ X CDL/max char= 25 Authors: Polito

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1		label of ADS resource number 1
2		label of ADS resource number 2
		etc.
		NOTE: This data list is included for future growth of MOPADS. The current models do not use this information.

MOPADS/DATA BASE: DATA LIST DESCRIPTION S-17/DLD - 37

Data List Label (25 char.max): OP-OPERATOR-STATE-VECTOR Page 1 of 2

Owner Directory Code: 11 DL Id: NECC(ACC)1151610 Rev.Date: 23 MAY 1983

DL Type: IDL/R C DIM = X RDL/(R)t DIM = VAR CBL/max char= Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C 1	OPERATOR-TYPE	Type of operator for this ADS (1,2,etc)
C 2	OPERATOR-ID	Unique ID assigned at run time by MSAINT
C 3	POINTER-TO-HF-TASK-DATA	Index of first element of Human Factors Task Data (D) = 32
C 4	POINTER-TO-GOAL-PARAMETERS	Index of first element of Goal Parameter Data (D) = K
C 5	POINTER-TO-OBJ-FUNCTION	Index of first element of Objective Function Parameters (D) = J
C 6	POINTER-TO-DISPLAY-INFO	Index of first element of Display info for operator (D) = M
C 7	POINTER-TO-TASK-STACK	Index of first element of Task Stack information (D) = L
C 8	UNUSED	
C 9	UNUSED	
C10	UNUSED	
C11	UNUSED	
C12	UNUSED	
C13	UNUSED	
C14	UNUSED	
C15	UNUSED	
C16	UNUSED	
C17	UNUSED	
C18	UNUSED	
C19	UNUSED	
C20	UNUSED	
C21	UNUSED	

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 07

Data List Label (25 char.max): OP-OPERATOR-STATE-VECTOR
Page 2 of 7
Owner Directory Code: 11
DL Id: NECC(ACC)1516in10
Rev.Date: 23 MAY 1983
DL Type: ___IDL/R C DIM =_____ X RDL (R) C DIM =VAR ___CBL/max char=_____
Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE	
C22	UNUSED		
C23	UNUSED		
C24	UNUSED		
C25	UNUSED		
C26	UNUSED		
C27	UNUSED		
C28	UNUSED		
C29	UNUSED		
C30	UNUSED		
C31	UNUSED		
C32	CORE-TEMPERATURE	Core body temperature, C degree	(D) = 37
C33	CIO-VALUE	Measure of clothing heat retention	(D) = 1
C34	TIME-ON-TASK	Total hours spent performing the task today	(I) = 0
C35	DAYS-OF-DUTY	Number of sequential days of task performance	(D) = 0
C36	SEARCH-DIMENSIONS	Number of dimensions by which objects can be discriminated	(D) = 1
C37	NUMBER-FIRE-UNITS	Number of Fire units operator is responsible for	(D) = 0
C38	PERCENTAGE-RECOVERY	Function of Previous Work and Previous Rest	(D) = 100
C39	PREVIOUS-WORK	Hours worked in the previous work session	(D) = 0
C40	PREVIOUS-REST	Hours rest between previous and current work sessions	(D) = 16
C41	FLASH-INTENSITY	Measure of Flash intensity (Foot Lambert seconds)	(D) =
C42	TARGET-SPEED	Velocity of target being observed,knots	

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLB - 07
Data List Label (25 char.max): OP-OPERATOR-STATE-VECTOR		Page 3 of 7
Owner Directory Code: 11	DL Id: NECC(ACC)11516100	Rev.Date: 23 MAY 1983
DL Type: ___IDL/R C DIM = _____	X RDL/(R) C DIM = VAR ____CDL/max char= ____	Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C43	TARGET-TYPE	See Attached
C44	TARGET-SIZE	Determined by target type
C45	TARGET-COLOR	Determined by target type (see attached)
C46	SEARCH-AREA	Degrees search area operator examines (D) = 360
C47	BINOCULAR-USAGE	Human is (value = 1) or is not (value = 0) using binoculars (D) = 0
C48	SLANT-RANGE-TO-TARGET	Distance to target (nau.miles)
C49	TARGET-TRAJECTORY	Travel direction of target (degrees relative to North)
C50	TARGET-BAKGRND-COMPLEXITY	Depends upon height of target,terrain, and weather (Look up table) (D) = 1
C51	NUM-BACKGROUND-CHARACTERS	Total background characters on the display(clutter)
C52	MESSAGE-BACKLOG	Number of Messages waiting for the operator
C53	SIGNALS-PER-MINUTE	Number of signals operator per minute (D) = 1
C54	HOURS-WORKED-PER-WEEK	Number of hours worked during the past week (D) = 40
C55	DAYS-WITHOUT-SLEEP	Consecutive days without sleep (D) = 0
C56	DAYS-OF-NIGHT-DUTY	Consecutive days of working between 2200 & 0800 (D) = 0
C57	SIMULTANEOUS-TASKS	Number of tasks performed at once (D) = 1
C58	CONTRAST-RATIO	Target/background ratios from table of target type, sky conditions, terrain type, altitude (D) = 1
C59	AVE-HOURS-SLEEP	Ave hours sleep/night (D) = 8

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 07

Data List Label (25 char.max):_OP-OPERATOR-STATE-VECTOR_ Page 4 of 7

Owner Directory Code:11_ DL Id:_NECC(ACC)(1516)n(0_____ Rev.Date:23 MAY 1983

DL Type:____IDL/R C DIM =____ X RBL (R) C DIM =VAR ____CDL/max char=___ Author:Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C60	OBJECTIVE-FUNCTION	Same as objective function parameters (D) = 1
C61	GOALS-CONSIDERED	Number of goals considered in Task Sequencing (D) = 1
C62	TARGET-BRIGHTNESS	Target brightness - foot lamberts (D) = 1
C63	NIGHTS	Number of consecutive nights with known hrs of sleep (D) = 2
C64	SKY-GROUND-RATIO	Sky-ground-luminence ratio foot lamberts (D) = 1
C65	AIRCRAFT-ALTITUDE	Absolute altitude above the ground - ft.
C66	METEOROLOGICAL-RANGE	Visibility - nau.miles (D) = 4
C67	THRESHOLD-CONTRAST	Threshold object/background contrast (D) = 1
C68	TARGET-HEIGHT	Vertical dimension of target, ft.
C69	TARGET-WIDTH	Target width, ft.(front)
C70	TARGET-DEPTH	Depth of target, ft.
C71	HORIZONTAL-RANGE	Horizontal range - nau.miles
C72	NUM-OF-RESOLUTION-ELEM	Number of linear resolution elements to resolve the target (look up table) (D) = 1
C73	NUM-OF-SUSPECT-AREAS	Number of areas where target may show up (look up table) (D) = 1
C74	AIRCRAFT-SPEED	Speed of airplane-knots
C75	FIELD-OF-VIEW	Field of view - degrees (D) = 360
C76	OBSERVER-OFFSET	Perpendicular distance to the aircraft trajectory - nau. miles
C77	UNUSED	
C78	DISPLAY-TARGET-LOCATION	3 by 3 matrix of screen location,values=1,2,3...9 (D) = 5
C79	TARGET-LOCATION	Target Displacement from the observer in degrees (negative = left, + = right)

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - C7
Data List Label (25 char.max): OP-OPERATOR-STATE-VECTOR		Page 5 of 7
Owner Directory Code: 11	DL Id: NECC(ACC)1151610	Rev.Date: 23 MAY 1983
DL Type: ___IDL/R C DIM = _____	X RDL/R C DIM = VAR ___CDL/max char= ____	Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE	
C80	DISPLAY-RESOLUTION	Resolution of CRT display (pixels/10 in), 240, 480, 800, etc.	(D) = 240
C81	DISPLAY-BACKGROUND-HEIGHT	Height of non-target display background,feet	(D) = 0.1
C82	DISPLAY-BACKGROUND-WIDTH	Width of non-target display background	(D) = 0.1
C83	DISPLAY-BACKGROUND-DEPTH	Depth of non-target display background (ft)	(D) = 0.1
C84	DISTANCE-TO-DISPLAY	Distance to display (ft)	(D) = 1.33
C85	DISPLAY-HEIGHT	Height of display (ft)	(D) = 1
C86	DISPLAY-WIDTH	Width of display (ft)	(D) = 1
C87	TARGET-NOISE-LEVEL	Target noise level at the operator's proximity	(D) = 60
C88	TARGET-DURATION	Time since target appeared on screen (minutes)	
C89	EXPERIENCE	# times operator performed this task (ever)	(D) = 20
C90	SIGNAL-PROBABILITY	Probability of a signal occurring. Subjective expectation.	(D) = 0
C91	REST-PERIODS	# of rest periods for work shift	(D) = 1
C92	TASK-ERROR-FACTOR	Factor for computing task success probabilities (between 1 and -INF)	(D) = 0
C93	TASK-ELEMENT-ERROR-FACTOR	Factor for computing task element success probabilities (between 1 and -INF)	(D) = 0
C94	DAYS-SINCE-PRACTICE	Days since last practiced the task	(D) = 0
C95	SENSE-OF-DIRECTION	0 - poor, 1 - good	(D) = 1
C96	SKIN-TEMPERATURE	Temperature at skin (Degrees C)	(D) = 20
C97	TIME-IN-TEMPERATURE	Minutes in current ambient temperature	(D) = 240

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 07

Data List Label (25 char.max): OP-OPERATOR-STATE-VECTOR — Page 6 of 7
Owner Directory Code: 11 — DL Id: NECC(ACC)15161n10 — Rev.Date: 23 MAY 1983
DL Type: ___IDL/R C DIM =_____ — X RDL/(R) C DIM =VAR ____CDL/max char=_____ — Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C98	PREVIOUS-SKIN-TEMPERATURE	Skin temperature prior to changing to current ambient temperature (D) = 20
C99	X-SCREEN-CENTER	(x) center of screen viewing area (D) = -9999
C100	Y-SCREEN-CENTER	(y) center of screen viewing area (D) = -9999
C101	SCREEN-RANGE	Range the operator can see on his screen (mau.mi) (D) = -9999
K	GOAL	Goal number (negative if operator does not have this goal)
K+1	LITTLE-M	Minimum satisfaction level
K+2	BIG-M	Maximum satisfaction level
K+3	LITTLE-A	Multiplier for downside goal priority function
K+4	LITTLE-B	Exponent for downside goal priority function
K+5	BIG-A	Multiplier for upside goal priority function
K+6	BIG-B	Exponent for upside goal priority function
K+7	GOAL-STATE-LOW-1	
K+8	PRIORITY-LOW-1	Two points on downside priority function from which LITTLE-A and LITTLE-B are calculated
K+9	GOAL-STATE-LOW-2	
K+10	PRIORITY-LOW-2	
K+11	GOAL-STATE-HIGH-1	
K+12	PRIORITY-HIGH-1	Two points on upside priority function from which BIG-A and BIG-B are calculated
K+13	GOAL-STATE-HIGH-2	
K+14	PRIORITY-HIGH-2	Repeats of K through K+14

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 07
Data List Label (25 char.max):OP-OPERATOR-STATE-VECTOR		Page 2 of 2
Owner Directory Code: 11	DL Id: NECC(ACC)1516[n]0	Rev.Date:23 MAY 1983
DL Type:____IBL/R C DIM =_____	____RDL/(R)C DIM =VAR ____CBL/max char=____	Author:Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
J	OBJECTIVE-FUNCTION	Objective function option 1-(maximize the reduction of the average of the N largest goal priorities), 2- (max reduction/time)(same as C60) (D) = 1
J+1	NUMBER-OF-GOALS	N for objective function (same as C61)
M	Display information	The number of words and their definitions are dependent upon the system module.
L		Task Stack Information (this is at end of the OSV). Points to next element of the stack to be performed. Zero if no stack or stack complete.
L+1		Task number. Negative if the task was previously interrupted.
L+2 to L+5		Four values whose meaning is dependent upon the task.
		Repeats of L+1 to L+5.

TARGET TYPE CODES

CODE NO.	NAME	COUNTRY	MISSION
1	F4C	USA	
2	F100	USA	
3	T33	USA	
4	OTHER JET	USA	
5	U1A	USA	
6	U6A	USA	
7	OTHER PROP	USA	
8	O1A	USA	
9	OH23	USA	
10	OTHER HELICOPTER	USA	
11	TANK	USA	
12	JEEP	USA	
13	TROOP	USA	
14	APC	USA	
15	TRUCK	USA	
16	ZERO	USA	
17	HALFTRACK	USA	
18	F14	USA	
19	F15	USA	
20	F16	USA	
21	F18	USA	
22	MIG21	USA	
23	MIG23	USA	
24	MIG25	USA	
25	SOLDIER(FOOT)	ANY	
26	MIG-27	USSR	
27	SU-17	USSR	
28	QIANG Ji-5	PRC	Ground Attack
29	R-2358	FRANCE	Military surveillance
30	MIRAGE 3E	FRANCE	Fighter
31	MIRAGE F1	FRANCE	Fighter
32	MIRAGE 2000	FRANCE	Fighter
33	MIRAGE 4000	FRANCE	Fighter
34	MIRAGE 4	FRANCE	Bomber
35	MIRAGE 5	FRANCE	Ground Support
36	MIRAGE 50	FRANCE	Fighter
37	AV.660	GREAT BRITAIN	Military Transport
38	698	GREAT BRITAIN	Bomber
39	HS748	GREAT BRITAIN	Military Transport
40	HS.780	GREAT BRITAIN	Military Transport
41	P.1099	GREAT BRITAIN	Ground Attack

42	IAI202	ISRAEL	Military Transport
43	MIRAGE 3C	ISRAEL	Fighter
44	KfirC2	ISRAEL	Fighter
45	G.222	ITALY	Military Transport
46	F1045	ITALY	Interceptor
47	MB.326K	ITALY	Strike
48	S.M.1019E	ITALY	Military STOL
49	F-1	JAPAN	Fighter
50	C.207A	SPAIN	Military Transport
51	SF-5A	SPAIN	Fighter
52	HA-220	SPAIN	Ground Attack
53	35XD	SWEDEN	Ground Attack
54	JA37	SWEDEN	Fighter
55	J-1	YUGOSLAVIA	Strike
56	Light (e.g.blinking)		
57	Digit (digit on a display)		
58	MIG-17	USSR	
59	MIG-19	USSR	
60	SU-7	USSR	
61	SU-9PH	USSR	
62	SU-11	USSR	
63	SU-15	USSR	
64	SU-19	USSR	
65	SU-20	USSR	
66	YAK-28P	USSR	
67	YAK-36	USSR	
68	TU-28P	USSR	
69	IL-28	USSR	
70	M-4	USSR	
71	TU-16	USSR	
72	TU-20	USSR	
73	TU-22	USSR	
74	TU-26	USSR	
75	IL-38	USSR	
76	TU-126	USSR	
77	MIG-15	USSR	
78	L-29	USSR	
79	L-39	USSR	
80	J-5	USSR	
81	J-6	USSR	
82	J-7	USSR	
83	J-8	USSR	
84	H-5	USSR	
85	H-6	USSR	
86	CJ-6	USSR	
87	Y-11	USSR	
88	MIRAGE III	FRANCE	

COLOR CODES

1	White
2	Tan
3	Green
4	Blue
5	Red
6	Yellow

MISSION CODES

1	Ground attack
2	Military surveillance
3	Fighter
4	Bomber
5	Ground support
6	Military Transport
7	Interceptor
8	STOL
9	Strike

COUNTRY CODES

1	USA	7	Italy
2	USSR	8	Japan
3	PRC	9	Spain
4	France	10	Sweden
6	Israel	11	Yugoslavia

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 10

Data List Label (25 char.max): EN-ENVIRONMENT-STATE-VEC Page 1 of 2
Owner Directory Code: 11 DL Id: NECC(ACC)|514|n|0 Rev.Date: 04 AUG 1983
DL Type: ___IDL/R C DIM = _____ X RDL/(R) C DIM = 31 ____CDL/max char= ____ Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE	
C 1	POINTER TO HF DATA	Index of first word of Human Factors Data	(D) = 23
C 2	SYSTEM-MODE	NOT USED	(D) = 0
C 3	OPERATOR-MODE	1-Task Sharing,2-Not Sharing,etc.(NOT USED)	(I) = 0
C 4	UNUSED		
C 5	METHOD-OF-CONTROL	0-NA 1-Centralized, 2-Decentralized, 3-Independent, 4-Autonomous (NOT USED)	(D) = 1
C 6	WEAPONS-CONTROL-STATUS	0-NA 1-Weapons free 2-Weapons Tight, 3-Weapons Hold	(D) = 2
C 7	UNUSED		
C 8	INITIAL-AMMUNITION-HOT	Hot and Cold Ammunition at each Fire Section	
C 9	INITIAL-AMMUNITION-COLD		
C10	UNUSED		
C11	UNUSED		
C12	UNUSED		
C13	UNUSED		
C14	UNUSED		
C15	UNUSED		
C16	UNUSED		
C17	UNUSED		
C18	UNUSED		

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 10
Data List Label (25 char.max): EN-ENVIRONMENT-STATE-VEC		Page 2 of 2
Owner Directory Code: 11	DL Id: NECC(ACC)|514|n|0	Rev.Date: 04 AUG 1983
DL Type: ___IDL/R C DIM =_____	X RBL/(R) C DIM =31 ___CDL/max char=_____	Author: Polito/Laughery

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C19	X-POSITION	x-coordinate from reference point (n.miles)
C20	Y-POSITION	y-coordinate from reference point (n.miles)
C21	Z-POSITION	z-coordinate from reference point (ft)
C22	SAMPLING-OPTION	Task Element Time Sampling Option (D) = 3 1-unmoderated deterministic, 2-unmoderated stochastic, 3-moderated deterministic, 4-moderated stochastic
C23	DRY BULB TEMPERATURE	Environmental Temp.degree C (D) = 22
C24	RELATIVE HUMIDITY	% Humidity (D) = 50
C25	AIR MOVEMENT RATE	Mean Wind Velocity,mi/hr (D) = 6
C26	NOISE LEVEL	Background Noise in dBA (D) = 45
C27	WORK AREA ILLUMINATION	Work area luminance in Ft.Lamberts (D) = 50
C28	NUMBER-ON-DUTY	Number of operators in close proximity (D) = 1
C29	VIBRATION	Vertical Movement in Hz (D) = 0
C30	AMBIENT-VAPOR-PRESSURE	Vapor pressure in mb (D) = 10
C31	NOISE-PREDICTABILITY	0 - Consistent, 1 - Inconsistent (D) = 0

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	S-17/DLD - 11
Data List Label (25 char.max): QI-OPERATOR-TYPE Owner Directory Code: 11 DL Id: NECC(ACC)1520110 DL Type: ____ IDL/R C BIH = ____ ____ RDL/R C BIH = ____ X_CDL/max char= 25		Page 1 of 1 Rev.Date: 19 NOV 1982 Author: Colito

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1	Task node number where created	Name of operator type. Each operator in an ADSM must have a separate type. The index of the operator type is the same number as the 1st column of the IB of the operater state vector.
2		
etc.		

14-0 THE WORKING SYSTEM MODULE DIRECTORY (CODE 12)

MOPADS/DATA BASE	DIRECTORY DESCRIPTION
DIRECTORY LABEL(CODE/LABEL):	12|determined by MOPADS*
DIRECTORY LEVEL:	various
OWNER DIRECTORY(CODE):	7 or 12 various
ID IN OWNER DIRECTORY:	12|NECC(ACC)|sequence number|ACN
NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS:	4
RANKING CODE FOR OWNED DIRECTORIES:	4
RANKING CODE FOR OWNED DATA LISTS:	2

DESCRIPTION:

* See MOPADS Volume 3.3

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 12

DIRECTORY LABEL: determined by MOPADS

TYPE	CODE	LABEL	NUMBER
DL	-	TD-TASK-DATA*	1
DL	-	R-ADS-RESOURCES*	1
DL	-	RL-RESOURCE-LABELS*	1
DL	-	OP-OPERATOR-STATE-VECTOR*	as needed
DL	-	EN-ENVIRONMENTAL-STATE-VEC*	1
DL	-	OT-OPERATOR-TYPE*	1
DL	-	working ADSM label-FV **	1
DL	-	working ADSM label-TRD***	1
DR	10	ACKNOWLEDGEMENT-MESSAGES	1
DR	12	other working ADSM's	as needed

* These data lists are identical to those in Code 11 directories. Their descriptions are not repeated here.

** Field of View

***Track Data

MOPADS/DATA BASE: **DATA LIST DESCRIPTION** 5-17/DLD - 18

Data List Label (25 char.max): owner directory label-FV
Owner Directory Code: 12
DL Id: NECC(ACC)622 1 ACN
DL Type: ___IDL/R C DIM = _____ X RDL/® C DIM = 81 ____CDL/max char= _____
Page 1 of 2
Rev.Date: 21 JUN 83
Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C1	VIEWER-STATUS	Viewer status (1-viewing,2-not viewing, 0-unused row (D) = 1
C2	RANGE	Range (n.mi) (D) = 25
C3	MIN-ALTITUDE	Minimum altitude (above radar) to acquire target (feet) (D) = 0
C4	MAX-ALTITUDE	Maximum altitude (above radar) to acquire target (feet) (D) = 25000
C5	PROBABILITY-OF-DETECTION	Probability a view acquires a target (D) = .9
C6	VIEWER-X	x-position of viewer (n.mi) (D) = 0
C7	VIEWER-Y	y-position of viewer (n.mi) (D) = 0
C8	VIEWER-Z	z-position of viewer (feet) (D) = 0
C9	SECTOR-START	C9 and C10 are the start and end azimuth (degrees) (D) = 0
C10	SECTOR-END	This information may not be relevant for all viewers. C9 must be less than C10. The sector is from C9 to C10. (D) = 360
C11	NUMBER-OF-BARRIERS	Number of barriers (10 max) (D) = 0
C12	BARRIER-TYPE	1 - line barrier, 2 - wedge (0-UNUSED block of 7 columns)
C13	R1	r1 - distance from viewer to first point of barrier (n.mi),(unused for wedge barriers)

MOPADS/DATA BASE: **DATA LIST DESCRIPTION** 5-17/DLD - 18

Data List Label (25 char.max):owner directory label-FV
Owner Directory Code: 12
DL Id: NECC(ACC)1622111ACN
DL Type: IDL/R C DIM = X RDL/R C DIM = 81 CDL/max char=
Page2 of 2
Rev.Date:21 JUN 83
Author:POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C14	THETA1	theta1 - true bearing from north to the first point of a line barrier (degrees).True bearing of first edge for a wedge barrier.
C15	H1	h1 - height above or below z-position of viewer for the first point of the barrier (ft) (unused for wedge barriers).
C16	R2	R2,THETA2 and H2 are analogous to R1,THETA1 and H1 for point 2 of the barrier.
C17	THETA2	
C18	H2	
		NOTE: For wedge barriers the barrier is from azimuth THETA1 to THETA2.
		REPEAT C12 - C18 FOR EACH BARRIER.

NOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 22
Data List Label (25 char.max): Owner Directory Label-IRD		Page 1 of 2
Owner Directory Code: 121	DL Id: NECC(ACC(NECC('IRD')1010	Rev.Date: 26 MAY 1983
DL Type: ___IDL/R C DIM =_____	_X_RDL/® C DIM =13 ___CDL/max char=____	Author: POLIIO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
		Track Data for the ADSM. One row for each track.
C1	POINTER-TO-NTRACK	Column in NTRACK array for the track. Zero means is unused.
C2	LOCAL/REMOTE	1 - local 2 - remote
C3	ALERT-STATUS	not used
C4	THREAT-PRIORITY	minimum time to reach a protected asset or AD unit (threat)
C5	TRACK-STATUS	0 - video, 1 - unidentified track, 2 - identified, 3 - assigned, 4 - engaged
C6	TRACK-TYPE	not used
C7	PERCEIVED-PRIMARY-ID	0 - not identified, 1 - hostile, 2 - friendly, 3 - unknown
C8	TIME-LAST-ATTENDED-TO	Time track was last attended to
C9	OWNER-ID	ID of operator who owns the track (zero if none)
C10	SYMBOL	

MOPADS/DATA BASE:	DATA LIST DESCRIPTION	5-17/DLD - 22
Data List Label (25 char.max): Owner_Directory_Label-IRD_		Page 2 of 2
Owner Directory Code: 12	DL Id: NECC(ACC)NECCC'IRD'1010	Rev.Date: 26 MAY 1983
DL Type: IDL/R C DIM =	X RDL/R C DIM = 13 CDL/max char=	Author: POLITO

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
C11	SYMBOL-DATA-1	(See below)
C12	SYMBOL-DATA-2	
C13	ASSIGNED-FU	Columns in NFSTOR of the fire unit the track is assigned to a zero if not assigned.
		Group 0-73 C10 - not used C11 - 1 if a hold fire or cease engage message has been sent for this track. Zero if not. C12 - not used
		Battalion 0-73 C10 - not used C11 - 1 if a hold fire or cease engage message has been sent for this track. Zero if not. C12 - time last attempted to assign the track
		IHAWK C10 - 0 - not categorized, 1 - IPAR, 2 - CWAR, 3 - both C11 - time track sent to TCA by ASO C12 - 0 - not accepted yet by TCO, 1 - CWAR target accepted by TCA

15-0 THE "CRITICAL-ASSET-CONFIGURATION" DIRECTORY (CODE 13)

MOPADS/DATA BASE **DIRECTORY DESCRIPTION**

DIRECTORY LABEL(CODE/LABEL): 13|CRITICAL-ASSET-CONFIGURATION

DIRECTORY LEVEL: 2

OWNER DIRECTORY(CODE): 4 SCENARIOS

ID IN OWNER DIRECTORY: 13|(user assigned)

NUMBER OF WORDS IN ID'S OF OWNED DIRECTORIES & DATA LISTS: 2

RANKING CODE FOR OWNED DIRECTORIES: 2

RANKING CODE FOR OWNED DATA LISTS: 0

DESCRIPTION:

MOPADS/DIRECTORY CONTENTS

DIRECTORY CODE: 13

DIRECTORY LABEL: CRITICAL-ASSET-CONFIGURATION

TYPE	CODE	LABEL	NUMBER
DR	8	air scenario	as needed
DL	-	COORDINATE-AND-ASSET-DATA	1

MOPADS/DATA BASE: DATA LIST DESCRIPTION 5-17/DLD - 14

Data List Label (25 char.max):COORDINATE-AND-ASSET-DATA Page 1 of 1
Owner Directory Code:13|CRITICAL-ASSET-CONFIGURATION DL Id:NECC('CD')11 Rev.Date:6 MAY 83
DL Type:____IDL/R C DIM =_____ X RDL/R C DIM =1 ____CDL/max char=____ Author:_POLITO____

ELEMENT NO.	ELEMENT LABEL(25 char)	ELEMENT DESCRIPTION & INITIAL (I) OR DEFAULT (D) VALUE
1	REFERENCE-LATITUDE	decinal degrees (negative for south latitude)
2	REFERENCE-LONGITUDE	decinal degrees (negative for east longitude)
3	REFERENCE-ALTITUDE	ft.above MSL
4	NUMBER-OF-CRITICAL-ASSETS	he number of critical asset data sets which follow
5	X-COORDINATE	(n.miles) for first critical asset
6	Y-COORDINATE	(n.miles) for first critical asset
7	Z-COORDINATE	(ft.) for first critical asset
8	SITE-TYPE	A code value (currently always zero)
9	asset label(5 char.max)	Status (0-functional, 1-destroyed)
10	(NOT USED)	zero
		Repeat (5) - (10) for each additional asset

IV. REFERENCES

Polito, J. The MOPADS data base control system (MOPADS/DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Polito, J. Performing MOPADS simulations (MOPADS Vol. 3.3). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Polito, J. MOPADS user interface (MOPADS/UI) (MOPADS Vol. 5.12), Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Polito, J. Creating reference air defense system modules (MOPADS Vol. 4.6). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d)

Polito, J. Creating MOPADS air scenarios (MOPADS Vol. 4.7). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (e)

V. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U.S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, TX 79916

Pritsker & Associates, Inc.
P. O. Box 2413
West Lafayette, IN 47906

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, IN 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Room 108
Boulder, CO 80301

VI. CHANGE NOTICES

APPENDIX Y

MOPADS FINAL REPORT:

THE MOPADS DATA BASE

TABLE OF CONTENTS

LIST OF FIGURES

APPENDIX Z

MOPADS FINAL REPORT:

THE MOPADS DATA BASE APPLICATION PROGRAMS (MOPADS/DBAP)

TABLE OF CONTENTS

Standard MOPADS Terminology

AIR DEFENSE SYSTEM	A component of Air Defense which includes equipment and operators and for which technical and tactical training are required. Examples are IHAWK and the AN/TSQ-73.
AIR DEFENSE SYSTEM MODULE	Models of operator actions and equipment characteristics for Air Defense Systems in the MOPADS software. These models are prepared with the SAINT simulation language. Air Defense System Modules include the SAINT model and all data needed to completely define task element times, task sequencing requirements, and human factors influences.
AIR SCENARIO	A spatial and temporal record of aerial activities and characteristics of an air defense battle. The Air Scenario includes aircraft tracks, safe corridors, ECM, and other aircraft and airspace data. See also Tactical Scenario.
BRANCHING	A term used in the SAINT simulation language to mean the process by which TASK nodes are sequenced. At the completion of the simulated activity at a TASK node, the Branching from that node determines which TASK nodes will be simulated next.
DATA BASE CONTROL SYSTEM	That part of the MOPADS software which performs all direct communication with the MOPADS Data Base. All information transfer to and from the data base is performed by invoking the subprograms which make up the Data Base Control System.
DATA SOURCE SPECIALIST	A specialist in obtaining and interpreting Army documentation and other data needed to prepare Air Defense System Modules.

ENVIRONMENTAL STATE VARIABLE	An element of an Environmental State Vector.
ENVIRONMENTAL STATE VECTOR	An array of values representing conditions or characteristics that may affect more than one operator. Elements of Environmental State Vectors may change dynamically during a MOPADS simulation to represent changes in the environment conditions.
MODERATOR FUNCTION	A mathematical/logical relationship which alters the nominal time to perform an operator activity (usually a Task Element). The nominal time is changed to represent the operator's capability to perform the activity based on the Operator's State Vector.
MOPADS DATA BASE	A computerized data base designed specifically to support the MOPADS software. The MOPADS Data Base contains Simulation Data Set(s). It communicates interactively with MOPADS Users during pre- and post-run data specification and dynamically with the SAINT software during simulation.
MOPADS MODELER	An analyst who will develop Air Defense System Modules and modify/develop the MOPADS software system.
MOPADS USER	An analyst who will design and conduct simulation experiments with the MOPADS software.
MSAINT (MOPADS/SAINT)	The variant of SAINT used in the MOPADS system. The standard version of SAINT has been modified for MOPADS to permit shareable subnetworks and more sophisticated interrupts. The terms SAINT and MSAINT are used interchangeably when no confusion will result. See also, SAINT.

OPERATOR STATE VARIABLE	One element of an Operator State Vector.
OPERATOR STATE VECTOR	An array of values representing the condition and characteristics of an operator of an Air Defense System. Many values of the Operator State Vector will change dynamically during the course of a MOPADS simulation to represent changes in operator condition.
OPERATOR TASK	An operator activity identified during weapons system front-end analyses.
SAINT	The underlying computer simulation language used to model Air Defense Systems in Air Defense System Modules. SAINT is an acronym for Systems Analysis of Integrated Networks of Tasks. It is a well documented language designed specifically to represent human factors aspects of man/machine systems. See also MSAINT.
SIMULATION DATA SET	The Tactical Scenario plus all required simulation initialization and other experimental data needed to perform a MOPADS simulation.
SIMULATION STATE	At any instant in time of a MOPADS simulation the Simulation State is the set of values of all variables in the MOPADS software and the MOPADS Data Base.
SYSTEM MODULES	See Air Defense System Modules.
TACTICAL SCENARIO	The Air Scenario plus specification of critical assets and the air defense configuration (number, type and location of weapons and the command and control system).

TACTICAL SCENARIO COMPONENT	An element of a Tactical Scenario, e.g., if a Tactical Scenario contains several Q-73's, each one is a Tactical Scenario Component.
TASK	See Operator Task.
TASK ELEMENTS	Individual operator actions which, when grouped appropriately, make up operator tasks. Task elements are usually represented by single SAINT TASK nodes in Air Defense System Modules.
TASK NODE	A modeling symbol used in the SAINT simulation language. A TASK node represents an activity; depending upon the modeling circumstances, a TASK node may represent an individual activity such as a Task Element, or it may represent an aggregated activity such as an entire Operator Task.
TASK SEQUENCING MODERATOR FUNCTION	A mathematical/logical relationship which selects the next Operator Task which an operator will perform. The selection is based upon operator goal seeking characteristics.

MOPADS Abbreviations

DBCS	Data Base Control System.
DBAP	Data Base Application Programs
DBF	Data Base File.

I. PURPOSE

This document describes a set of utility programs used by the MOPADS software to implement the specific data base structure of MOPADS. The MOPADS Data Base Control System (DBCS), Polito (1983a), is a set of programs that manage the data base files (DBF). The DBCS, however, is generic in the sense that nothing in these programs is unique to MOPADS. The particular data and its organization in a MOPADS data base is documented in Polito (1983b). The Data Base Applications Programs (DBAP), which are dicussed here, use the DBCS programs to implement and access the particular DBF discussed in Polito (1983b).

Thus, the DBAP implements high level data access functions for the MOPADS modeler to use rather than requiring lowel level data base accesses. For example, the DBAP has a program to retrieve information from a operator's operator state vector directly. The modeler could accomplish the same thing bv making direct calls to the DBCS, but this latter would require that the modeler know details of how data is stored in the DBF. The DBAP masks the modeler from having to know this low level information which is not relevant to his data access requirements. Furthermore, many such high level functions involve intricate combinations of low level functions that would be a burden to reprogram each time they were required.

This is a reference document intended for MOPADS modelers who must extend or revise the MOPADS software. The MOPADS program suffix for the DBAP is "A." All of the subprogram names and variables in labelled COMMON in the DBAP end with the letter "A." Note that not all programs in MOPADS that end with "A" are part of the DBAP. SAINT and programs in FFIN2, Polito (1983c), were written before MOPADS and do contain some programs that end with the letter "A."

II. DATA STRUCTURE DESCRIPTION

The DBAP is a collection of more or less unrelated subprograms that perform a variety of functions for the MOPADS modeler. Therefore, there is no unified data structure that is internal to the DBAP. All information is exchanged through formal subprogram parameters.

The structure of the data files created by the DBCS is discussed in Polito (1983a) and the contents of the DBF used by MOPADS is presented in detail in Polito (1983b). The contents of the single labeled COMMON area contained in the DBAP is discussed in SECTION VIII.

III. OVERVIEW OF THE FLOW OF CONTROL

The DBAP programs are generally intermediaries between the DBCS and the programs which implement the MOPADS User Interface and the Air Defense System Modules. DBAP programs are used extensively by both the User Interface, Polito (1983d), and the SAINT simulation phases of MOPADS. See Polito (1983e) for a description of this software organization.

DBAP programs normally are called by these external programs and in turn call other programs in the DBCS, MOPADS/UTIL, Polito & Goodin (1983), and, in some cases, FFIN2, Polito (1983c). DBAP programs seldom call other programs in the DBAP, and programs called from the DBAP do not in turn call other programs in the DBAP.

IV. EXTERNAL FILE USAGE

The DBAP operates directly on only three files, and none of these are opened or closed by DBAP programs. These files are the following:

1. MOPADS Line Printer Output
2. User Interface Interactive Input
3. User Interface Interactive Output

The line printer output file is used only for error processing which will be discussed in Section VII. The interactive input and output files are used only by subroutine CREATA to request information while creating a new MOPADS DBF.

The interactive input and output unit numbers are passed to the DBAP as formal parameters. The line printer file unit number is maintained in COMMON, but it is initialized through a subprogram call. All of these files are sequential, formatted files.

To summarize, the DBAP has no responsibility to open, close, or otherwise manage any files. Unit numbers of all files which it reads or writes are passed to it through subprogram parameters.

V. SUBPROGRAM DESCRIPTIONS

The following pages contain description of all subprograms that are part of the DBAP. Each program description contains an explanation of its purpose, parameters, and alternate returns.

```
      SUBROUTINE APOPTA(IOPT,IVL,NIV,RVL,NRV)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       APOPTA SETS OPTIONS FOR THE DBAP
C
C--INPUT PARAMETERS:
C       IOPT-OPTION NUMBER(SEE BELOW)
C       IVL(NIV)-INTEGER VALUES FOR OPTIONS
C       RVL(NRV)-REAL VALUES FOR OPTIONS
C
C      OPTIONS
C     ------------------------------------------------------
C        1          OUTPUT FILE NUMBER ( IVL(1)  )
C
```

```
      SUBROUTINE ASROWA(NROW,NR,IDBA,ROW,KERR,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       TO RETURN A ROW OF AN AIR SCENARIO TRACK DATA LIST.
C--INPUT PARAMETERS:
C       NROW-THE DESIRED ROW NUMBER
C       NR-THE NUMBER OF ELEMENTS IN A ROW
C--INPUT/OUTPUT PARAMETERS:
C       IDBA(2)-THE DBAA OF THE TRACK DATA DATA LIST IN THE AIR SCENARIO
C--OUTPUT PARAMETERS:
C       ROW(NR)-AN ARRAY TO HOLD THE VALUES IN THE SPECIFIED ROW.
C               SEE THE DD DESCRIPTION FOR "AIRCRAFT-TRACKS-HOSTILE
C               (FRIENDLY)(OTHER)" FOR THE CONTENTS OF ROW.
C       KERR-ERROR CODE
C            0-NO ERROR
C            1-ROW NROW DOES NOT EXIST
C            2-NROW.LE.0
C--ALTERNATE RETURNS:
C       1-KERR.NE.0
```

```
      SUBROUTINE CCDLA(IOPT,NROW,IDRF1I,ROW)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       CCDLA MANIPULATES THE 'COPY-COUNTER' DL OF A SIMULATION DATA SET
C       CCDLA ASSUMES THE DBAA OF THE COPY COUNTER DL IS IN COLUMN 3
C       OF IDRF1I. THE COPY COUNTER HAS 2 COLUMNS. FOR ROW N,
C       COL. 1 HAS VALUE N*1000+I WHICH MEANS I COPIES OF THE ADSM
C       WITH BASIC ACN N*1000 HAVE BEEN CREATED(BUT NEED NOT STILL
C       EXIST.)  COLUMN 2 IS THE CURRENT NUMBER OF COPIES THAT DO
C       EXIST.
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-RETURN CONTENTS OF ROW NROW ONLY.
C            1-ADD ONE TO BOTH COLUMNS OF ROW NROW AND RETURN RESULTS IN
C              'ROW'.
C            2-SUBTRACT 1 FROM COLUMN 2 OF ROW NROW. DO NOTHING IF
C              COLUMN 2 IS ZERO.  RETURN RESULTS IN 'ROW'
C       NROW-THE ROW NUMBER TO PROCESS
C--INPUT/OUTPUT PARAMETERS:
C       IDRF1I(4,3)-DRAA'S
C                   COL. 1- DRAA OF THE SIMULATION DATA SET ON A
C                           DIRECT LINE UP THE DR TREE FROM THE
C                           CURRENT DR.
C                   COL. 2- DRAA OF THE COMMAND AND CONTROL DR
C                           ON A DIRECT LINE UP FROM THE CURRENT DR.
C                   COL. 3- DBAA OF THE COPY COUNTER DL OF THE
C                           SIMULATION DATA SET IN COL. 1.
C--OUTPUT PARAMETERS:
C       ROW(2)-CONTENTS OF ROW NROW AFTER IOPT IS PROCESSED.
```

```
      SUBROUTINE CENCA (NECC,LABEL,*)
C
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C**PURPOSE:  THIS SUBROUTINE IS USED TO DECODE INTEGER INDEX VALUES
C            INTO THEIR CHARACTER STRING EQUIVALENTS (LABELS).
C
C**INPUT PARAMETERS:   NECC=INTEGER VALUE TO BE DECODED
C
C**OUTPUT PARAMETERS:  LABEL=DECODED CHARACTER STRING
C
```

```
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN OCCURS IF NECC
C                      CONTAINS AN INTEGER COMPONENT THAT CAN NOT BE
C                      DECODED. FOR EXAMPLE,INDEX VALUES OF LESS THAN
C                      OR EQUAL TO 00,OR GREATER THAN 36 CAN NOT BE
C                      DECODED. ALSO, IF LABEL IS NOT
C                      LONG ENOUGH TO HOLD THE CHARACTERS.
C
```

```
      SUBROUTINE CHKGLA(IGL,IOPR,KGL)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      CHKGLA WILL DETERMINE IF A SPECIFIED OPERATOR HAS
C      A PARTICULAR GOAL.
C--INPUT PARAMETERS:
C      IGL-A GOAL NUMBER. CHKGLA WILL CHECK TO SEE IF THE OERATOR
C          HAS THIS GOAL
C--INPUT/OUTPUT PARAMETERS:
C      IOPR(2)-DBAA OF THE OPERATOR
C--OUTPUT PARAMETERS:
C      KGL-GOAL STATUS
C          1- THE OPERATOR HAS THE GOAL
C          2- THE OPERATOR DOES NOT HAVE THE GOAL
```

```
      SUBROUTINE CLRSTA(IDOP)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      CLRSTA WILL CLEAR (EMPTY)  AN OPERATOR'S TASK STACK.
C--INPUT PARAMETERS:
C      IDOP-THE OPERATOR'S ID
```

```
      SUBROUTINE CPDLA(NDBF,NDBT,ID,NID,IDBAF,IDRT,IDBAT,IERR,*)
C--MODULE: MOPADS/UI
C--REFERENCE: MOPADS VOL. 5.12
C--PURPOSE:
C      CPDLA WILL COPY A DL FROM ONE DR TO ANOTHER WITH REPLACEMENT.
C
C--INPUT PARAMETERS:
C      NDBF-DB(1-WORKING,2-REFERENCE) OF THE DL TO BE COPIED
C      NDBT-DB(AS ABOVE) OF THE DR TO BE COPIED TO.
C      ID(NID)-ID OF THE DL IN THE NEW DR
```

```
C--INPUT/OUTPUT PARAMETERS:
C       IDBAF(2)-DBAA OF THE DL THAT IS TO BE COPIED
C       IDRT(4)-DRAA OF THE DR TO RECEIVE A COPY OF THE DL
C--OUTPUT PARAMETERS:
C       IDBAT(2)-DBAA OF THE COPY OF THE DL ON THE "TO" DR.
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0

------------------------------------------------------------------------

      SUBROUTINE CPDRA(NDBF,NDBT,ID,NID,LABEL,IDRF,IDROT,IDRT,IERR,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       CPDRA WILL COPY A DR FROM ONE DB TO ANOTHER WITH REPLACEMENT
C       (OR WITHIN ONE DB). THE DR MAY OWN DL'S BUT NO DR'S.  MORE
C       CORRECTLY, ONLY THE OWNED DL'S WILL BE COPIED.
C--INPUT PARAMETERS:
C       NDBF-"FROM" DB (1-WORKING, 2-REFERENCE)
C       NDBT-"TO" DB
C       ID(NID)-THE ID OF THE DR ON THE 'TO' DIRECTORY
C       LABEL-(CHARACTER) LABEL OF THE DR ON THE 'TO' DR
C--INPUT/OUTPUT PARAMETERS:
C       IDRF(4)-DRAA OF THE DR TO BE COPIED
C       IDROT(4)-DRAA OF THE OWNER DIRECTORY ON NDBT
C--OUTPUT PARAMETERS:
C       IDRT(4)-DRAA OF THE COPIED DIRECTORY ON NDBT
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0. IF A NON-DBCS ERROR OCCURS, ALTERNATE RETURN
C                    1 WILL BE TAKEN WITH IERR=0.

------------------------------------------------------------------------

      SUBROUTINE CREATA(NDB,JIN,JOUT,IERR,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       CREATA WILL CREATE THE SHELL OF A MOPADS DATA BASE ON A NEW
C       DB FILE.  IT ASSUMES THAT THE DB FILE IS VIRGIN AND WRITES
C       ON IT.  THEREFORE, ANY INFORMATION WHICH MAY HAVE BEEN ON
C       THE FILE WILL BE LOST. CREATA WILL SET WRITE PERMISSION
C       ON THE FILE (MODED=2)
```

```
C--INPUT PARAMETERS:
C       NDB-DATA BASE(1-WORKING,2-REFERENCE)
C       JIN-INPUT FILE
C       JOUT-OUTPUT FILE
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE. CREATA WILL CLOSE NDB BEFORE
C                 RETURNING.
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE ERRORA(MSG,NMSG,SUBP,IPAR,INPAR,PAR,NPAR,IDA)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       ERRORA WILL PROCESS RUN TIME ERRORS WITH DATA BASE ACCESSES
C       THAT INDICATE FAULTY OR MISUSED MOPADS DATA BASES. AS
C       COMPLETE A DUMP AS POSSIBLE WILL BE DONE TO DIAGNOSE THE
C       PROBLEM. ERRORA ASSUMES THE WORKING DB, AND IT WILL TERMINATE
C       EXECUTION.
C--INPUT PARAMETERS:
C        MSG(NMSG)-CHARACTER ARRAY WITH ERROR MESSAGE. EACH ELEMENT
C                  WILL BE TREATED AS A ROW OF AN ERROR MESSAGE.
C       SUBP-CHARACTER VARIABLE WITH THE NAME OF THE CALLING PROGRAM
C       IPAR(INPAR)-INTEGER ARRAY TO PRINT IF INPAR .GT. 0
C       PAR(NPAR)-  REAL ARRAY TO PRINT IF NPAR .GT.0
C--INPUT/OUTPUT PARAMETERS:
C       IDA(4)-THE DRAA OR DBAA OF THE OFFENDING DR OR DL. IF IDA IS
C              A DBAA FOR A DL, IT NEED BE DIMENSIONED ONLY TO 2.
```

```
      SUBROUTINE EVLPA(KGOAL,GS,IDBOP,KASE,GPRI,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       EVLPA WILL EVALUATE THE GOAL PRIORITY FUNCTION FOR A PARTICULAR
C       GOAL AND OPERATOR.
C--INPUT PARAMETERS:
C       KGOAL-THE GOAL NUMBER TO EVALUATE
C       GS-THE GOAL STATE FOR GOAL KGOAL
C--INPUT/OUTPUT PARAMETERS:
C       IDBOP(2)-DBAA OF THE OPERATOR
```

```
C--OUTPUT PARAMETERS:
C      KASE-OUTPUT CONDITION
C           1-PRIORITY CALCULATED AND STORED IN GPRI
C           2-THE OPERATOR DOES NOT HAVE GOAL KGOAL. GPRI=0.
C           3-KGOAL IS INVALID(I.E. .LE.0 OR .GT. THE MAX NUMBER OF
C             GOALS)
C      GPRI-GOAL PRIORITY FOR GOAL KGOAL
C--ALTERNATE RETURNS:
C      1-KASE=3

------------------------------------------------------------------------

      SUBROUTINE FNDAKA(IDOP,IELL,IREL,IVAL,NCNDS,MCOL,MSID,MSAA,NCOL)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      FNDAKA WILL FIND ACKNOWLEDGMENT MESSAGES BELONGING TO A
C      PARTICULAR OPERATOR THAT SATISFY SPECIFIED CONDITIONS.
C--INPUT PARAMETERS:
C      IDOP-THE OPERATOR'S ID
C      IELL(NCNDS),
C        IREL(NCNDS),
C          IVAL(NCNDS)-CONDITIONS TO CHECK FOR THE MESSAGE ID.
C                      ELEMENT IELL(-) OF THE MESSAGE MUST RELATE
C                      TO IVAL(-) WITH THE RELATION IREL(-).
C                      IELL(-) MAY HAVE THE VALUES:
C                          3-A CONDITION ON THE FUNCTION TYPE
C                          4-A CONDITION ON THE SUBTYPE
C                          5-A CONDITION ON THE MESSAGE PRIORITY
C
C                      VALUES FOR IREL(-) ARE:
C                          1-LT
C                          2-LE
C                          3-EQ
C                          4-GE
C                          5-GT
C                          6-NE
C
C                      A)  ALL NCNDS CONDITIONS MUST BE SATISFIED FOR
C                          AN ACKNOWLEDGMENT MESSAGE TO BE FOUND BY
C                          FNDAKA.
C                      B)  IF NCNDS=0, ALL ACKNOWLEDGMENT MESSAGES
C                          BELONGING TO OPERATOR IDOP WILL BE FOUND
C                      C)  NCNDS MAY NOT BE GREATER THAN 6
C                      D)  EXAMPLE:  NCNDS=2
C                           IELL(1)=3, IREL(1)=3, IVAL(1)=1
C                           IELL(2)=4, IREL(2)=4, IVAL(2)=8
```

```
C
C                          THE ABOVE EXAMPLE WILL FIND ONLY
C                          COMMAND MESSAGES(FUNCTION TYPE=1) WITH
C                          SUBTYPE .GE. 1
C
C      MCOL-LENGTH OF MSID AND MSAA ARRAYS
C--OUTPUT PARAMETERS:
C      MSID(5,MCOL)-MESSAGE ID'S OF ALL MESSAGES FOR OPERATOR IDOP
C                   THAT SATISFY THE CONDITIONS.
C      MSAA(2,MCOL)-DATA BASE ADDRESSES OF ALL MESSAGES IN MSID
C      NCOL-NUMBER OF MESSAGES FOUND.
C           0.LE.NCOL.LE.MCOL.  FNDAKA WILL GENERATE AN ERROR IF
C           THE NUMBER OF MESSAGES IS GREATER THAN MCOL.

------------------------------------------------------------------------

      SUBROUTINE FNDMSA(IDOP,IELL,IREL,IVAL,NCNDS,MCOL,MSID,MSAA,NCOL)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      FNDMSA WILL FIND MESSAGES BELONGING TO A PARTICULAR OPERATOR
C      THAT SATISFY SPECIFIED CONDITIONS.
C--INPUT PARAMETERS:
C      IDOP-THE OPERATOR'S ID
C      IELL(NCNDS),
C        IREL(NCNDS),
C          IVAL(NCNDS)-CONDITIONS TO CHECK FOR THE MESSAGE ID.
C                      ELEMENT IELL(-) OF THE MESSAGE MUST RELATE
C                      TO IVAL(-) WITH THE RELATION IREL(-).
C                      IELL(-) MAY HAVE THE VALUES:
C                          3-A CONDITION ON THE FUNCTION TYPE
C                          4-A CONDITION ON THE SUBTYPE
C                          5-A CONDITION ON THE MESSAGE PRIORITY
C
C                      VALUES FOR IREL(-) ARE:
C                          1-LT
C                          2-LE
C                          3-EQ
C                          4-GE
C                          5-GT
C                          6-NE
C
```

```
C                    A)  ALL NCNDS CONDITIONS MUST BE SATISFIED FOR
C                        A MESSAGE TO BE FOUND BY FNDMSA
C                    B)  IF NCNDS=0, ALL MESSAGES BELONGING TO
C                        OPERATOR IDOP WILL BE FOUND
C                    C)  NCNDS MAY NOT BE GREATER THAN 6
C                    D)  EXAMPLE:  NCNDS=2
C                         IELL(1)=3, IREL(1)=3, IVAL(1)=1
C                         IELL(2)=4, IREL(2)=4, IVAL(2)=8
C
C                         THE ABOVE EXAMPLE WILL FIND ONLY
C                         COMMAND MESSAGES(FUNCTION TYPE=1) WITH
C                         SUBTYPE .GE. 8
C
C      MCOL-LENGTH OF MSID AND MSAA ARRAYS
C--OUTPUT PARAMETERS:
C      MSID(5,MCOL)-MESSAGE ID'S OF ALL MESSAGES FOR OPERATOR IDOP
C                   THAT SATISFY THE CONDITIONS.
C      MSAA(2,MCOL)-DATA BASE ADDRESSES OF ALL MESSAGES IN MSID
C      NCOL-NUMBER OF MESSAGES FOUND.
C           0.LE.NCOL.LE.MCOL.  FNDMSA WILL GENERATE AN ERROR IF
C           THE NUMBER OF MESSAGES IS GREATER THAN MCOL.

------------------------------------------------------------------------

      SUBROUTINE FTKAA(NCUR,NROW)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      FTKAA WILL FIND AN UNUSED ROW IN THE "TRACK DATA" DL OF A
C      PARTICULAR ADSM
C--INPUT PARAMETERS:
C      NCUR-ADSM COPY ROW NUMBER
C--OUTPUT PARAMETERS:
C      NROW-AN AVAILABLE ROW

------------------------------------------------------------------------

      SUBROUTINE GETEVA(IDE,N,NEE,KE,XE)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C
C     GETEVA RETURNS THE VALUES OF SPECIFIED HUMAN FACTORS ELEMENTS OF
C     ENVIRONMENTAL STATE VECTORS. IT IS CALLED FROM THE HUMAN FACTORS
C     MODULE.
C
```

```
C--INPUT PARAMETERS:
C       IDE(N)-THE DBAA OF THE ESV
C       NEE(KE)-THE DESIRED ELEMENT NUMBERS. E.G. NEE(2)=7 IMPLIES THAT
C               HUMAN FACTORS ELEMENT 7 OF THE ESV IS TO BE RETURNED.
C--OUTPUT PARAMETERS:
C       XE(KE)-THE CURRENT VALUES OF THE ELEMENTS SPECIFIED IN NEE.
C              E.G. IF NEE(2)=7, THEN XE(2)=THE VALUE OF HF ELEMENT 7.

-----------------------------------------------------------------------

      SUBROUTINE GETOVA(IDO,N,NEO,KO,XO)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GETOVA RETURNS THE VALUES OF SPECIFIED HUMAN FACTORS ELEMENTS OF
C       STATE VECTORS(OSV). IT IS CALLED FROM THE HUMAN FACTORS
C       MODULE.
C--INPUT PARAMETERS:
C       IDO(N)-DBAA OF THE OSV
C       NEO(KO)-THE DESIRED ELEMENT NUMBERS. E.G. NEO(2)=7 IMPLIES THAT
C               HUMAN FACTORS ELEMENT 7 OF THE OSV IS TO BE RETURNED.
C--OUTPUT PARAMETERS:
C       XO(KO)-THE CURRENT VALUES OF THE HUMAN FACTORS ELEMENTS
C              SPECIFIED IN NEO. E.G. IF NEO(2)=7, THEN XO(2)=THE
C              VALUE OF HF ELEMENT 7.

-----------------------------------------------------------------------

      SUBROUTINE GETTSA(IOPT,ITASK,NTS,NTSK,NT,XTSK)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GETTSA RETURNS THE VALUES OF SPECIFIED TASK DATA ELEMENTS.
C       IT IS CALLED  TO OBTAIN
C       INFORMATION ABOUT THE TASK BEING PERFORMED.
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-REQUESTING TASK TIME INFORMATION
C            1-REQUESTING HUMAN FACTORS DATA SPECIFIC TO THE TASK.
C            2-REQUESTING INFORMATION ON THE SKILLS REQUIRED TO
C              PERFORM THE TASK.
C            3-REQUESTING INFORMATION ON SYSTEM RESOURCES NEEDED.
C       ITASK(NTS)-THE DBAA OF THE TASK DATA LIST+TASK NUMBER.
C                  ITASK(1) & (2) ARE THE DBAA. ITASK(3)=NODE NUMBER.
```

```
C      NTSK(NT)-THE DESIRED ELEMENT NUMBERS. THERE ARE 4 CASES:
C               IOPT=0--NTSK NOT USED
C               IOPT=1--NTSK CONTAINS ELEMENT NUMBERS OF THE TASK
C                       SPECIFIC CATEGORIES.
C               IOPT=2--NTSK CONTAINS SKILL NUMBERS
C               IOPT=3--NTSK CONTAINS RESOURCE NUMBERS
C--OUTPUT PARAMETERS:
C      XTSK(NT)-THE VALUES OF THE VARIABLES REQUESTED IN NTSK ABOVE.
C               4 CASES:
C               IOPT=0--XTSK(1) TO (3) ARE DISTRIBUTION TYPE, MEAN, AND
C                       STANDARD DEVIATION.
C               IOPT=1--IF(NTSK(2)=3, XTSK(2)=THE VALUE OF TASK
C                       SPECIFIC CATEGORY 3.
C               IOPT=2--IF(NTSK(2)=3, XTSK(2)=THE WEIGHT OF SKILL
C                       CATEGORY 3 REQUIRED TO PERFORM THE TASK.
C                       IF SKILL 3 IS NOT REQUIRED, XTSK(2)=0 IS
C                       RETURNED.
C               IOPT=3--IF NTSK(2)=3, THEN XTSK(2)=RESOURCE PARAMETER
C                       FOR RESOURCE 3. ZERO IF RESOURCE 2 NOT NEEDED.

------------------------------------------------------------------------

      SUBROUTINE GTKAA(NCUR,ID,ICOL,IREL,VAL,NCND,LEN,IDATA,NCL,MORE)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      GTKAA IS USED TO LOCATE TRACK INFORMATION IN AN ADSM'S
C      "TRACK-DATA" DATA LIST THAT SATISFIES CERTAIN CONDITIONS.
C--INPUT PARAMETERS:
C      NCUR-ADSM COPY ROW NUMBER
C      ID-ID OF THE OPERATOR. IF ID.NE.0, GTKAA ONLY FINDS TRACKS
C         THAT BELONG TO OPERATOR ID. IF ID=0, ALL TRACKS SATISFYING
C         THE CONDITIONS ARE FOUND.
C      ICOL(NCND),
C        IREL(NCND),
C          VAL(NCND)-THE CONDITIONS TO SEARCH FOR.  TRACK DATA IS
C                    STORED IN ROWS OF A MATRIX IN THE DB.  ICOL(-)
C                    IS A COLUMN NUMBER, IREL(-) IS A RELATION CODE,
C                    AND VAL(-) IS A VALUE FOR COMPARISON. GTKAA WILL
C                    SEARCH FOR ROWS OF THE "TRACK DATA" DL WHOSE
C                    ICOL(-) COLUMN RELATES TO VAL(-)  IN THE MANNER
C                    SPECIFIED BY IREL(-).  IREL(-) MAY HAVE THE
C                    FOLLOWING VALUES:
```

```
C
C                                   1-LT
C                                   2-LE
C                                   3-EQ (TOLERANCE OF 0.001 IS USED)
C                                   4-GE
C                                   5-GT
C                                   6-NE (TOLERANCE OF 0.001 IS USED)
C
C                       ALL NCND CONDITIONS MUST BE TRUE FOR A ROW TO BE
C                       FOUND.
C                       EXAMPLE:  ICOL(2)=4,IREL(2)=5,VAL(2)=2. WILL
C                                 SEARCH FOR ROWS WHOSE 4-TH COLUMN
C                                 IS GREATER THAN 2.
C
C                       IF NCND=0, GTKAA SEARCHES ONLY FOR TRACKS THAT
C                       BELONG TO OPERATOR ID. IF NCND=0 AND ID=0,
C                       GTKAA RETURNS ALL TRACKS.
C
C                       GTKAA CAN BE USED TO SEARCH FOR EMPTY ROWS OF
C                       THE DL BY SENDING ID=0,ICOL(1)=1,IREL(1)=3,
C                       VAL(1)=0., AND NCND=1.
C
C       LEN-THE NUMBER OF COLUMNS IN THE IDATA ARRAY.
C--OUTPUT PARAMETERS:
C       IDATA(3,LEN)-EACH COLUMN CAN REPRESENT A TRACK THAT SATISFIES
C                    THE ABOVE CONDITIONS. FOR A PARTICULAR COLUMN,
C                         IDATA(1,-)-ROW NUMBER IN THE TRACK DATA
C                                    DATA LIST OF THE ADSM
C                         IDATA(2,-)-COLUMN IN NTRACK FOR THE TRACK
C                         IDATA(3,-)-SYMBOL CODE FOR THE TRACK
C       NCL-COLUMNS 1 THRU NCL OF IDATA WILL CONTAIN TRACK DATA
C           THAT SATISFIES THE CONDITIONS. ZERO IF NONE.
C       MORE-MORE=0 IF IDATA CONTAINS ALL OF THE TRACKS SATISFYING
C            THE CONDITIONS.
C            MORE.NE.0 IMPLIES IDATA IS FILLED AND MORE TRACKS REMAIN
C            WHICH SATISFY THE CONDITIONS. TO GET THESE TRACKS, CALL
C            GTKAA AGAIN WITHOUT CHANGING THE VALUE OF MORE.
C            INITIALLY, MORE SHOULD BE SET TO ZERO BY THE USER.

-----------------------------------------------------------------------

      SUBROUTINE GTKBA(NCUR,NROW,IOK,ROW)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GTKBA WILL RETURN A ROW OF THE "TRACK-DATA" DL FOR A SPECIFIED
C       ADSM
```

```
C--INPUT PARAMETERS:
C       NCUR-ADSM COPY ROW NUMBER
C       NROW-ROW OF THE DATA LIST TO GET
C--OUTPUT PARAMETERS:
C       IOK-OUTPUT CONDITION
C           1-ROW NROW CONTAINED TRACK DATA
C           2-ROW NROW DID NOT CONTAIN TRACK DATA. THIS IS AN ERROR.
C           3-ROW NROW DOES NOT EXIST-BEYOND END OF DATA LIST
C       ROW(13)-CONTENTS OF ROW NROW IF IOK=1

------------------------------------------------------------------------

      SUBROUTINE GTKCA(NCUR,ICODE,NPOINT,NCOL,VAL,IOK)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GTKCA IS USED TO GET A SINGLE ELEMENT OF A ROW IN THE
C       "TRACK DATA" DATA LIST OF AN ADSM.
C--INPUT PARAMETERS:
C       NCUR-ADSM COPY ROW NUMBER
C       ICODE-MEANING OF NPOINT
C             1-NPOINT IS A ROW NUMBER IN THE "TRACK DATA" DL(THIS
C               IS THE EFFICIENT WAY)
C             2-NPOINT IS A COLUMN NUMBER IN THE NTRACK ARRAY(THIS
C               IS THE INEFFICIENT WAY)
C      NPOINT-ROW NUMBER OR NTRACK COLUMN NUMBER (SEE ICODE)
C       NCOL-COLUMN NUMBER OF THE ROW TO GET
C--OUTPUT PARAMETERS:
C       VAL-VALUE OF THE NCOL-TH ELEMENT OF THE SPECIFIED
C           ROW.
C       IOK-OUTPUT CONDITION
C           1-ALL OK
C           2-NCOL NOT VALID OR THE SPECIFIED ROW DOES NOT CONTAIN
C             TRACK DATA. VAL NOT CHANGED.

------------------------------------------------------------------------

      SUBROUTINE GTSPA(NCUR,IPAR,SYSP)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       GTSPA WILL GET A SINGLE SYSTEM PARAMETER FROM THE ENVIRONMENT
C       STATE VECTOR OF AN ADSM.
C--INPUT PARAMETERS:
C        NCUR-COPY ROW NUMBER OF THE ADSM
C       IPAR-SYSTEM PARAMETER NUMBER (1.LE.IPAR.LE.21)
C--OUTPUT PARAMETERS:
C       SYSP-VALUE OF THE SYSTEM PARAMETER
```

```
      SUBROUTINE INIDLA(NDB,IRCF,IRCT,IDBAA,IERR,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       INIDLA WILL COPY A REAL DL ROW(COLUMN) TO ANOTHER ROW(COLUMN)
C       OF THE SAME DL
C
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING, 2-REFERENCE)
C       IRCF-ROW OR COLUMN TO BE COPIED. IF IRCF.GT.0, COPY ROW IRCF.
C            IF IRCF.LT.0, COPY COLUMN (-IRCF).
C       IRCT-ROW OR COLUMN TO BE COPIED INTO.
C--INPUT/OUTPUT PARAMETERS:
C       IDBAA(2)-DBAA OF THE DL
C
C--OUTPUT PARAMETERS:
C       IERR-ERROR CODE
C            0-NO ERROR
C            .NE.0-DBCS ERROR CODE
C--ALTERNATE RETURNS:
C       1-IERR.NE.0
```

```
      SUBROUTINE NCOPYA
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       NCOPYA WILL EXAMINE THE DATA BASE AND FILL THE NCOPY ARRAY.
C       COLUMNS 3 AND LFDB+4 HAVE ALREADY BEEN FILLED BY MSAINT.
C       NCOPYA ALSO FILLS THE NOPRI,NDB4,AND NMES ARRAYS
```

```
      SUBROUTINE NECCA (LABEL,NECC,*)
C
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C**PURPOSE:  THIS SUBROUTINE CODES A CHARACTER STRING INTO AN INTEGER
C            EQUVALENT BY CODING EACH CHARACTER IN THE STRING AS A
C            TWO DIGIT INTEGER VALUE.THE ONLY CHARACTERS THAT MAY BE
C            CODED ARE A-Z AND 0-9.THE LETTERS OF THE ALPHABET ARE
C            CODED ACCORDING TO THEIR ORDER (E.G. A IS CODED AS 01,D AS
C            04,F AS 06,ETC.).THE DIGITS 1,2,3,4,5,6,7,8,9,0 ARE A
C            ASSIGNED THE VALUES OF 27-36 RESPECTIVELY.
C
```

```
C**INPUT PARAMETERS:   LABEL=CHARACTER STRING TO BE CODED
C
C**OUTPUT PARAMETERS:  NECC=CODED INTEGER VALUE REPRESENTING THE
C                           CHARACTER STRING (LABEL)
C
C**ALTERNATE RETURNS:  THE SINGLE ALTERNATE RETURN OCCURS IF THERE IS
C                      NOT AN INDEX VALUE FOR A PARTICULAR CHARACTER.
C
```

```
      SUBROUTINE NFST1A
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       NFST1A IS CALLED IMMEDIATELY AFTER NCOPYA. IT FILLS THE
C       NSITE AND NFSTOR ARRAYS WITH DATA FOR THE AIR DEFENSE UNITS.
C       IT DOES NOT FILL NSITE FOR PROTECTED SITES OR VIEWERS.
C       NEITHER NSITE NOR NFSTOR CAN BE COMPLETELY FILLED HERE BECAUSE
C       OF REFERENCES TO PROTECTED SITES AND VIEWERS.
```

```
      SUBROUTINE NFST2A
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       NFST2A FILLS THE NSITE ARRAY WITH COLUMNS FOR REMOTE VIEWERS
```

```
      SUBROUTINE NFST3A
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       NFST3A WILL FINISH INITIALIZING THE NFSTOR ARRAY AND WILL LOAD
C       PROTECTED SITE DATA IN NSITE. IT ALSO LOADS OTHER VALUES
C       FROM RUN-DATA.
```

```
      SUBROUTINE POPSTA(IOPT,IDOP,NT,TASKS,NSL,*)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       POPSTA WILL POP TASK DATA OFF OF AN OPERATOR'S TASK STACK.
C       THE STACK IS A LAST-IN-FIRST-OUT STACK OF TASK DATA.
```

```
C--INPUT PARAMETERS:
C      IOPT-OPTION
C           1-GET TASK DATA FROM THE TASK STACK AND REMOVE IT
C             FROM THE STACK.
C           2-GET TASK DATA BUT LEAVE IT ON THE STACK(I.E. EXAMINE
C             THE STACK)
C      IDOP-OPERATOR ID NUMBER
C      NT-NUMBER OF COLUMNS OF THE TASKS ARRAY.  THE NEXT NT TASKS
C         ON THE STACK WILL BE RETURNED IN TASKS. TASKS(-,1) WILL
C         BE THE NEXT TASK TO BE PERFORMED.
C--OUTPUT PARAMETERS:
C      TASKS(5,NT)-ARRAY TO HOLD TASK DATA FROM THE STACK.  IF
C                  TASKS(-,1).LT.0,  THE  TASK WAS PREVIOUSLY INTERRUPTED
C                  EACH COLUMN OF TASKS CAN HOLD DATA FOR ONE TASK.
C      NSL-THE NUMBER OF TASKS RETURNED IN TASKS(). IF NSL.LT.NT, IT
C          MEANS THAT FEWER THAN NT TASKS ARE ON THE STACK.  NSL=0 IF
C          ALTERNATE RETURN 1 IS TAKEN.
C--ALTERNATE RETURNS:
C      1-STACK EMPTY

-------------------------------------------------------------------------

      SUBROUTINE PSHSTA(IDOP,TASKS,NT)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      PSHSTA WILL PUSH TASK DATA ONTO AN OPERATOR'S TASK STACK.
C      THE STACK IS A LAST-IN-FIRST-OUT STACK OF TASK DATA.
C--INPUT PARAMETERS:
C      IDOP-OPERATOR ID NUMBER
C      TASKS(5,NT)-ARRAY OF TASK DATA FOR THE STACK.  IF
C                  TASKS(-,1).LT.0, THE TASK IS BEING INTERRUPTED.
C                 TASKS(1,NT) WILL BECOME THE NEXT TASK TO BE PERFORMED.

-------------------------------------------------------------------------

      SUBROUTINE PTKDA(NCUR,NROW,ROW)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      PTKDA WILL FILL A ROW OF THE "TRACK-DATA" DATA LIST FOR A
C      SPECIFIED ABSM
C--INPUT PARAMETERS:
C      NCUR-ABSM COPY ROW NUMBER
C      NROW-ROW TO FILL
C      ROW(13)-VALUES TO PUT IN ROW NROW
```

```
      SUBROUTINE PTKCA(NCUR,ICODE,NPOINT,NCOL,VAL,IOK)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       PTKCA IS USED TO CHANGE A SINGLE ELEMENT OF A ROW IN THE
C       "TRACK DATA" DATA LIST OF AN ADSM.
C--INPUT PARAMETERS:
C       NCUR-ADSM COPY ROW NUMBER
C       ICODE-MEANING OF NPOINT
C             1-NPOINT IS A ROW NUMBER IN THE "TRACK DATA" DL(THIS
C               IS THE EFFICIENT WAY)
C             2-NPOINT IS A COLUMN NUMBER IN THE NTRACK ARRAY(THIS
C               IS THE INEFFICIENT WAY)
C      NPOINT-ROW NUMBER OR NTRACK COLUMN NUMBER (SEE ICODE)
C       NCOL-COLUMN NUMBER OF THE ROW TO CHANGE
C       VAL-NEW VALUE FOR THE NCOL-TH ELEMENT OF THE SPECIFIED
C           ROW.
C--OUTPUT PARAMETERS:
C       IOK-OUTPUT CONDITION
C           1-ALL OK
C           2-NCOL NOT VALID OR THE SPECIFIED ROW DOES NOT CONTAIN
C             TRACK DATA.
```

```
      SUBROUTINE PTSPA(NCUR,IPAR,SYSP)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       PTSPA WILL SET A SINGLE SYSTEM PARAMETER FROM THE ENVIRONMENT
C       STATE VECTOR OF AN ADSM.
C--INPUT PARAMETERS:
C        NCUR-COPY ROW NUMBER OF THE ADSM
C       IPAR-SYSTEM PARAMETER NUMBER (1.LE.IPAR.LE.21)
C       SYSP-VALUE OF THE SYSTEM PARAMETER
```

```
      SUBROUTINE PUTEVA(IDE,N,NEE,KE,XE)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C
C      PUTEVA SETS THE VALUES OF SPECIFIED HUMAN FACTORS ELEMENTS OF
C      ENVIRONMENTAL STATE VECTORS.
C
```

```
C--INPUT PARAMETERS:
C       IDE(N)-THE DBAA OF THE ESV
C       NEE(KE)-THE DESIRED ELEMENT NUMBERS. E.G. NEE(2)=7 IMPLIES THAT
C               HUMAN FACTORS ELEMENT 7 OF THE ESV IS TO BE SET.
C       XE(KE)-THE NEW VALUES FOR THE ELEMENTS SPECIFIED IN NEE.
C              E.G. IF NEE(2)=7, THEN XE(2)=THE VALUE OF HF ELEMENT 7.

------------------------------------------------------------------------

      SUBROUTINE PUTOVA(IDO,N,NEO,KO,XO)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       PUTOVA SETS THE VALUES OF SPECIFIED HUMAN FACTORS ELEMENTS OF
C       STATE VECTORS(OSV).
C--INPUT PARAMETERS:
C       IDO(N)-DBAA OF THE OSV
C       NEO(KO)-THE DESIRED ELEMENT NUMBERS. E.G. NEO(2)=7 IMPLIES THAT
C               HUMAN FACTORS ELEMENT 7 OF THE OSV IS TO BE SET.
C       XO(KO)-THE NEW VALUES OF THE HUMAN FACTORS ELEMENTS
C              SPECIFIED IN NEO. E.G. IF NEO(2)=7, THEN XO(2)=THE
C              VALUE OF HF ELEMENT 7.

------------------------------------------------------------------------

      SUBROUTINE PUTTSA(IOPT,ITASK,NTS,NTSK,NT,XTSK)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       PUTTSA SETS THE VALUES OF SPECIFIED TASK DATA ELEMENTS.
C       IT IS CALLED  TO SET
C       INFORMATION ABOUT THE TASK BEING PERFORMED.
C
C--INPUT PARAMETERS:
C       IOPT-OPTION
C            0-SETTING TASK TIME INFORMATION
C            1-SETTING HUMAN FACTORS DATA SPECIFIC TO THE TASK.
C            2-SETTING INFORMATION ON THE SKILLS REQUIRED TO
C              PERFORM THE TASK.
C            3-SETTING INFORMATION ON SYSTEM RESOURCES NEEDED.
C       ITASK(NTS)-THE DBAA OF THE TASK DATA LIST+TASK NUMBER.
C                  ITASK(1) & (2) ARE THE DBAA. ITASK(3)=NODE NUMBER.
C       NTSK(NT)-THE DESIRED ELEMENT NUMBERS. THERE ARE 4 CASES:
C                IOPT=0--NTSK NOT USED
C                IOPT=1--NTSK CONTAINS ELEMENT NUMBERS OF THE TASK
C                        SPECIFIC CATEGORIES.
C                IOPT=2--NTSK CONTAINS SKILL NUMBERS
C                IOPT=3--NTSK CONTAINS RESOURCE NUMBERS
```

```
C      XTSK(NT)-THE VALUES OF THE VARIABLES REQUESTED IN NTSK ABOVE.
C               4 CASES:
C               IOPT=0--XTSK(1) TO (3) ARE DISTRIBUTION TYPE, MEAN, AND
C                       STANDARD DEVIATION.(ALL MUST BE SET)
C               IOPT=1--IF(NTSK(2)=3, XTSK(2)=THE VALUE OF TASK
C                       SPECIFIC CATEGORY 3.
C               IOPT=2--IF(NTSK(2)=3, XTSK(2)=THE WEIGHT OF SKILL
C                       CATEGORY 3 REQUIRED TO PERFORM THE TASK.
C                       IF SKILL 3 IS NOT REQUIRED, SEND XTSK(2)=0.
C                       NEW SKILLS MAY NOT BE ADDED, BUT WEIGHTS OF
C                       EXISTING SKILLS MAY BE CHANGED.
C               IOPT=3--IF NTSK(2)=3, THEN XTSK(2)=RESOURCE PARAMETER
C                       FOR RESOURCE 3. ZERO IF RESOURCE 2 NOT NEEDED.
C                       NEW RESOURCES MAY NOT BE ADDED, BUT PARAMETERS
C                       OF EXISTING RESOURCES MAY BE CHANGED.

------------------------------------------------------------------------

      SUBROUTINE RECMSA(IOPT,IDOP,NDATA,MSAA,CHARS,DATA,MSID,NLEN)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      RECMSA WILL RETRIEVE A MESSAGE. SEE SUBROUTINES SNDMSA AND
C      FNDMSA FOR MORE INFORMATION.
C--INPUT PARAMETERS:
C      IOPT-OPTION
C           1-RETRIEVE MESSAGE AND DELETE IT FROM THE MESSAGE DIRECTORY
C           2-RETRIEVE MESSAGE BUT DO NOT DELETE IT( I.E. EXAMINE IT
C             ONLY)
C      IDOP-OPERATOR ID WHO IS TO RECEIVE THE MESSAGE
C      NDATA-LENGTH OF THE DATA ARRAY
C--INPUT/OUTPUT PARAMETERS:
C      MSAA(2)-DATA BASE ADDRESS OF THE MESSAGE. SEE FNDMSA.
C--OUTPUT PARAMETERS:
C      CHARS(10)-MESSAGE CHARACTERISTICS. SEE SNDMSA.
C      DATA(NDATA)-ARRAY TO HOLD MESSAGE
C      MSID(5)-MESSAGE ID
C      NLEN-MESSAGE LENGTH.  ELEMENTS 1 TO NLEN OF 'DATA' HOLD THE
C           MESSAGE. IF NLEN.LT.0, THEN THE MESSAGE IS TOO LONG
C           TO FIT IN 'DATA'.  -NLEN IS THE TOTAL MESSAGE LENGTH AND
C           THE FIRST NDATA ELEMENTS ARE RETURNED IN 'DATA'.
```

```
      SUBROUTINE RETAKA(IOPT,NCOP,NDATA,MSAA,CHARS,DATA,NLEN)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      RETAKA WILL RETRIEVE AN ACKNOWLEDGMENT MESSAGE. SEE SUBROUTINES
C       SVAKA AND FNDAKA FOR MORE INFORMATION.
C--INPUT PARAMETERS:
C      IOPT-OPTION
C           1-RETRIEVE MESSAGE AND DELETE IT FROM THE MESSAGE DIRECTORY
C           2-RETRIEVE MESSAGE BUT DO NOT DELETE IT( I.E. EXAMINE IT
C             ONLY)
C      NCOP-COPY ROW NUMBER OF ACKNOWLEDGMENT-MESSAGE DIRECTORY
C      NDATA-LENGTH OF THE DATA ARRAY
C--INPUT/OUTPUT PARAMETERS:
C      MSAA(2)-DATA BASE ADDRESS OF THE MESSAGE. SEE FNDAKA.
C--OUTPUT PARAMETERS:
C      CHARS(10)-MESSAGE CHARACTERISTICS. SEE SVAKA.
C      DATA(NDATA)-ARRAY TO HOLD MESSAGE
C      NLEN-MESSAGE LENGTH.  ELEMENTS 1 TO NLEN OF 'DATA' HOLD THE
C           MESSAGE. IF NLEN.LT.0, THEN THE MESSAGE IS TOO LONG
C           TO FIT IN 'DATA'.  -NLEN IS THE TOTAL MESSAGE LENGTH AND
C           THE FIRST NDATA ELEMENTS ARE RETURNED IN 'DATA'.
```

```
      SUBROUTINE SNDMSA(IDIST,MSID,DATA,NDATA,CHARS,MSGNO)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C
C     SNDMSA WILL SEND A MESSAGE TO ONE OR MORE OPERATORS
C--INPUT PARAMETERS:
C      IDIST-DISTRIBUTION CODE(SEE MSID)
C           0-SEND MESSAGE TO OPERATOR TYPE MSID(2) IN ALL ADSM'S
C           1-SEND ONLY TO SYSTEM AND OPERATOR SPECIFIED IN MSID(1)
C             AND (2)
C           2-SEND TO ALL NEXT LEVEL DESCENDENTS (TO OPERATOR TYPE
C             MSID(2))
C           3-SEND TO ALL DESCENDENTS(OPERATOR TYPE MSID(2))
C           4-SEND TO ADSM MSID(1) AND ALL OF ITS DESCENDENTS
C             (OPERATOR TYPE MSID(2))
```

```
C      MSID(5)-THE MESSAGE ID.  THE SENDER MUST LOAD THIS ARRAY.
C              MSID(1)-COPY ROW NUMBER OF RECEIVER (FOR IDIST=1 AND 4)
C                  (2)-OPERATOR TYPE OF RECEIVER (FOR ALL IDIST VALUES)
C                  (3)-FUNCTION TYPE (1-COMMAND,2-REQUEST FOR INFO,
C                      3-REPORT OF STATUS,4-ACKNOWLEDGMENT,5-
C                      COMPUTER GENERATED)
C                  (4)-MESSAGE SUBTYPE(APPLICATION DEPENDENT)
C                  (5)-MESSAGE PRIORITY
C
C              SNDMSA DOES NOT CHANGE THE VALUES OF MSID.
C
C      DATA(NDATA)-CONTENTS OF THE MESSAGE
C                  (NDATA MAY BE ZERO FOR A NULL MESSAGE.)
C
C--INPUT/OUTPUT PARAMETERS:
C      CHARS(10)-MESSAGE CHARACTERISTICS
C
C                CHARS (ALL INPUT BY THE USER UNLESS NOTED OTHERWISE)
C                ---------------------------------------------------
C                 1   1-VOICE, 2-ATDL
C                 2   ACKNOWLEDGMENT REQUIRED, 1-YES, 2-NO
C                 3   UNUSED
C                 4   ATDL MESSAGE CODE (CURRENTLY UNUSED)
C                 5   TIME MESSAGE WAS SENT (OUTPUT)
C                 6   MESSAGE NUMBER (OUTPUT)
C                 7   SENDER COPY ROW NUMBER
C                 8   SENDER OPERATOR TYPE
C                 9   UNUSED
C                10   UNUSED
C
C                SNDMSA DOES NOT CHECK THE INPUT VALUES OF CHARS.
C--OUTPUT PARAMETERS:
C      MSGNO-MESSAGE NUMBER. SAME AS CHARS(6)

-------------------------------------------------------------------------

      SUBROUTINE SVAKA(MSID,DATA,NDATA,CHARS)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C      SVAKA WILL SAVE A MESSAGE IN THE ACKNOWLEDGMENT-MESSAGES
C      DIRECTORY OF A WORKING ADSM. SVAKA ASSUMES THAT TNOW IS
C      THE RECEIVE-TIME FOR THE MESSAGE AND STORES IT IN CHARS(3).
```

```
C--INPUT PARAMETERS:
C       MSID(5)-THE MESSAGE ID
C              (1)-COPY ROW NUMBER OF THE ADSM THAT IS SAVING THE
C                  MESSAGE.
C              (2)-OPERATOR TYPE OF RECEIVER
C              (3)-FUNCTION TYPE(1-COMMAND,2-REQUEST FOR INFO,
C                  3-REPORT OF STATUS,4-ACKNOWLEDGMENT,5-COMPUTER
C                  GENERATED)
C              (4)-MESSAGE SUBTYPE
C              (5)-MESSAGE PRIORITY
C       DATA(NDATA)-CONTENTS OF MESSAGE. NDATA MAY BE ZERO FOR A
C                  NULL MESSAGE.
C--INPUT/OUTPUT PARAMETERS:
C       CHARS(10)-MESSAGE CHARACTERISICS
C                 CHARS (ALL INPUT BY THE USER UNLESS NOTED OTHERWISE)
C                 ---------------------------------------------------
C                   1   1-VOICE, 2-ATDL
C                   2   ACKNOWLEDGMENT REQUIRED, 1-YES, 2-NO
C                   3   TIME MESSAGE WAS RECEIVED(OUTPUT)
C                   4   ATDL MESSAGE CODE (CURRENTLY UNUSED)
C                   5   TIME MESSAGE WAS SENT
C                   6   MESSAGE NUMBER
C                   7   SENDER COPY ROW NUMBER
C                   8   SENDER OPERATOR TYPE
C                   9   UNUSED
C                  10   UNUSED
C
```

```
      FUNCTION TIMEA()
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       TIMEA WILL RETURN THE PRESENT SIMULATED CLOCK TIME IN
C       MILITARY UNITS. E.G. IF THE TIME IS 4:10 P.M., TIMEA=1610.
C       0000.LE.TIMEA.LE.2359. TIMEA IS ACCURATE TO ONLY THE
C       NEAREST MINUTE.
C
```

```
      SUBROUTINE TRAVA(NDB,IOPT,ICSQ,IDB12,ID12,IRSLT)
C--MODULE: MOPADS DATA BASE APPLICATION PROGRAMS
C--REFERENCE: MOPADS VOLUME 5.18
C--PURPOSE:
C       TRAVA WILL TRAVERSE A COMMAND AND CONTROL TREE AND RETURN
C       THE DRAA'S OF ALL CODE 12 DR'S.  IT IS CALLED REPEATEDLY
C       RETURNING A NEW CODE 12 DRAA EACH TIME UNTIL IT RETURNS AN
C       INDICATION THAT THE TREE IS EXHAUSTED. THE ORDER IN WHICH TRAJA
C       RETURNS CODE 12 DR'S IS NOT PREDICTABLE.
C--INPUT PARAMETERS:
C       NDB-DATA BASE (1-WORKING,2-REFERENCE)
C       IOPT-OPTION
C            1-INITIALIZE TRAVA FOR A NEW SEARCH OF THE TREE.  TRAVA
C              DOES RETURN A CODE 12 DRAA FROM THIS CALL IF ONE EXISTS
C              (SEE IRSLT)
C            2-LOCATE ANOTHER CODE 12 DR. A CALL WITH IOPT=1 MUST
C              BE MADE BEFORE CALLING WITH IOPT=2. THEN EACH SUBSEQUENT
C              CALL SHOULD HAVE IOPT=2.
C            3-DEACTIVATE TRAVA. A CALL WITH IOPT=3 IS NOT NECESSARY
C              IF IRSLT=2 ON A PREVIOUS CALL.
C--INPUT/OUTPUT PARAMETERS:
C       ICSQ(4)-DRAA OF THE COMMAND AND CONTROL (CODE 7) DR AT THE
C               ROOT OF THE TREE. TRAVA DOES NOT CHECK THAT ICSQ
C               IS ACTUALLY A CODE 7 DR.
C--OUTPUT PARAMETERS:
C       IDB12(4)-DRAA OF ANOTHER CODE 12 DR IF IRSLT=1. ZERO OTHERWISE.
C       ID12(4)-ID OF THE CODE 12 DR IN IDB12.  ZERO IF IRSLT=2.
C       IRSLT-RESULT
C             1-ANOTHER CODE 12 DR HAS BEEN FOUND. ITS DRAA AND ID ARE
C               IN IDB12 AND ID12, RESPECTIVELY.
C             2-THE CODE 12 TREE IS EXHAUSTED. IDB12 AND ID12 ARE ZERO.
C--LOCAL VARIABLES:
C       IDDL(2)-DBAA OF A DUMMY DL CREATED TO STORE CODE 12 DRAA'S
C               AND ID'S AS TRAVA TRAVERSES THE BRANCHES OF THE TREE.
```

VI. USER INSTRUCTIONS

1-0 SETTING DBAP OPTIONS

SUBROUTINE APOPTA is used to set internal options for the DBAP. At present, there is only one such option, the unit number of the MOPADS output file. MOPADS sets this to the line printer output file. DBAP will write error messages to this file.

2-0 RETRIEVING AIR SCENARIO DATA

SUBROUTINE ASROWA will retrieve the data for an aircraft initiation point or checkpoint from the data base. ASROWA is called only from the control system module.

3-0 MISCELLANEOUS DATA BASE OPERATORS

The DBAP has several programs that perform needed data base operations. These programs are explained below:

CCDLA	-	maintains the "Copy Counter" data list for a simulation data set (see Polito (1983b).
CPDLA	-	copies data lists from one directory to another
CPDRA	-	copies directories from one place in a directory to another
CREATA	-	creates an empty MOPADS data base
INIDLA	-	copies information within s single data list
NCOPYA, NFST1A, NFST2A, NFST3A	-	performs initialization of the MSAINT data structure from the data base file
TRAVA	-	finds all air defense units in a simulation data set

4-0 OPERATOR AND ENVIRONMENTAL STATE VECTORS AND TASK DATA

Several programs are available for accessing information in the operator state vectors, the environmental state vectors, and the task specific data. The programs are:

GETEVA	-	retrieve information from an environmental state vector
GETOVA	-	retrieve information from an operator state vector
GETTSA	-	retrieve information from the task data
GTSEA	-	retrieve a single element of an environmental state vector
PTSPA	-	change a single element of an environmental state vector
PUTEVA	-	change information in an environmental state vector
PUTOVA	-	change information in an operator state vector
PUTTSA	-	change information in the task data

5-0 MESSAGES

There are two types of messages, 1) original messages and 2) acknowledgement messages. The DBAP provides facilities to manage both types. The following programs are provided for original messages and acknowledgement messages.

5-1. Original Messages.

FNDMSA	-	locate an outstanding message that satifies specified criteria
RECMSA	-	receive a message
SNDMSA	-	send a message

5-2. Acknowledgement messages.

FNDAKA	-	locate an acknowledgement message that satisfies specified conditions
RETAKA	-	retrieve an acknowledgement message
SVAKA	-	save a message that requires acknowledgement

Acknowledgement messages are simply copies of original messages that require an acknowledgement. The operator saves these in his ACKNOWLEDGEMENT-MESSAGES directory for later reference. FNDAKA, RETAKA, and SVAKA operate on this directory.

6-0 TRACK DATA

Each air defense unit maintains information on tracks in its "Track Data" data list. The DBAP provides programs to manage this information.

FTKAA	-	finds an unused row in the "track data" data list
GTKAA	-	locate a track that satisfies specifiied conditions
GTKBA	-	retrive one row of track data
GTKCA	-	retrieve one element of a row of track data
PTKBA	-	change the contents of one row of track data
PTKCA	-	change one element of a row of track data

7-0 STACK OPERATIONS

The DBAP has programs to manipulate the operators' task stack:

CLRSTA	- empty one operator's task stack
POPSTA	- retrieve the last element from the stack
PSHSTA	- add an element to the end of the stack

8-0 GOALS

The DBAP has two programs dealing with operator goals:

CHKGLA	- check to see if a particular operator has a specified goal
EVLPA	- evaluate a goal priority function for a specified operator

9-0 MISCELLANEOUS FUNCTIONS

NECCA	- implements the fucntion NECC (Number Equivalent of a Character Code) that converts character strings to integers
CENCA	- performs the inverse NECC function. Converts an integer to a character string.
TIMEA	- returns the current time in military format (e.g., 1620)

10-0 ERROR PROCESSING

The error processing program in the DBAP is discussed in Section VII.

VII. ERROR PROCESSING

SUBROUTINE ERRORA processes run time errors for the DBAP and, in fact, is called by programs outside the DBAP, too. The parameters of ERRORA include a text message of arbitrary length, the calling subprogram name, and lists of integer and real parameters to print. Also, if a data base error occurred, the data base address of the offending directory or data list can be passed to ERRORA. ERRORA will attempt to print the contents of this data base entry.

ERRORA always terminates execution by performing a deliberate FORTRAN error. It does this so that Traceback information will be obtained (if it is available) to aid in debugging.

VIII. COMMON VARIABLE DEFINITIONS

There is only one labeled COMMON area in the DBAP. It is labeled COM1A, and it has only one variable.

Variable Name	Default Name	Definition
JOUTA	-	unit number of the file to write error information to

IX. REFERENCES

Polito, J. The MOPADS data base control system (MOPADS/DBCS) (MOPADS Vol. 5.13). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (a)

Polito, J. The MOPADS data base (MOPADS Vol. 5.17). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (b)

Polito, J. MOPADS free-format input programs (MOPADS/FFIN2) (MOPADS Vol. 5.14). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (c)

Polito, J. MOPADS user interface (MOPADS/UI) (MOPADS Vol. 5.12). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (d)

Polito, J. MOPADS architecture manual (MOPADS Vol. 4.1). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983 (e)

Polito, J. & Goodin, J. R. MOPADS utility programs (MOPADS Vol. 5.9). Pritsker & Associates, Inc., prepared for the U. S. Army Research Institute Field Unit, Ft. Bliss, TX, 1983.

X. DISTRIBUTION LIST

Dr. Mike Strub (5)
PERI-IB
U. S. Army Research Institute Field Unit
P. O. Box 6057
Ft. Bliss, Texas 79916

Pritsker & Associates, Inc. (5)
P. O. Box 8345
Albuquerque, New Mexico 87198

ACO-Loretta McIntire (2)
DCASMA (S1501A)
Bldg. #1, Fort Benjamin Harrison
Indianapolis, Indiana 46249

Mr. E. Whitaker (1)
Defense Supply Service-Washington
Room 1D-245
The Pentagon
Washington, DC 20310

Dr. K. R. Laughery (2)
Calspan Corporation
1327 Spruce St., Rm. 108
Boulder, Colorado 80301

XI. CHANGE NOTICES