THESIS

COMMUNICATIONS PROCESSOR FOR
C³ ANALYSIS AND WARGAMING

by

Lloyd Neil Clark
Larry Douglas Pless
Robert Lee Rapp

March 1982

Approved for public release; distribution unlimited.
# Communications Processor for C³ Analysis and Wargaming

Lloyd Neil Clark  
Larry Douglas Pless  
Robert Lee Rapp

Naval Postgraduate School  
Monterey, California 93940

Naval Postgraduate School  
Monterey, California 93940

**Approved for public release; distribution unlimited.**

**This thesis developed the software capability to allow the investigation of C³ problems, procedures and methodologies. The resultant communication model, while independent of a specific wargame, is currently implemented in conjunction with the McClintic Theater Model (MTM). It provides a computerized message handling system (C³ Model) which allows simulation of communication links (circuits) with user-definable delays; garble and low rates; and multiple circuit types, addressees, and levels of command.**

**Key Words:** C³ Model  
Wargame  
Communications Processor  
Circuit Parameters  
Message Garble  
Message Loss

---

**Abstract:**

This thesis developed the software capability to allow the investigation of C³ problems, procedures and methodologies. The resultant communication model, while independent of a specific wargame, is currently implemented in conjunction with the McClintic Theater Model (MTM). It provides a computerized message handling system (C³ Model) which allows simulation of communication links (circuits) with user-definable delays; garble and low rates; and multiple circuit types, addressees, and levels of command.
It is designed to be used for test and evaluation of command and control problems in the areas of organizational relationships, communication networks and procedures, and combat doctrine or tactics.
Communications Processor for C3 Analysis and Gating

by

Lloyd Neil Clark
Captain, United States Air Force
B.S., Ohio State University, 1975
M.A., Webster College, 1978

Larry Douglas Pless
Captain, United States Air Force
B.S., University of Missouri, 1972

Robert Lee Rapp
Major, United States Air Force
B.S., University of Louisville, 1967
M.S., University of Southern California, 1979

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS TECHNOLOGY
(COMMAND, CONTROL AND COMMUNICATIONS)

from the

NAVAL POSTGRADUATE SCHOOL

March 1982

Authors: Lloyd Neil Clark
Robert Lee Rapp

Approved by: Dr. Barry
Thesis Advisor

Co-advisor:

Michael O. Weaver
Chairman, C3 Academic Group

H. Henry
Academic Dean
ABSTRACT

This thesis developed the software capability to allow the investigation of C3 problems, procedures and methodologies. The resultant communication model, while independent of a specific wargame, is currently implemented in conjunction with the McClintic Theater Model (MTM). It provides a computerized message handling system (C3 Model) which allows simulation of communication links (circuits) with user-definable delays; garble and loss rates; and multiple circuit types, addressees, and levels of command. It is designed to be used for test and evaluation of command and control problems in the areas of organizational relationships, communication networks and procedures, and combat doctrine or tactics.
**TABLE OF CONTENTS**

I. INTRODUCTION--------------------------------------------- 9
II. GENERAL GAMING CONCEPTS------------------------------- 12
III. COMBINED OBJECTIVES AND TASKS--------------------- 15
IV. COMMUNICATIONS MODEL CAPABILITIES------------------ 19
V. WARGAMING CAPABILITIES
   A. MCCLINTIC THEATER MODEL----------------------------- 23
   B. ADAPTATION------------------------------------------ 27
VI. CAPABILITIES DEMONSTRATION
   A. SCENARIO--------------------------------------------- 29
   B. COMMAND/COMMUNICATIONS STRUCTURE------------------ 30
   C. RESULTS--------------------------------------------- 30
VII. CONCLUSION------------------------------------------- 34
VIII. RECOMMENDATIONS FOR FURTHER WORK------------------ 36

APPENDIX A: USER INSTRUCTIONS-------------------------- 40
APPENDIX B: CONTROLLER CHECKLIST------------------------ 50
APPENDIX C: DIALOGUE AND SAMPLE OUTPUT----------------- 55
APPENDIX D: C3 MODEL BLOCK DIAGRAM, MODULES AND FILES---- 86
APPENDIX E: C3 MODEL PROGRAM CODE----------------------- 93

LIST OF REFERENCES---------------------------------------- 160
INITIAL DISTRIBUTION LIST---------------------------------- 181
LIST OF FIGURES

1. MTM SUBROUTINE ORGANIZATION------------------------25
2. COMMAND CONFIGURATION (DEMO)------------------------31
3. C3 MODEL USER SCHEMATIC (PART 1)---------------------44
4. C3 MODEL USER SCHEMATIC (PART 2)---------------------45
5. C3 MODEL USER SCHEMATIC (PART 3)---------------------46
6. C3 MODEL USER SCHEMATIC (PART 4)---------------------47
7. C3 MODEL USER SCHEMATIC (PART 5)---------------------48
8. C3 MODEL USER SCHEMATIC (PART 6)---------------------49
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Full Form</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACCAT</td>
<td>Advanced Command &amp; Control Architectural Testbed</td>
</tr>
<tr>
<td>ARPANET</td>
<td>DOD Packet-Switched Computer Network</td>
</tr>
<tr>
<td>C2</td>
<td>Command and Control</td>
</tr>
<tr>
<td>C3</td>
<td>Command, Control &amp; Communications</td>
</tr>
<tr>
<td>CPU</td>
<td>Central Processing Unit</td>
</tr>
<tr>
<td>CR</td>
<td>Carriage Return</td>
</tr>
<tr>
<td>DCL</td>
<td>Digital Control Language</td>
</tr>
<tr>
<td>DEC</td>
<td>Digital Equipment Corporation</td>
</tr>
<tr>
<td>IBM</td>
<td>International Business Machines</td>
</tr>
<tr>
<td>JTIDS</td>
<td>Joint Tactical Information Distribution System</td>
</tr>
<tr>
<td>MTM</td>
<td>McClintic Theater Model</td>
</tr>
<tr>
<td>NOSC</td>
<td>Naval Ocean Systems Center</td>
</tr>
<tr>
<td>NPS</td>
<td>Naval Postgraduate School</td>
</tr>
<tr>
<td>TELNET</td>
<td>Telephone Network for Intercomputer Connection</td>
</tr>
<tr>
<td>VAX</td>
<td>DEC 11/780 Model Computer</td>
</tr>
<tr>
<td>VMS</td>
<td>Virtual Memory System</td>
</tr>
<tr>
<td>VT-100</td>
<td>DEC Video Terminal, Model 100</td>
</tr>
<tr>
<td>WES</td>
<td>Warfare Environmental Simulator</td>
</tr>
</tbody>
</table>
ACKNOWLEDGEMENTS

The authors wish to express their gratitude and appreciation to Professors Samuel H. Parry and F. Russell Richards for their faith and support in making this thesis possible, and to Dr. Al Zied, Mr. Albert Wong, and Mr. William Thomas for their help in solving many of the conversion problems associated with a new computer and a new operating system.

The authors would also like to thank Mr. Fred McClintic and the staff of the Wargaming Division, U.S. Army War College, for their assistance in transferring the McClintic Theater Model to the Naval Postgraduate School.
I. INTRODUCTION

One of the most vexatious problems facing Defense Department analysts is how to evaluate Command Control and Communications (C3) systems. While there are a myriad of models, both computerized and manual, available to evaluate the effectiveness of operational tactics and procedures, very few tools exist to aid in the analysis of command and communication structures.

An effort was made by Stack and Secorsky [Ref. 1] to provide such a tool at the Secure Command Control and Communication Exercise Laboratory. The product of their thesis was computer software for a user-defined communications network supporting a multi-level chain of command which allowed for communications delays, garbled messages, message loss and player attrition. This communications model used the Navy's Warfare Environmental Simulator (WES) as a gaming framework and was implemented on the DEC 11/70 computer in the Secure C3 laboratory at NPS.

The Stack/Secorsky model was only partially successful for two major reasons. First, the model allowed for only one communication channel between any two players. As a result, it is difficult to experiment with partial circuit losses or degradations. Second, the WES wargame model, besides having no capability to simulate ground units, is felt by many to be very unwieldy and time-consuming to set.
up, and difficult to play. Expenditures of more than 40 man-hours to set up a scenario are not uncommon. Additionally, WES is resident at the Naval Ocean Systems Center (NOSC) in San Diego. In order to run the combined models, it is necessary to link two computers via the TELNET facility on the secure ACCAT network.

The prime objective of this thesis is to remedy the above-mentioned shortcomings. The first of the problems was solved by implementing the capability for up to six user-selectable communication channels between players; the communications model was also freed from dependence upon the WES model. It is now possible to utilize the communication model with any wargame. Additionally, an improved capability to gather message traffic statistics for post-game analysis was developed. Secondly, a new wargame, the McClintic Theater Model (MTM), developed at the Army War College, was imported and implemented on the VAX 11/780 located in the Computer Science Department at NPS. MTM, in addition to providing the capability to simulate ground, air and naval surface units, is also much simpler to manage and play than the WES model.

The final output of this thesis is a combined software model providing a capability for C3 experimentation in the areas of organizational relationships, communication networks and procedures, link and circuit analysis, and combat doctrine and tactics.
This document contains five appendices designed to assist a new player to understand and use the C3 Model. Appendix A includes generalized player and controller instructions for set-up and play, along with schematics of the model program flow. Appendix B consists of a checklist and suggested worksheets to be used by the controller to compile data for entry into the model. Appendix C is an example user/computer dialogue, followed by several sample message inputs and outputs. Appendix D contains a block diagram of C3 Model module and file organization; and Appendix E includes the C3 Model computer code, as well as code for a HELP facility for MTM.
II. GENERAL GAMING CONCEPTS

There are several ways to test and evaluate combat scenarios, doctrine, and C3 systems. Perhaps the most realistic method makes use of field exercises with troops performing simulated combat under battlefield conditions. Field exercises, however, have many drawbacks. Many months (or years) of planning and coordination are required in preparation for an exercise of any size. Besides being enormously expensive to stage, data collection (a secondary consideration for participants whose primary concern is generally training) and analysis are difficult and time-consuming. In addition, replication and sensitivity testing are virtually impossible to achieve.

An alternative to massive field exercises is the use of wargame simulations. Many relatively sophisticated board-type games are available today; PEGASUS, developed by the U.S. Army, is an example of this type. These games involve commanders maneuvering their "forces" on a grid map in accordance with game rules; movements and combat results must be computed and tabulated manually. Additional artificiality is present because the game is played in "turns", each team responding to the other team's last move. A sense of time criticality is difficult to achieve under these conditions. The usefulness of such games is generally
limited to procedural training; they are virtually worthless as test and evaluation tools.

One step up from these board games are computer wargame simulations. These free the players from the clerical requirements involved with board games because movements and combat results are handled by the computer and effects reported to the players, who need only keep their displays current with the data base. The simulation is time rather than event-driven, and simultaneous inputs from both teams are possible. The fact that the combat algorithms within these games are generally stochastic in nature requires considerable testing against historical data in order to refine parameters before the results can be truly credible. Replicability of a single set of results is virtually impossible, but a statistical average of results from a number of runs becomes meaningful. Sensitivity testing on input parameters is possible only after the internal algorithms and parameters have been refined and validated.

From a C3 point of view, the authors were aware of no computer wargame simulation capable of modeling a Command Control structure and less-than-perfect communication systems which such a structure must utilize for the transfer of information. During game play with some wargames, members of each team (Red or Blue) are typically located within the same room, sharing the same intelligence data and "view" of the battle. Problems of coordination among
members of the command hierarchy and transmission of orders to lower level operational unit commanders are simplified to the point of triviality. Thus, these simulations fail to adequately reflect reality—commanders are often not collocated and may have different, sometimes conflicting, information. Real world commanders must rely on real world communications networks to share information, coordinate operations, and promulgate orders; and each of these activities is subject to degradation in the form of delay, garbling, loss, and errors of interpretation. It was toward developing a realistic model of these processes that the authors' efforts were directed.
III. COMBINED OBJECTIVES AND TASKS

The first task was to select a wargame on which it would be possible to overlay the proposed new C3 Model. Basic prerequisites for selection were:

1) ease of use and simplicity of set-up;

2) capability of simulating land, sea and air combat;

3) size compatible with current and/or projected mainframe computers at NPS.

The McClintic Theater Model met all of these requirements satisfactorily and the Army War College was willing to export this program to NFS. The second objective was to develop both MTM and the C3 Model for eventual implementation on a DEC VAX 11/780 computer to be installed in the Secure 'C3 Laboratory at some time in the future. Since the War College had plans for acquiring the same computer, the arrangement was felt to be mutually beneficial.

Since the size of MTM exceeded the CPU capacity of the computer currently installed in the C3 Lab, and, at the time MTM was received, the VAX had not yet been installed in the Computer Science Lab, the first implementation effort for the wargame was on the NPS IBM 3033 computer. This meant that almost every line of code (6000 lines) needed to be changed to convert the original Honeywell Fortran to IBM
Fortran. The conversion required over a month to complete, followed by several weeks to solve system interface problems due to differences in capabilities of the computers. However, an excellent understanding of the MTM algorithms was achieved during the conversion, which led to an expedited transfer to the VAX 780.

The third objective was to implement MTM on the VAX 780, build a workable Fortran communication model, and integrate the two. Care was taken to make only those changes to MTM which were required to accommodate characteristic differences between Fortran versions. The transfer of MTM proceeded smoothly and the VAX proved to be well-suited for the combined models.

Using the Stack-Secorsky model as a starting point, the C3 Model was developed on the VAX using Digital Control Language (DCL). DCL is a high level language feature of the VAX/VMS operating system which facilitates the writing of interactive programs that operate very close to the system level.

The direct translation of the Stack/Secorsky algorithms (written in the "C" programming language) to Fortran was considered unsatisfactory because they did not provide all the desired features and lacked the needed flexibility. A new Fortran message handler was therefore written to simulate:
1) delays based on MTM battle factor (ratio of battle time to real time),

2) lost messages, and

3) message garbling.

An additional feature built into the message handler was the ability to operate virtually independent of a particular wargame. The only changes required to use it with a different game are those required because of different file names and access methods.

The final objective was to integrate the new communication model with MTM, to ensure software and procedural compatibility, and to demonstrate the full combined capabilities of the two models for evaluation of C3 characteristics. Since the integration requirements were designed into the message handler, the actual integration of the models was relatively simple.

In the combined model there are three levels of communication participants: those who will be communicating directly with the MTM wargame, those who are part of the command hierarchy, and the overall game controller (umpire). The one player on each team who interfaces directly with MTM is also a part of the command and communication structure in that he receives orders from commanders and inputs them into the game. All other Red or Blue players participate only by
sending and receiving messages via the C3 Model and can neither input orders nor receive MTM outputs directly.

On-line documentation in the form of a HELP facility was added to assist users in entering commands into both the C3 Model and into the wargame. A total of thirteen VAX directories are available for play—ten for players at the C3 Model level and one each for the Red and Blue game players plus one for the game controller. More directories may easily be added as needs grow and additional terminals become available.
IV. COMMUNICATIONS MODEL CAPABILITIES

The command structure and communication links (circuits) to be used during a game are defined by the game controller prior to game start. (Specific instructions are contained in Appendix A.) The controller specifies player game names and associated VAX directory names as well as the communication circuits (up to six) available between each player pair. In addition, the following link characteristics are defined for each link type:

1. Message arrival and service rates to be used in the delay (queuing) algorithm. Standard or controller-determined rates may be used.

2. Probability of message garbling.


It is possible to change these characteristics as well as the available circuits during game play. Players may also be removed, temporarily or permanently, to simulate communication circuit failure and restoration.

The algorithm to determine message delay times is basically that used by the Stack-Secorsky model.

"The queuing time represents the amount of time a message will be delayed in arriving at its destination and is calculated from the single-server queuing theory equation:

$$W_C = \frac{A}{S(S-A)}$$

where

19
WQ is the average queuing time; 
A is the average message arrival rate; 
S is the average message service rate.

Unscheduled arrival rates for messages are viewed as conforming to a Poisson distribution and the inter-arrival times between these messages as following a negative exponential distribution. During game play the program transforms a uniformly distributed random number \((y)\) to an exponentially distributed random number \((x)\) using the relationship:

\[ x = (-1/L) \ln(y) \]

where \(L\) is the average message arrival rate specified by the experimenter. ...The arrival times for unscheduled arrivals are described by the following relationships:

\[ f(y) = \frac{A}{e^{Ay}} \]

Expected value\([y]\) = \(1/A\) and Variance\([y]\) = \(1/A\)

where \(A\) is the mean of the message arrival [time]. The service time is also negative exponentially distributed with a mean of \(1/S\) where \(S\) is the average service rate. For the above equations to hold, an infinite population of messages must be assumed. It is also necessary that the ratio \(A/S\) be less than 1 or the queue and the waiting times will increase without bound." [Ref.1]

Since computed delay times are based on the system (computer) clock, and the MTM game time factor is capable of being varied by the controller during game play, this computed time is adjusted to reflect the current ratio of battle time to real time being used.

Message loss and garbling are determined by use of a pseudo-random number generator within the message handler and are based on a strict comparison with the probabilities.
and rates specified by the controller. Garbling is performed on a character-by-character basis.

As soon as a message is sent by a player, an undelayed and unadulterated copy is sent to the controller's directory. Additionally, the controller receives one copy of each message as it is received by each addressee. This message copy also contains statistics which show the results of the random number comparisons mentioned above as well as the actual delay time. These message copies are stored in the controller's file for post-game analysis.

Some concessions must be made to play the combined game in its present implementation due to the numbers and locations of VAX terminals. Red players are all collocated in the same room, and are not players in the communications model. Red interfaces only with the MTM game through a hard-copy Miniterm via a telephone link to the VAX. Blue players are all collocated in a different room, with the various commanders utilizing six VT-100 terminals for passing message traffic among themselves and orders to the Blue player interfacing with MTM. This Blue player is also linked to the VAX through a hard-copy Miniterm via phone link.

It is realized that these conditions are less than ideal for several reasons. First, a certain degree of artificiality is introduced by requiring collocated Blue players to communicate by message traffic rather than
verbally. Second, each player shares a common view of the battle with all his teammates. Ideally, players should be broken up into logical groups representing command and staff elements, each with its own game board and each capable of communicating with other team elements as well as interfacing with MTM. An example would be several elements of Red and Blue staffs at the Army War College, Air War College and NPS playing the combined game via ARPANET. Only minor modifications to the C3 Model should be necessary to accommodate this mode of play.
V. WARGAMING CAPABILITIES

A. McClintic Theater Model

During the design of the McClintic Theater Model, an area of primary concern was keeping the game play simple so that it could be easily used by players lacking in computer experience. This was accomplished by using free-form keyword inputs. Thus, alignment and spacing are unimportant when entering orders. Because keyword searching is used, the order of input is also unimportant and the computer will ignore words that it doesn't recognize rather than considering them to be errors. This allows an almost English-like interface dialogue, and the player can spend more time using the model and less time learning to use it.

Each input is checked for validity before action is attempted by the model. If an invalid order is entered, it is ignored and an error message is printed immediately. Such errors as moving enemy units, firing beyond an artillery's range, moving a unit that is out of POL or initiating a nuclear or chemical strike without controller permission are prevented by this feature. However, tactical blunders such as attacking friendly units or stumbling into one's own mine fields are permitted.

The model is based on a variable-size hexagonal grid network which is applicable to any part of the world. A small interactive data generation program is used to define
a new scenario. A new data file can be created in a day or less (after data collection) by simply answering questions asked by the program.

The program is modular in structure and is organized as depicted in Fig 1. Each block is a separate subroutine within the program, simplifying algorithm changes to more accurately reflect reality or to examine specific functions in more detail. Parameters within the data base or the modules can easily be changed by the controller during game play by entering the CONTROL subroutine. The game is time driven vice event driven to improve realism. Status reports on units and logistics are available upon demand, while intelligence reports are generated at specified intervals or upon request.

MTM was designed to serve as an evaluation model for Corps commanders in evaluating and modifying strategies and tactics in the Tactical Command Readiness Program and to aid in the training of senior officers at the Army War College. The game is a high level game for division and larger size forces and thus some generalizations were required in the details of combat and other factors applicable to forces of this size. The MTM algorithms are not detailed enough to permit evaluation of the micro elements of battle. However, the model does take into account virtually every macro aspect of battle and does a reasonable job of integrating all of the elements together into a single model. MTM
Figure 1

MN SUBROUTINE ORGANIZATION
models Red and Blue forces equally well with all features available to both sides.

MTM is capable of modeling any sized combat unit from a single ship or aircraft up to a fleet or a wing of aircraft. Attack and tactical aircraft, as well as utility and transport aircraft and helicopters, are permitted. Naval units can also sea-lift ground units and supplies. Ground units are limited to division and battalion size for practical reasons, since only 300 units can be modeled at a time. The model also has a restart capability to allow the user to recover if there is a computer system failure or play is extended over several days.

Movement of ground and sea units is permitted up to and including the maximum allowable speed set for that unit. Ground unit movement is negatively affected by terrain and barriers such as mountains, forests, cities, rivers, bridged rivers, anti-tank ditches and minefields. Roads act positively by increasing movement speed. Actual routing is computed to take the shortest, quickest and safest track through the terrain and barriers.

The model handles the placement and clearing of minefields, air defense artillery, artillery, air-to-air combat, twelve classes of supplies and electronic warfare. Nuclear, chemical and biological, as well as conventional, attacks are available, but permission is required from the game controller for either team to exercise these options.
B. ADAPTATION

As mentioned above, the structure of MTM has been preserved as much as possible. The changes that have been made were generally to accommodate Fortran version differences. All of the original MTM subroutines, functions and variables have been retained and, from the user's standpoint, the version being run at the Army War College and the version at NPS are identical.

One feature which has been added to the combined models has been the inclusion of MTM Volume II (Users Manual) on-line as a HELP facility. By entering the word "HELP", the Red or Blue player will be shown a menu of allowable MTM commands. By then entering the name of one of these commands, he will be shown either an abbreviated (format only) or verbose description of that command. This feature should help reduce training/familiarization time.

There are two MTM subroutines which have been suppressed in the current NPS implementation. The first of these is the TALLY subroutine which gathers controller-specified game statistics as a function of time. The statistics are then displayable via the graphics capabilities provided with MTM. TALLY is not used because no compatible graphics terminals were immediately available during implementation efforts at NPS and, perhaps more importantly, the statistics-gathering appeared to slow down game processing on the VAX, particularly when high battle time factors were being used.
and other users were sharing the CPU. This omission is not a significant drawback at this point because the game is serving primarily as a framework for C3 experimentation.

The second suppressed subroutine—LOGISTICS—does have a major impact on communications experimentation. This subroutine is not used because the additional players which would be required to handle resupply, airlift and sealift problems exceeds the number of terminals available. Thus, a major portion of the message traffic which would normally be stressing the communications circuits does not occur. This is the one major shortcoming in the present implementation of the combined models.
VI. CAPABILITIES DEMONSTRATION

A relatively informal wargame utilizing the combined models was conducted on March 12, 1982 for the purpose of demonstrating the capabilities of both MTM and the C3 Model. Participants in this wargame included professors from the Operations Research, Electrical Engineering and C3 curricula; systems specialists from the Computer Science Department; and contractor representatives from Jet Propulsion Laboratory, who will eventually be responsible for software maintenance in the Secure C3 Laboratory.

A. SCENARIO

The game scenario involved a simulated invasion of Central Europe by Warsaw Pact forces. The invasion force included 69 tank, motorized rifle and infantry divisions and 21 air wings. Defending were NATO forces which included 65 infantry, armor and mechanized divisions; airborne brigades and air cavalry regiments; and 16 air wings.

Initial force dispositions were in accordance with the standard MTM NATO unclassified data base. Major PACT force concentrations and points of attack occurred at the Fulda, Cheb and Highway 12 Gaps. The mission of the Blue force was to delay the advance of the invading force as long as possible to allow for the arrival of U.S. and other NATO reinforcements.
B. COMMAND/COMMUNICATIONS STRUCTURE

The Blue command structure modeled for the demonstration is shown by the portion of Figure 2 outlined by the dashed line. The air and ground units under the operational control of NORTHAG, CENTAG, 2ATAF and 4ATAF were assigned arbitrarily using the center line of the playing board as the dividing line. The Red team consisted of only two players and was located in a separate room with its own display board.

Initially each Blue player was given a complete communication capability of all six possible circuit types with all other Blue players. Circuit traffic loading was set to "normal" at the start of the game and increased to "heavy" as play progressed. The initial circuit parameters, as well as sample message inputs and outputs are included in Appendix C. These parameters as well as the availability of circuits between player pairs, were degraded over time to reflect combat effects on communications.

C. RESULTS

An initial briefing was followed by a period of user familiarization and experimentation with the C3 Model constructs, MTM orders and the VT-100 terminals. After this familiarization period, the Blue players appeared to begin to feel comfortable with combined model play. (It seems to take approximately two to three hours of fairly concentrated
FIGURE 2

COMMAND CONFIGURATION (ZEMC)
There were two problem areas which surfaced during the demonstration.

The first of these problems related to the implementation of the models on the VAX. When the demonstration began, another user had two long-term processes running on the system batch queue, which had only a four-job capacity. Since the C3 Model uses the same batch queue to store messages which have been processed by the message handler but are awaiting expiration of the computed delay period, this meant that only two pending messages could exist at any one time. Additional messages were rejected by the queue and discarded. This problem was solved during the demonstration by the creation of a new batch queue dedicated to the C3 Model message handler. The new queue has a capacity of 50 pending messages and the capacity can easily be increased by a system operator if required.

A second problem developed during the familiarization period when the players began to focus their attention on the MTM game to the exclusion of the communications model; coordination and planning were performed by direct verbal communications among players, thus bypassing the communications portion of the game. This problem was not totally unexpected and was a result of all Blue players being collocated, with only one game board among them. The problem could have been alleviated to some degree by
requiring inputs from various commanders to the Blue player interfacing with MTM to take the form of messages via the C3 Model, rather than using preformatted order forms.

As a result of this demonstration, it is apparent that, while the combined model functions basically as desired, it will be difficult to use for experimentation in the mode of play used for the demonstration; maintaining artificial barriers to communication is virtually impossible. The usefulness of the model will be realized only after computer terminal availability allows Blue and/or Red team members to play on a physically separated, distributed network basis. (See Section F of Chapter VIII.)
VII. CONCLUSION

The authors feel that, by and large, the goals which were established when this thesis was first discussed have been achieved. A new and easier-to-play (than WES) wargame is now fully functional on both the IBM 3633 and the DEC VAX11/780. Sufficient game materials have been procured to support a two-sided (NATC vs. Warsaw Pact) game using the Central European data base. Another unclassified MTM data base is currently available for Southwest Asia as well as a classified NATC data base; it is only necessary to acquire the associated maps, overlays, and game pieces to exercise scenarios in these arenas.

The C3 Model developed as part of this thesis provides a flexible and adaptable tool for C3 experimentation and analysis. Virtually any command structure and its associated communications network can be modeled for experimentation, subject to hardware limitations. Within the communications model, circuit parameters may be adjusted to simulate any possible combination of circuit types. While only six circuits are included between players in this initial version of the model, it would be a simple matter to increase this number to 256—the limiting factor being the controller's ability to effectively manage this number of links (See Chapter VIII).
The one initial objective which was not met due to time constraints was to completely validate model performance and perform experimentation and analysis using the combined models. Validation is a critical step in any model development and must be performed prior to any serious experimentation.

The authors strongly recommend that a close working relationship be established and maintained between NPS and the Army War College to assure configuration control of the MTM game; this area has long been a problem with the WES model. It is also strongly urged that the Army War College expedite publication of Volume IV (Software Description) of the MTM Manual. This volume is critical if programmers are to be thoroughly familiar with the algorithms and be able to suggest improvements to the game.
VIII. RECOMMENDATIONS FOR FURTHER WORK

The work done in this thesis should be viewed as only a beginning and not as a project that has been or will ever be completed. An experimenter now has a tool needed to begin to build the data base for analysis. The following areas, however, require additional development and testing.

A. CIRCUIT PARAMETERS

The circuit parameters used in the current C3 Model are estimates only and are not based on actual communications circuit data. Actual circuit parameters are available from the service communications commands on circuit qualities for various types of circuits. Qualities are typically expressed as error rates, average message delivery times under different load conditions (messages per hour) and message retransmission rates, which loosely convert to lost message rates. With this information in hand, different circuit configurations should be catalogued for use in game analysis and then tested under varying game conditions to determine circuit sensitivity to extremes of loading and parameter variation.

B. COMMAND STRUCTURES

Wiring diagrams are also needed to show how command structures are tied together and organized for Tactical and Strategic C2. Determining how commands are organized for C2
may be a very difficult undertaking. Peacetime and wartime configurations are often very different, yet both must be understood and modeled if improvements are to be made. Once this has been accomplished, the communications links should be overlayed onto the command structure to represent the C3 configurations. With the circuit parameters and command structures, the C3 model can be loaded to represent actual C3 systems with their inherent advantages and faults.

C. PCST GAME ANALYSIS

A post-game analysis feature is needed for the C3 model. Messages and individual message data are collected by the model software, but no provisions have been made to collect the cumulative message data and to do any on-line data analysis. The information is already available with the current model (delay times, percent garbling, lost message values, circuit numbers, etc.) so that, as each message is sent, data could be extracted and stored. Once the data is stored, it could easily be called up and game statistics calculated, freeing the experimenter from a lengthy post-game data reduction. A methodology for combining all available measures of effectiveness into a single Effectiveness Index should be developed to allow comparisons between configurations.
D. GRAPHICS

The MTM wargame model needs a graphics package which provides more display capability than the current version. Commanders need to see views of the battlefield with differing degrees of resolution. The graphics capabilities exist at NPS to achieve this capability, both in hardware and software development support.

E. LINK MANAGEMENT

As mentioned in the previous chapter, the capability to specify up to 256 circuits between players is a relatively simple matter to implement in software. More than six circuits between players will be required if the model is to simulate complicated switched networks realistically. However, procedures and/or software must be developed to enable the controller to manage this number of circuits effectively.

F. MTM INTERFACE

As implemented on the VAX, the combined model allows only one Red, one Blue and one Controller to be interfacing with MTM at any time. All orders from each team must thus be "funneled" through this single input point to MTM. Besides an almost unmanageable workload on this player during periods of intense play, the inability of a commander to order his forces directly contributes significantly to a sense of artificiality.
The original implementation on the IBM 3033 allowed for multiple points of interface with MTM on each side. MTM responses to orders were channeled to the player from which the input was received. In effect, this allowed independence among commanders and would allow the game to be played in a distributed mode. The VAX version of MTM did not contain this feature due to time constraints. Addition of the player distribution feature should be among the first priorities for further work on the combined model.

G. IMPROVED CIRCUIT MODELING

It is desirable to have the capability to simulate packet-switched circuits with multiple nodes, such as JTIDS. Provision for node deletion and automatic message forwarding should be included.
A. GENERAL

A total of 14 VAX directories are currently dedicated to the C3 Model; names are as shown in Figure 3. The CLARK directory contains the entire model, including MTM; it also collects copies of message traffic during game play. (Controller: See Note 1.) Figures 3 through 8 of this appendix are schematic representations of the C3 Model and depict how a player or controller moves about among various sections of the model by using the various commands shown between brackets (< >).

B. CONTROLLER PREPARATION

Prior to the start of play, a controller must log onto the computer and enter the BUILD/MODIFY section of the model in order to define the command structure and communication network to be used for the game. Data preparation worksheets and general instructions for this purpose are provided in Appendix B. A controller who is unfamiliar with these procedures will find Figures 3 and 5 in Appendix A helpful. The BUILD/MODIFY portion of the model is interactive: the controller need only respond with menu selections and values as requested by the program. Parameters entered in this section of the model are stored
in various files of the CLARK directory for use during play (see Appendix D). MTM startup is also done from this section using INITIALIZE.

Passwords are requested at various points in the program for entry into certain parts of the model. A copy of this thesis with passwords annotated on Figures 3 and 5 will be available for qualified controllers in the C3 Curricular Office.

The data base which MTM uses is named [CLARK]WARDAT.DAT. This file is read by the game upon initialization and is periodically updated during play. It is this feature which allows MTM to be stopped at any time and resumed at that same point later. If it is desired to start a new game from time zero, the appropriate data file, either standard (NATO=NATO000.DAT, IRAN=SWA0000.DAT.) or built/modified using the DATA section of INITIATE, must be copied into [CLARK]WARDAT.DAT prior to initializing MTM. Creating a new data base or making changes to an existing data base is done interactively by appropriately responding to the questions asked by the program.

C. PLAYER INSTRUCTIONS

The schematics shown in Figures 4, 5 and 8 of this appendix are applicable to game players. Game names and associated directories, as well as directory and game passwords, must be provided by the controller prior to play.
Once logged into the appropriate directory on a VT-100 terminal, entry of "C3MODEL" gives access to the game.

Once in the model, the program will ask for two passwords. Red, Blue and MTM Controller will enter the MTM password when requested. All other players enter a carriage return <CR>. All players enter only a carriage return when asked for the controller password. At this point, the program will branch each player to the appropriate point in the program. A list of valid commands (except for MTM Controller) is available by typing "HELP". A description of controller commands is available in the MTM Manual, Volume III (Controller).

There are two idiosyncrasies of the VAX which must be discussed. When MAIL is commanded, the program branches to the system mailer (prompt = MAIL>) and previously unexamined messages may be viewed sequentially by depressing RETURN. Exit from this mode is accomplished by typing "EX" followed by a RETURN. When SENDMSG is commanded, entries are made into a preformatted message header (TO:, FROM:, SUBJECT:) followed by a request for a circuit type over which the traffic is to be sent. At this point, the system text editor is invoked and the message text entered by the player. Once text entry is completed, depress PF1 followed by 7 on the numeric keypad. This will cause "COMMAND:" to appear at the bottom of the screen. Typing "EX" followed by the ENTER key on the keypad exits the editor. The message
will be automatically sent and the program will branch back to the command level. If "QUIT" is used instead of "EX", the latest previously sent message will be forwarded to the current addressees.

D. DATA ANALYSIS FILE

Message copies are collected during play and stored in [CLARK]MAIL.MAI. This file may be reviewed for both post-game analysis as well as during play to immediately assess the impact of changes to communication circuits and parameters.

Note 1: Because execution of the MTM portion of the model will halt if the disk quota of the directory from which MTM was initiated is exceeded, it is recommended that the CLARK directory not be used (logged into) for active play. If MTM execution halts for this reason (a printout describing the cause of the halt is provided in the main computer room each time execution stops), the offending directory must be purged and MTM restarted.
VI.

DEDICATED DIRECTORIES

[CLARK]
[BEN]
[BLUE]
[CONT]
[WG1]
[WG2]
.
.
[WG10]

CALL GAME

ENTER PASSWORDS

PRIVILEGES

COMM PLAYER ONLY

COMM CONTROLLER, PLAYER

MIN CONTROLLER, PLAYER
COMM PLAYER

COMM CONTROLLER, PLAYER
MIN CONTROLLER, PLAYER

FIGURE 3

C3 MODEL USER SCHEMATIC (PART I)
<MAIL> VIEW UNEXAMINED MESSAGES
USE <FX> TO EXIT THIS MODE

SENDMSG> MESSAGE HEADER/TEXT EDITOR
USE <PF1>, <Y> (NUMERIC KEYPAD), <FX> TO EXIT

READMSG> VIEW ENTIRE MESSAGE FILE
AUTOMATIC EXIT

HELP>

QUIT>
SYSTEM

FIGURE 4
C3 MODEL USER SCHEMATIC (PART 2)
<BUILD> ENTER NO. OF PLAYERS, PLAYER NAMES, LINKS

<MODIFY> F

<PLAY> A

<STATUS> DISPLAYS ALL COMM PARAMETERS

<INITIATE> STARTS UP MTM WARGAME

<INITIATE MTM> <BUILD> CREATE NEW OR MODIFY OLD MTM DATABASE (PASSWORD REC'D)

<HELP>

<QUIT> §

<DONE> B

FIGURE 2

C3 MODEL USER SCHEMATIC (PART 3)
<0> CHANGE COMM LINKS BETWEEN PLAYERS

<1> SET MESSAGE ARRIVAL/SERVICE RATES

<2> SET PROBABILITY OF MESSAGE GARBILING

<3> SET MESSAGE LOSS RATE

<4> SET RATE OF MESSAGE (CHARACTER) GARBILING

<5> REMOVE COMM PLAYERS FROM THE GAME

<6> RETURN MAIN MENU W/O IMPLEMENTING CHANGES

<7> IMPLEMENT CHANGES, RETURN MAIN MENU

FIGURE 6

C3 MODEL USER SCHEMATIC (PART 4)
APPENDIX B

C3 MODEL

CONTROLLER CHECKLIST

I. COMMUNICATIONS:

STEP 1:

DESCRIBE THE BASIC COMMUNICATIONS PATHS BETWEEN UNITS, WORKSHEET 1.

STEP 2:

ENTER CIRCUIT CHARACTERISTICS IN WORKSHEET 2A.

STEP 3:

ENTER RESERVE AND/OR REMOVED PLAYER DATA IN WORKSHEET 2B.

STEP 4:

YOU NOW HAVE ALL THE NECESSARY DATA TO BUILD THE C3 MODEL.

LOG INTO ONE OF THE GAME DIRECTORIES AND ISSUE THE COMMAND "C3MODEL".

STEP 5:

DISREGARD THE MTM GAME PASSWORD, ENTER <CR>. 52
HOWEVER, DO ENTER THE CONTROLLER'S PASSWORD WHEN PROMPTED: "ENTER <CR> TO CONTINUE".

SELECT THE BUILD (B) PORTION OF THE C3 MODEL AND ENTER DATA FROM WORKSHEET 1.

STEP 6:

SELECT THE MODIFY (M) PORTION OF THE C3 MODEL AND ENTER THE DATA FROM WORKSHEETS 2A AND 2B AS IT CORRESPONDS TO ITEMS 1 THROUGH 5.

REMEMBER: EXECUTE ITEM #7 WHEN FINISHED WITH MODIFY.

YOU ARE NOW READY TO USE THE C3 MODEL.
### BASIC COMMUNICATIONS PATHS

**WORKSHEET 1**

**TOTAL NUMBER OF PLAYERS (INCLUDING CONTROLLER):**

<table>
<thead>
<tr>
<th>PLAYER NAME</th>
<th>FROM/TO</th>
<th>CONTROLLER</th>
<th>PLAYER1</th>
<th>PLAYER2</th>
<th>PLAYER3</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTROLLER</td>
<td>XXXXX</td>
<td>1101100</td>
<td>111000</td>
<td>001110</td>
<td></td>
</tr>
<tr>
<td>PLAYER 1</td>
<td>101100</td>
<td>XXXXX</td>
<td>001110</td>
<td>000111</td>
<td></td>
</tr>
<tr>
<td>PLAYER 2</td>
<td></td>
<td>XXXXX</td>
<td></td>
<td>XXXXX</td>
<td></td>
</tr>
<tr>
<td>PLAYER 3</td>
<td></td>
<td></td>
<td></td>
<td>XXXXX</td>
<td>XXXXX</td>
</tr>
</tbody>
</table>

**NOTE:** XXXXX REPRESENTS NO ENTRY
ENTER A SIX DIGIT NUMBER CONSISTING OF 0'S AND 1'S TO INDICATE CIRCUIT AVAILABILITY BETWEEN PLAYER PAIRS IN THE POSITIONAL ORDER SHOWN BELOW. (I.E. 100010 REPRESENTS AN ENCRYPTED LINE AND A VOICE CIRCUIT)

1 -- ENCRYPTED LANDLINE
2 -- NON-ENCRYPTED LANDLINE
3 -- DIGITAL RF WITH ANTI-JAM
4 -- ANALOG RF WITHOUT ANTI-JAM
5 -- VOICE
6 -- PERFECT LINK
CIRCUIT PARAMETERS/PLAYER STATUS

WORKSHEET 2

A. CIRCUIT PARAMETERS.

<table>
<thead>
<tr>
<th>CIRCUIT</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>ARRIVAL RATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SERVICE RATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PROBABILITY MESSAGE GARBLED</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LOSS RATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GARBLE RATE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

B. PLAYERS HELD IN RESERVE OR REMOVED FROM GAME

NUMBER OF RESERVED/REMOVED PLAYERS: ____

RESERVED/REMOVED PLAYER NAMES.

1.
2.
3.
4.

etc.
APPENDIX C

DIALOGUE AND SAMPLE OUTPUTS

This appendix is divided into two sections. Section A is part of the dialogue which took place between the computer and a user (player or controller) when the demonstration scenario data was entered into the C3 Model; inputs are shown within brackets (< >). Note that not all sections of the C3 Model are entered in this dialogue, but only those needed to support the scenario structure. Entry into other areas of the C3 Model is relatively simple using the program prompting (requests for inputs) and the instructions contained in Appendix A. Section B contains some of the actual input messages and the resulting outputs of the C3 Model during the game.
A. USER DIALOGUE

<C3MODEL>

WELCOME TO THE NAVAL POSTGRADUATE SCHOOL C3 WARGAME

THIS GAME IS THE RESULT OF A THESIS FOR THE C-3 CURRICULUM
BY CAPTAIN LLOYD CLARK, CAPTAIN LARRY PLESS AND MAJOR BOB RAPP.

IT REPRESENTS AN IMPROVED ADAPTATION OF A STRUCTURED COMMUNICATIONS
MODEL FIRST DEVELOPED BY LT COL TOM STACK AND LCDR TOM SECORSKY.

THE WARGAME ITSELF IS THE OSCINTIC THEATER MODEL (MTM) DEVELOPED
AT THE US ARMY WAR COLLEGE.

***************************

MTM PLAYERS ENTER GAME PASSWORD. (ENTER <CR> IF UNKNOWN): <CR>
CONTROLLER ENTER PASSWORD (ENTER <CR> IF UNKNOWN): <PWD>
WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS

BUILD (B)
MODIFY (M)
PLAY (P)
STATUS (S)
INITIATE MTM (I)
QUIT (Q)

ENTER B, M, P, S, I OR Q: <B>

THIS PORTION OF THE PROGRAM WILL ALLOW THE CONTROLLER TO:

1. DESIGN THE ORGANIZATIONAL STRUCTURE FOR THE GAME
2. DESIGN THE COMM NET SUPPORTING THE ORGANIZATION

HOW MANY PLAYERS, INCLUDING THE CONTROLLER, ARE THERE? <U>

THE NUMBER OF PLAYERS IS: 8
WHAT ARE THE (GAME) NAMES OF THE PLAYERS IN THE GAME?
ENTER <CR> WHEN FINISHED

PLAYER 1: CONTROLLER
WHAT IS PLAYER 1 VAX DIRECTORY NAME?: WC1
PLAYER 2: AFCEMT
WHAT IS PLAYER 2 VAX DIRECTORY NAME?: WC2
PLAYER 3: NORTHAG
WHAT IS PLAYER 3 VAX DIRECTORY NAME?: WC3
PLAYER 4: AAWECE
WHAT IS PLAYER 4 VAX DIRECTORY NAME?: WC4
PLAYER 5: CENTAG
WHAT IS PLAYER 5 VAX DIRECTORY NAME?: WC5
PLAYER 6: 4ATAF
WHAT IS PLAYER 6 VAX DIRECTORY NAME?: WC6
PLAYER 7: 2ATAF
WHAT IS PLAYER 7 VAX DIRECTORY NAME?: WC7
PLAYER 8: BLUE
WHAT IS PLAYER 8 VAX DIRECTORY NAME?: BLUE

YOU HAVE NAMED 8 PLAYERS

IF YOU WISH EACH PLAYER TO HAVE A COMPLETE COMMUNICATION
SUITE (I.E., TO BE ABLE TO COMMUNICATE DIRECTLY WITH EVERY
OTHER PLAYER VIA ANY LINK), TYPE "ALL". OTHERWISE, TYPE <CR>.

CHOICE: <ALL>

RETURN TO MAIN MENU BY DEPRESSING <CR>: <CR>
WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS?

BUILD (B)
MODIFY (M)
PLAY (P)
STATUS (S)
INITIATE MTM (I)
QUIT (Q)

ENTER B, M, P, S, I OR Q: <M>

PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLING.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLING.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)
# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU. (STATUS DISPLAYED)
#NOTE#: CHANCES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0, 1, 2, 3, 4, 5, 6 OR 7: <0>

YOU WISH TO CHANGE THE COMMUNICATION LINK CODE BETWEEN TWO GAME PLAYERS.

CHANGE LINK CODE FROM PLAYER (NAME): <AFCENT>
TO PLAYER: <AAFCENT>

ENTER A SIX DIGIT NUMBER CONSISTING OF 0'S AND 1'S TO INDICATE CIRCUIT AVAILABILITY BETWEEN PLAYER PAIRS IN THE POSITIONAL ORDER SHOWN BELOW (I.E., 100010 REPRESENTS AN ENCRYPTED LANDLINE AND A VOICE CIRCUIT).

1 -- ENCRYPTED LANDLINE (AUTODIN)
2 -- NON-ENCRYPTED LANDLINE (TELETYPE)
3 -- DIGITAL RF CIRCUIT (HF/VHF/UHF) WITH A/J
4 -- ANALOG RF WITHOUT A/J
5 -- VOICE (TELEPHONE)
6 -- PERFECT LINK (NO DELAY, NO DEGRADATION)

123456 THE CURRENT LINK CODE IS:
111111

ENTER NEW LINK CODE: <111000>
PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLING.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLING.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
   (STATUS DISPLAYED)

#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0,1,2,3,4,5,6 OR 7 : <1>
Based on the queuing algorithm for single server facilities, the average amount of transmission delay for messages addressed to a given facility (WQ) can be expressed as a function of the average message arrival rate (A) and the average message service rate (S).

\[ WQ = \frac{A}{S(S-A)} \]

You may specify the actual arrival rates and service rates for each class of communications (1 through 5) or you may rely upon a pre-established service rate and arrival rate relationship and vary only the message arrival rate by requesting:

A. Normal Traffic
B. Medium Traffic (twice the normal arrival rate)
C. Heavy Traffic (three times the normal rate)

Type "1" to insert specific arrival and service rates.
Type "2" to use the general rates (normal, medium, heavy).

Note: If you do not wish to change values currently set -- type <CR>.

Enter 1, 2, or <CR> : <2>
The pre-established relationship between arrival and service rates for a normal arrival rate is as follows:

For Circuit Type 1: \( S = 3.01 \)

For Circuit Type 2: \( S = 3.05 \)

For Circuit Type 3: \( S = 3.04 \)

For Circuit Type 4: \( S = 3.03 \)

For Circuit Type 5: \( S = 3.02 \)

Enter "normal", "medium" or "heavy" to establish the initial message arrival rates.

Changes to these values which are desired during the game should be made by re-entering "modify".

Normal, medium or heavy?: (Normal)
PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLED.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLED.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
(STATUS DISPLAYED)

#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0,1,2,3,4,5,6 OR 7 : <2>,

ENTER THE PROBABILITY THAT A MESSAGE WILL BE GARbled DURING TRANSMISSION. FIVE EQUALS FIVE PERCENT. USE INTEGER VALUES.

TYPE "1" IF YOU WISH TO ENTER A SEPARATE PROBABILITY FOR EACH CIRCUIT TYPE. (1-5)

TYPE "2" IF YOU WANT A STANDARD PROBABILITY FOR ALL CIRCUITS.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>

ENTER THE MESSAGE GARble PROBABILITY FOR EACH CIRCUIT

FOR CIRCUIT TYPE [ ]:

PROBABILITY =

OR:

ENTER THE STANDARD PROBABILITY FOR ALL CIRCUITS

PROBABILITY =

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY".

ENTER 1, 2, OR <CR> : <1>
ENTER THE MESSAGE GARBLE PROBABILITY FOR EACH CIRCUIT

FOR CIRCUIT TYPE [ 1 ]:
PROBABILITY = <5>

FOR CIRCUIT TYPE [ 2 ]:
PROBABILITY = <5>

FOR CIRCUIT TYPE [ 3 ]:
PROBABILITY = <10>

FOR CIRCUIT TYPE [ 4 ]:
PROBABILITY = <7>

FOR CIRCUIT TYPE [ 5 ]:
PROBABILITY = <6>

PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLING.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLING.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)
# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0, 1, 2, 3, 4, 5, 6 OR 7 : <3>

ENTER THE RATE FOR MESSAGES TO BE "LOST". FIVE EQUALS FIVE PERCENT. USE INTEGER VALUES.

ENTER "1" IF YOU WISH TO ENTER A SEPARATE LOSS RATE FOR EACH CIRCUIT TYPE. (1-5)

ENTER "2" IF YOU WANT A STANDARD RATE FOR ALL CIRCUITS.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>

ENTER THE LOSS RATE BY COMM CIRCUIT TYPE FOR CIRCUIT TYPE [ ]:

LOSS RATE =

OR:

ENTER THE STANDARD RATE FOR ALL CIRCUITS

LOSS RATE =

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY".

ENTER 1, 2, OR <CR> : <1>
ENTER THE LOSS RATE FOR COMM CIRCUIT TYPE [1]
LOSS RATE = 6
ENTER THE LOSS RATE FOR COMM CIRCUIT TYPE [2]
LOSS RATE = 6
ENTER THE LOSS RATE FOR COMM CIRCUIT TYPE [3]
LOSS RATE = 8
ENTER THE LOSS RATE FOR COMM CIRCUIT TYPE [4]
LOSS RATE = 8
ENTER THE LOSS RATE FOR COMM CIRCUIT TYPE [5]
LOSS RATE = 3

PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLED.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLED.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0,1,2,3,4,5,6 OR 7: <4>

ENTER THE RATE AT WHICH YOU WISH MESSAGES TO BE GARBLED DURING TRANSMISSION. FIVE EQUALS FIVE PERCENT. USE INTEGER VALUES.

TYPE "1" IF YOU WISH TO ENTER A SEPARATE RATE FOR EACH CIRCUIT TYPE. (1-5)

TYPE "2" IF YOU WANT A STANDARD RATE FOR ALL CIRCUITS.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>

ENTER THE MESSAGE GARBLE RATE FOR EACH CIRCUIT

FOR CIRCUIT TYPE [ ]:

GARBLE RATE =

OR:

ENTER THE STANDARD RATE FOR ALL CIRCUITS

GARBLE RATE =
CHANGES TO THESE VALUES WHICH ARE DESIRED DURING THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY".

ENTER 1, 2, OR <CR> : <1>

ENTER THE MESSAGE GARBLE RATE FOR EACH CIRCUIT

FOR CIRCUIT TYPE [ 1 ]
RATE = 3

FOR CIRCUIT TYPE [ 2 ]
RATE = 5

FOR CIRCUIT TYPE [ 3 ]
RATE = 10

FOR CIRCUIT TYPE [ 4 ]
RATE = 10

FOR CIRCUIT TYPE [ 5 ]
RATE = 10
PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE
   USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLING.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RATE OF MESSAGE GARBLING.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
    (STATUS DISPLAYED)

#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7
       IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY
       RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0,1,2,3,4,5,6 OR 7 : <5>
ENTER THE NUMBER OF PLAYERS AND THE NAME OF EACH
PLAYER WHO HAS BEEN "DESTROYED" OR WHO FOR SOME
REASON IS TO BE REMOVED FROM THE GAME AFTER GAME
START. ENTER THE NUMBER AND A <CR> AND THE NAME
OF EACH PLAYER WITH A <CR> FOLLOWING EACH NAME.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES
CURRENTLY SET -- TYPE <CR>.

NUMBER OF PLAYER(S) TO BE REMOVED = <1>
PLAYER NAME : <NORTHAG>

REMOVED PLAYER IS: NORTHAG
PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

0. CHANGE COMMUNICATION LINKS BETWEEN TWO PLAYERS.
1. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE
   USED FOR EACH TYPE LINK.
2. SET PROBABILITY OF MESSAGE GARBLING.
3. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
4. SET RARE OF MESSAGE GARBLING.
5. REMOVE PLAYERS FROM THE GAME.
6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
     (STATUS DISPLAYED)

#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7
         IS REQUESTED. THIS ALLOWS YOU TO RICOVER BY SIMPLY
         RETURNING TO THE MAIN MENU (ITEM #6)

ENTER 0, 1, 2, 3, 4, 5, 6 OR 7: <?>
DISPLAY GAME STATUS

GAME STATUS AS OF 15-MARCH-1982 21:40:44.94

PLAYER1 : CONTROLLER IN DIRECTORY WG1
PLAYER2 : A#CENT IN DIRECTORY WG2
PLAYER3 : NORTHAG IN DIRECTORY WG3
PLAYER4 : AA$CE IN DIRECTORY WG4
PLAYER5 : GENTAG IN DIRECTORY WG5
PLAYER6 : 4ATAF IN DIRECTORY WG6
PLAYER7 : 2ATAF IN DIRECTORY WG7
PLAYER8 : BLUE IN DIRECTORY BLUE

LINK1T02 -------> 111111
LINK1T03 -------> 111111
LINK1T04 -------> 111111
LINK1T05 -------> 111111
LINK1T06 -------> 111111
LINK1T07 -------> 111111
LINK1T08 -------> 111111
LINK2T01 -------> 111111
LINK2T03 -------> 111111
LINK2T04 -------> 111100  ****NOTE CHANGE
LINK2T05 -------> 111111
LINK2T06 -------> 111111
LINK2T07 -------> 111111
LINK2T08 -------> 111111
LINK3T01 -------> 111111
LINK3T02 -------> 111111
LINK3T04 -------> 111111
LINK3T05 -------> 111111
LINK3T06 -------> 111111
LINK3T07 -------> 111111
LINK3T06 -------> 111111
LINK4T01 -------> 111111
LINK4T02 -------> 111111
<table>
<thead>
<tr>
<th>Arrival Rate for Circuit</th>
<th>Messages per Minute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.0</td>
</tr>
<tr>
<td>2</td>
<td>3.0</td>
</tr>
<tr>
<td>3</td>
<td>3.0</td>
</tr>
<tr>
<td>4</td>
<td>3.0</td>
</tr>
<tr>
<td>5</td>
<td>3.0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Service Rate for Circuit</th>
<th>Messages per Minute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9.30</td>
</tr>
<tr>
<td>2</td>
<td>9.15</td>
</tr>
<tr>
<td>3</td>
<td>9.12</td>
</tr>
<tr>
<td>4</td>
<td>9.09</td>
</tr>
<tr>
<td>5</td>
<td>9.06</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Loss Rate for Circuit</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6%</td>
</tr>
<tr>
<td>2</td>
<td>12%</td>
</tr>
<tr>
<td>3</td>
<td>15%</td>
</tr>
<tr>
<td>4</td>
<td>22%</td>
</tr>
<tr>
<td>5</td>
<td>10%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Garble Rate for Circuit</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6%</td>
</tr>
<tr>
<td>2</td>
<td>10%</td>
</tr>
<tr>
<td>3</td>
<td>12%</td>
</tr>
<tr>
<td>4</td>
<td>17%</td>
</tr>
<tr>
<td>5</td>
<td>13%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Garble Probability for Circuit</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8%</td>
</tr>
<tr>
<td>2</td>
<td>12%</td>
</tr>
<tr>
<td>3</td>
<td>17%</td>
</tr>
<tr>
<td>4</td>
<td>22%</td>
</tr>
<tr>
<td>5</td>
<td>15%</td>
</tr>
</tbody>
</table>

The following players have been removed:
1. Northag
WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS?

BUILD (B)
MODIFY (M)
PLAY (P)
INITIATE MTM (I)
STATUS (S)
QUIT (Q)

ENTER B,M,P,S,I OR Q: P

WELCOME TO THE NAVAL POSTGRADUATE SCHOOL C3 WARGAME

HERE IS A LIST OF ALLOWABLE COMMANDS AND THEIR DEFINITIONS

MAIL READ INCOMING MAIL
SENDMSG SEND A MESSAGE
READMSG READ ENTIRE MESSAGE FILE
HELP PRINTS THIS HELP MESSAGE
QUIT EXITS ENTIRE PROGRAM

COMMANDER, WHAT IS YOUR COMMAND? <SENDMSG>

TO: <AFCF1, 4ATAF, 2ATAF>
FROM: <AFCENT>

SELECT CIRCUIT TYPE 0,1,2,3,4,5,6 OR 7 <3>

SUBJECT: <INTELLREP>

<INTELLIGENCE SOURCES HAVE DETECTED ENEMY FORCE CONCENTRATIONS BEGINNING TO DEVELOP IN THE VICINITY OF FULDA GAP. SUGGEST PLACING ALL FORCES IN READY ALERT STATUS.>

<PF1, ?, EX, ENTER>
AAFGA --- OK
AAFGA --- OK
AAFGA --- OK

COMMANDER, WHAT IS YOUR COMMAND? <QUIT>

WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS?

BUILD (B)
MODIFY (M)
PLAY (P)
INITIATE MTM (I)
STATUS (S)
QUIT (Q)

ENTER B, M, P, S, I OR Q: Q

$
B. SAMPLE INPUT/OUTPUT

1. PHASE ONE

Traffic loading was set to NORMAL and all parameters were set to their initial values. The transmitted message appeared as illustrated below:

FROM CONTROLLER
TO ARGENT, NORTHAG, AAYCE, CENTAC

SUBJECT: TEST 7.1
THIS MESSAGE CONSTITUTES A TEST OF THE COMMAND, CONTROL AND COMMUNICATIONS SYSTEM. ALL ADDRESSEES RESPOND WITH THE SPECIAL CODE CONTAINED IN THIS MESSAGE.

SPECIAL CODE: 7777

*NOTE* THIS MESSAGE WILL BE USED TO DETERMINE DEGRADATION OF C3 AT DIFFERENT TIMES DURING BATTLE EVOLUTION.

TRANSMIT REPLY VIA CIRCUIT: 2

Copies of the received messages are stored in [CLARK]MAIL.MAI for post game analysis. The original message, as transmitted, is concatenated with the message statistical data as one copy; another copy of the message is received as it appears to the addressees. Different addressees will receive different copies of the same transmitted message due to variations in circuits between
different locations. The analysis copy of the above message appears as follows:

From: WC1
To: CLARK
Subj: MSG

VOICE
11-MAR-1982 12:35

FM CONTROLLER
TO AFGENT, NORTHAG, AAFCE, CENTAG

BY
SUBJECT: TEST 7.1
THIS MESSAGE CONSTITUTES A TEST OF THE COMMAND, CONTROL AND COMMUNICATIONS SYSTEM. ALL ADDRESSEES RESPOND WITH THE SPECIAL CODE CONTAINED IN THIS MESSAGE.

SPECIAL CODE: 7777

*NOTE* THIS MESSAGE WILL BE USED TO DETERMINE DEGRADATION OF C3 AT DIFFERENT TIMES DURING BATTLE EVOLUTION.

TRANSMIT REPLY VIA CIRCUIT: 2

MESSAGE DELAY TIME FOLLOWS:
0:00:01.00
THIS MESSAGE ADDRESSED TO: WG2

If URAN = 0.1037263 LESS THAN PGARB 5.999999E-02 GARBLE IT
GARBLE RATE 5 = 0.1000000
LOSS RATE 5 = 2.999999E-02
LINK TYPE = 5
LINK CODE = 111111
L.CKT = 5
SERVICE RATE = 29.35450
ARRIVAL RATE = 19.65381
FACTOR = 33.00000
LINK TYPE = 5
DELAY = 2.780150E-04 HOURS
STANDARDIZED REAL TIME DELAY = 8.340555E-03 HOURS

The next example shows the analysis copy of a lost message and since the message is lost this is the only place it will appear.

From: WG1
To: CLARK
Subj: MSG

VOICE
11-MAR-1982 13:05

To: CONTROLLER

FROM: CONTROLLER,AAFCF1,2ATAF
BT
SUBJECT: TEST 7.2
THIS MESSAGE CONSTITUTES A TEST OF THE COMMAND, CONTROL AND COMMUNICATIONS SYSTEM. ALL ADDRESSEES RESPOND WITH THE SPECIAL CODE CONTAINED IN THIS MESSAGE.

SPECIAL CODE: 7772

*NOTE* THIS MESSAGE WILL BE USED TO DETERMINE DEGRADATION OF C3 AT DIFFERENT TIMES DURING BATTLE EVOLUTION.

TRANSMIT REPLY VIA CIRCUIT: 5

THIS MESSAGE ADDRESSED TO: WG7

\[ \text{IF URAN} = 5.2703142E-02 \text{ LESS THAN } P(\text{LOSS}) = 7.9999999E-02 \text{ MESSAGE IS LOST} \]
\[ \text{GARBLE RATE} \quad 4 = 0.1000000 \]
\[ \text{LOSS RATE} \quad 4 = 7.9999999E-02 \]
\[ \text{LINK TYPE} = \quad 4 \]
\[ \text{LINK CODE} = 111111 \]
\[ \text{L_CKLT} = \quad 4 \]

There were no garbled messages in phase one due primarily to the low message volume and the low probability of garbling.
2. PHASE TWO

The next messages represent the before and after results of further circuit degradation after traffic loading was increased to heavy and the parameters of loss, garbling probabilities and garbling rates were all increased to simulate a jamming environment. The first message is the ungarbled analysis copy.

From: WGB
To: CLARK
Subj: MSG

DIGITAL RX CIRCUIT WITHOUT A/J

11-MAR-1982 13:46:11.77

RN: CENTAG
TO: AFCE, NORTAG

BT
SUBJECT: TEST 8.3
THIS MESSAGE CONSTITUTES A TEST OF THE COMMAND, CONTROL AND COMMUNICATIONS SYSTEM. ALL ADDRESSEES RESPOND WITH THE SPECIAL CODE CONTAINED IN THIS MESSAGE.

SPECIAL CODE: 8683
*NOTE* THIS MESSAGE WILL BE USED TO DETERMINE DEGRADATION OF C3 AT DIFFERENT TIMES DURING BATTLE EVOLUTION.

TRANSMIT REPLY VIA CIRCUIT: 5

MESSAGE GARbled

MESSAGE DELAY TIME FOLLOWS:
0:00:01.00

THIS MESSAGE ADDRESSED TO: W64

IF URAN  = 5.2703142E-02 LESS THAN PGARB 0.2200000 CAREBLY IT
GARble RATE 4 = 0.1700000
ICSS RATE 4 = 0.2200000
LINK TYPE = 4
LINK CODE = 111111
I.CKT = 4
SERVICE RATE = 17.60469
ARRIVAL RATE = 5.810129
FACTOR = 35.2999
LINK TYPE = 4
DELAY = 9.327614E-04 HOURS
STANDARDIZED REAL TIME DELAY = 2.7981764E-02

The second message is the garbled version that would have been received by the addressee.
VOICY

11-MAR-1982 13:46:11.77

FM CENTAG
TO AAFCE, NORTHAG

BT
SUBJE-T: TEST 8.3S G

COMMUNICATIONS SYSUEL
SPECK-L CODE CONTENTED 15% LAWS MESSAGE.

AL CODE: 8863 SIS. 0 Q . 2 J 2 P A B 3 E" CP / W
. 1 97O K IE . 8 0 Q Q 5 B

* N - TE74TH

COSTAGE WILL BE USED TO DETERMINE DEHDR; DA. IG OF 4 5 G K? UI
C3 AS EIFFQREW T EJMES DQING BATOLF IV LUTICH.

T3 N S M I T DJPLLYZVWA*BIRC

T: 5 2 E P; WP ( A

V N 8 J 0 C " 1 A 7 X G *
FILES USED BY EACH MODULE:

1. [CLARK.CCC]MAIN.COM
   - [CLARK]PARAMETER.DAT
   - [CLARK]INTERFACE.DAT
   - [CLARK]DATA1.TMP
   - [CLARK]DATA1_TMP
   - DOCUMENT.DAT
   - STATUS.DAT

2. [CLARK.CCC]RTSSS.COM
   - [CLARK.CCC]MSG.TXT
   - MSGOUT.TXT
   - MESSAGE.TXT

3. [CLARK.CCC]SENDMSG.COM
   - HEADER.TXT

4. [CLARK.CCC]ACTMAIL.COM
   - MAIL.MAI

5. [CLARK.CCC]MTHELP.COM
   - NONE

6. [CLARK.CCC]CKMSG.COM
   - MAILING_TMP
   - MAILING.DAT

7. [CLARK.CCC]MSGHAN.FOR
   - [CLARK]FACTOR
   - [CLARK]PARAMETER.DAT
   - [CLARK.CCC]LETTER.TXT
   - MAILING_TMP
   - MSGOUT.TXT
   - MESSAGE.TXT
   - SEED0.DAT
   - SEED1.DAT
   - SEED2.DAT
   - SEED3.DAT
   - SEED4.DAT
   - SEED5.DAT
   - SEED6.DAT
SEED7.DAT
STAT1.TXT
STAT2.TXT
STAT3.TXT
STAT4.TXT
STAT5.TXT
STAT6.TXT
STAT7.TXT
STAT8.TXT
STAT9.TXT
GARBL1.TXT
GARBL2.TXT
GARBL3.TXT
GARBL4.TXT
GARBL5.TXT
GARBL6.TXT
GARBL7.TXT
GARBL8.TXT
GARBL9.TXT

6. CREATE.FOR

WARDATA.DAT
NEWDATA.DAT

9. [CLARK.CCC]RTCCC.COM

[CLARK.CCC]MSG.TXT
MSGOUT.TXT
MESSAGE.TXT

************
NOTES ON ORGANIZATION AND STRUCTURE OF THE C3 COMMUNICATIONS REQUIREMENTS PROGRAM (CRP), MAIN.CCM.

1. ALL COMMON MODULES ARE IN THE FOLLOWING DIRECTORY OR SUB-DIRECTORY.
   [CLARK]
   [CLARK.CCC]

2. MAIN PROGRAM.
   [CLARK.CCC]MAIN.COM (MAIN PROGRAM)

3. DESCRIPTION:

   THE MAIN MODULE FIRST CLEARS THE LOCAL SYMBOL TABLE. THIS IS SIMPLY A HOUSE-KEEPING MEASURE (STARTING WITH A CLEAN SLATE).

   THE INITIALIZE SECTION READS IN PREVIOUSLY DEFINED GAME DATA FROM [CLARK]PARAMETER.DAT

   DATA IN THIS FILE IS IN THE FOLLOWING FORMAT:

   ARRIVAL_RATE1
   ARRIVAL_RATE2
   ARRIVAL_RATE3
   ARRIVAL_RATE4
   ARRIVAL_RATE5
   SERVICE_RATE1
   SERVICE_RATE2
   SERVICE_RATE3
   SERVICE_RATE4
   SERVICE_RATE5
   LOSS_RATE1
   LOSS_RATE2
   LOSS_RATE3
   LOSS_RATE4
   LOSS_RATE5
   GARBLE_RATE1
   GARBLE_RATE2
   GARBLE_RATE3
   GARBLE_RATE4
   GARBLE_RATE5
   PGARB1
   PGARB2
   PGARB3
   PGARB4
   PGARB5
   RM PLAYER_CNT

   ***********************
   * NUMBERS 1-5 REFER TO *
   * THE CIRCUIT LINETYPE *
   ***********************
   (PERCENTAGE OF GARBLED TEXT OF MESSAGES THAT ARE TO BE GARBLED.)
   (PROBABILITY THAT A MESSAGE WILL BE GARBLED.)
   (NUMBER OF REMOVED PLAYERS)
NAME OF REMOVED PLAYER (IF ANY)
NAME OF REMOVED PLAYER (IF ANY)
NAME OF REMOVED PLAYER (IF ANY)

RM_PLAYER_CNT NUMBER OF NAMES LISTED

NAME OF REMOVED PLAYER (IF ANY)
END OF FILE

IT ALSO READS GAME DATA FROM A SECOND FILE [CLARK] INTERFACE.DAT

DATA IN THIS FILE IS IN THE FOLLOWING FORMAT.

PLAYCNT (NUMBER OF PLAYERS IN THE GAME)
PLAYER1 REAL NAME
PLAYER1 VAX DIRECTORY NAME
PLAYER2 REAL NAME
PLAYER2 VAX DIRECTORY NAME

. THERE WILL BE 'PLAYCNT' NAMES LISTED

. LINK1TO2
. LINK1TO3 (IF THREE PLAYERS AND SO ON UNTIL PLAYCNT)
LINC2TO1
LINC2TO3 (IF THREE PLAYERS AND SO ON UNTIL PLAYCNT)

. PLAYER DO NOT HAVE LINKS TO THEMSELVES

END_OF_FILE

IN ADDITION, THIS SECTION EXTRACTS THE LOGIN NAME OF THE PLAYER WHICH IS USED LATER TO DETERMINE THE PLAYER'S ELIGIBILITY TO PLAY THE GAME.

THE MODULE REQUESTS THE PLAYER TO ENTER ALL KNOWN PASSWORDS AND PRIVILEGE CODES.
THE PRIVILEGE CODE GIVES THE USER GAME PARAMETER MODIFICATION/CHANGE PRIVILEGES

IF THE PLAYER IS NOT PRIVILEGED, THEN THE PROGRAM BRANCHES TO PROCEDURE [CLARK.CCC]RTCCC.COM (IF USER IS A COMMUNICATIONS-ONLY PLAYER) OR TO THE MCCLINTIC THEATER MODEL INPUT/OUTPUT PROGRAM.

IF THE PLAYER IS PRIVILEGED, THEN HE HAS THE OPTION OF SELECTING THE BUILD, MODIFY, STATUS OR PLAY PORTION OF THE PROGRAM.

***************
NOTES ON ORGANIZATION AND STRUCTURE OF THE C3 COMMUNICATIONS REQUIREMENTS PROGRAM (CRP) COMMUNICATIONS PORTION ONLY.

1. ALL COMMON MODULES ARE IN THE FOLLOWING DIRECTORY OR SUB-DIRECTORY.

[CLARK]
[CLARK.CCC]

2. MAIN PROGRAM.

[CLARK.CCC]MAIN1.COM (MAIN PROGRAM)

DESCRIPTION:

THE MAIN1 MODULE FIRST CLEAR THE LOCAL SYMBOL TABLE. THIS IS SIMPLY A HOUSEKEEPING MEASURE (STARTING WITH A CLEAN SLATE).

THE INITIALIZE SECTION READS IN PREVIOUSLY DEFINED GAME DATA FROM [CLARK]PARAMETER.DAT DATA IN THIS FILE IS IN THE FOLLOWING FORMAT:

ARRIVAL_RATE1
ARRIVAL_RATE2
ARRIVAL_RATE3
ARRIVAL_RATE4
ARRIVAL_RATE5
SERVICE_RATE1
SERVICE_RATE2
SERVICE_RATE3
SERVICE_RATE4
SERVICE_RATE5
LOSS_RATE1
LOSS_RATE2
LOSS_RATE3
LOSS_RATE4
LOSS_RATE5
GARBLE_RATE1
GARBLE_RATE2
GARBLE_RATE3
GARBLE_RATE4
GARBLE_RATE5
PGARBLE1
PGARBLE2
PGARBLE3
PGARBLE4
PGARBLE5

(PERCENTAGE OF GARbled TEXT OF MESSAGES THAT ARE TO BE GARbled.)

(Probability that a message will be GARbled.)
RM PLAYER_CNT (NUMBER OF REMOVED PLAYERS)
NAME OF REMOVED PLAYER (IF ANY)
NAME OF REMOVED PLAYER (IF ANY)
NAME OF REMOVED PLAYER (IF ANY)

RM PLAYER_CNT NUMBER OF NAMES LISTED
NAME OF REMOVED PLAYER (IF ANY)
END OF FILE

IT ALSO READS GAME DATA FROM A SECOND FILE [CLARK] INTERFACE.DAT

DATA IN THIS FILE IS IN THE FOLLOWING FORMAT.

PLAYCNT (NUMBER OF PLAYERS IN THE GAME)
PLAYER1 REAL NAME
PLAYER1 VAX DIRECTORY NAME
PLAYER2 REAL NAME
PLAYER2 VAX DIRECTORY NAME

THERE WILL BE 'PLAYCNT' NAMES LISTED
LINK1TO2
LINK1TO3 (IF THREE PLAYERS AND SO ON UNTIL PLAYCNT)
LINK2TO1
LINK2TO3 (IF THREE PLAYERS AND SO ON UNTIL PLAYCNT)

PLAYER DO NOT HAVE LINKS TO THEMSELVES

END OF FILE

IN ADDITION, THIS SECTION EXTRACTS THE LOGIN NAME OF THE PLAYER WHICH IS USED LATER TO DETERMINE THE PLAYER'S ELIGIBILITY TO PLAY THE GAME.

THE MODULE REQUESTS THE PLAYER TO ENTER <CR> TO CONTINUE. HOWEVER, IN THE BLIND, IT IS REQUESTING A PRIVILEGE CODE THAT GIVES THE USER GAME PARAMETER MODIFICATION/CHANGE PRIVILEGES

IF THE PLAYER IS NOT PRIVILEGED, THEN THE PROGRAM BRANCHES TO PROCEDURE [CLARK.CCC]RTCCC.COM (SEE BELOW DESCRIPTION OF [CLARK.CCC]RTCCC)

IF THE PLAYER IS PRIVILEGED, THEN HE HAS THE OPTION OF SELECTING THE BUILD, MODIFY, STATUS OR PLAY PORTION OF THE PROGRAM.
APPENDIX F
C3 MODEL PROGRAM CODE

*** MAIN.COM ***

1 SET NOVERIFY
1 ERL/SYN/LOCAL/ALL 1 CLEAR LOCAL SYMBOL TABLE
1

******** INITIALIZE SECTION ********
1
1 THIS SECTION READS IN THE GAME DATA (IF ANY) AND STORES
1 THEM IN THE LOCAL TABLE.
1
1 INITIAL1:
1 OPEN/HEAD/ERROR=END_OF_DATA INFILE DATA:[CLARK]PARAMETER.DAT
1
1 CNT = 1
1 INLOOP1:
1 REAL/END_OF_FILE=END_OF_DATA INFILE ARRIVAL_RATE'CNT'
1 CNT = CNT + 1
1 IF CNT .LE. 5 THEN GO TO INLOOP1
1
1 CNT = 1
1 INLOOP2:
1 REAL/END_OF_FILE=END_OF_DATA INFILE SERVICE_RATE'CNT'
1 CNT = CNT + 1
1 IF CNT .LE. 5 THEN GO TO INLOOP2
1
1 CNT = 1
1 INLOOP3:
1 REAL/END_OF_FILE=END_OF_DATA INFILE LOSS_RATE'CNT'
1 CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO INLOOP3

CNT = 1

INLOOP4:
READ/END_OF_FILE=END_OF_DATA INFILE GARBLE_RATE'CNT'
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO INLOOP4

CNT = 1
INLOOP4A:
READ/END_OF_FILE=END_OF_DATA INFILE PGARB'CNT'
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO INLOOP4A

READ/END_OF_FILE=END_OF_DATA INFILE RM PLAYER_CNT
IF RM PLAYER_CNT .EQ. 0 THEN GOTO JUMPS
CNT = 1
INLOOP5:
READ/END_OF_FILE=END_OF_DATA INFILE RM PLAYER_KAMP'CNT'
CNT = CNT + 1
IF CNT .LE. RM PLAYER_CNT THEN GOTO INLOOP5

JUMPS:

END_OF_DATA:
CLOSE INFILE

OPEN/READ/ERROR=END_OF_DATA INFILE3 DATA: [CLARK] INTERFACE.DAT
READ/END_OF_FILE=END_OF_DATA INFILE3 PLAYCNT
CNT = 1
INLOOP6:
READ/END_OF_FILE=END_OF_DATA INFIL3S NAME
PLAYER 'CNT' := "NAME"
READ/END_OF_FILE=END_OF_DATA INFIL3S RNAMF
DIRECTORY NAME 'CNT' := "NAME"
CNT = CNT + 1
IF CNT .LE. PLAYCNT THEN GOTO INLOOP6

! CNT = 1
! KNT = 1
! INLOOP6:
! IF CNT .EQ. ANT THEN GOTO INSKIP3

! READ/END_OF_FILE=END_OF_DATA INFIL3S LINK
! LINK 'CNT' TO 'ANT' := "LINK"
!
! INSKIP3:
! KNT = KNT + 1
! IF KNT .LE. PLAYCNT THEN GOTO INLOOP7
! KNT = 1
! CNT = CNT + 1
! IF CNT .LE. PLAYCNT THEN GOTO INLOOP7
!
! END_OF_DATA:
!
! CLOSE INFIL3S
!
! THIS SECTION THE PLAYERS DIRECTORY NAME FOR LATER
! USE IN THE MAILER PROGRAM. IT DEFINES A GLOBAL
! SYMBOL (REAL_FROM) WHICH IDENTIFIES THIS DIRECTORY
! NAME TO THE MAILER.
!
TTT := 'F$DIRECTORY()' CNT = 'F$LOCATE(''.'',TTT)'
IF CNT .EQ. 'F$LENGTH(TTT) THEN GOTO CHECK_FOR_A_FRACKIT
CNT = CNT - 1
/*
  REAL_FROM := 'F$EXTRACT(1,CNT,TTT)'
  GOTO FOUND_DIRECTORY_NAME

  CHECK_FOR_A_BRACKET:
  CNT = 'F$LOCATE(",",TTT)'
  CNT = CNT - 1
  REAL_FROM := 'F$EXTRACT(1,CNT,TTT)'

  FOUND_DIRECTORY_NAME:

  INITIALIZATION_COMPLETE:
  LINK_CODE_CHANGED := "NO"  ! INITIALIZING LINK CHANGE FLAG.

  *************** END SECTION INITIALIZE ***************
*/

_TYPE SYS$INPUT

WELCOME TO THE NAVAL POSTGRADUATE SCHOOL C3 WARGAME

THIS GAME IS THE RESULT OF A THESIS FOR THE C-3 CURRICULUM
BY CAPTAIN LLOYD CLARK, CAPTAIN LARRY FLEES AND MAJOR BOB RAPP.

IT REPRESENTS AN IMPROVED ADAPTATION OF A STRUCTURED COMMUNICATIONS
MODEL FIRST DEVELOPED BY LT COL TOM STAGA AND LCDR TOM SICORSAY.

THE WARGAME ITSELF IS THE ACCLIMATIC THEATER MODEL (ATM) DEVELOPED
AT THE US ARMY WAR COLLEGE.

*******************************************************************************
SET TERMINAL /NOECHO
INQUIRE PWD "MTT: PLAYERS ENTER GAME PASSWORD. (ENTER <CR> IF UNKNOWN)"
SET TERMINAL /ECHO
IF PWD .NES. "CCC" THEN PWD := "NO"

SET TERMINAL /NOECHO
INQUIRE PRIV_CODE "CONTROLLER ENTER PASSWORD. (ENTER <CR> IF UNKNOWN)"
SET TERMINAL /ECHO

IF PRIV_CODE .EQS. "JWT" THEN GOTO PRIVILEGES
PRIVATE_CODE := "NO PRIVILEGES"

IF THE USER IS NOT A PRIVILEGED USER THEN ACTIVATE THE PLAYER
PORTION OF THE MODEL ONLY.

GAME:

IF PWD .EQS. "CCC" THEN @|CLARK.CCC|RTSSS.COM ! PLAYER PORTION OF THE GAME
IF PWD .EQS. "NO" THEN @|CLARK.CCC|RTCC.COM ! PLAYER PORTION OF THE GAME

IF PRIV_CODE .EQS. "JWT" THEN GOTO PRIVILEGES

HOUSEKEEPING... CLEAN UP DIRECTORY OF MISC FILES

CLEAN_HOUSE:

SET MESSAGE /NOTEXT/HOACILITY/NOIDENTIFICATION/NOSEVERITY ! STOP DISPLAY
DELETE GAME*.TIT.*
DELETE MAILING*.DAT.*
DELETE *SED*.DAT.*
DELETE MAILING*.TMP.*
DELETE STAT*.TXT.*
DELETE MSGOUT.TXT.*
WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS?

BUILD (B)
MODIFY (M)
PLAY (P)
STATUS (S)
INITIATE MTM (I)
QUIT (Q)

! LOOP1:
! INQUIRE ANSWER "ENTER B,M,P,S,I OR Q"
! IF ANSWER .EQS. "B" .OR. ANSWER .EQS. "BUILD" THEN GOTO ENTRY1
! IF ANSWER .EQS. "M" .OR. ANSWER .EQS. "MODIFY" THEN GOTO ENTRY2
! IF ANSWER .EQS. "P" .OR. ANSWER .EQS. "PLAY" THEN GOTO GAME
! IF ANSWER .EQS. "S" .OR. ANSWER .EQS. "STATUS" THEN GOTO GAME_STATUS
! IF ANSWER .EQS. "I" .OR. ANSWER .EQS. "INITIATE MTM" THEN GOTO MTM
! IF ANSWER .EQS. "Q" .OR. ANSWER .EQS. "QUIT" THEN GOTO CLEAN_HOUSE
! GOTO LOOP1
THIS SECTION CALLS A PROCEDURES THAT ALLOWS THE USER TO INITIATE
THE MCCLINIC THEATER MODEL WARGAME AND/OR CREATE A NEW DATA BASE
FOR THE WARGAME.

@|CLARK.CCC|MTM.COM
GOTO MENU
GAME_STATUS:
TYPE SY$INPUT

DISPLAY GAME STATUS

OPEN/READ STATUS_FILE [CLARK]INTERFACE.DAT
OPEN/WHITE OUTFILE STATUS.DAT
OPEN/WHITE OUTFILE STAT_CNT

WRITE OUTFILE "GAME STATUS AS OF $TIME()"
WRITE OUTFILE
READ STATUS_FILE STAT_CNT
CNT = 1

STATUS_LOOP:
READ STATUS_FILE NAME1
READ STATUS_FILE NAME2
WRITE OUTFILE "PLAYER 'CNT' : 'NAME1' IN DIRECTORY 'NAME2'"
CNT = CNT + 1
IF CNT .LE. STAT_CNT THEN GOTO STATUS_LOOP
WRITE OUTFILE ""
WRITE OUTFILE ""

CNT = 1
KNT = 1
IN THE POSITIONAL ORDER SHOWN BELOW (I.E., 100010
REPRESENTS AN ENCRYPTED LANDLINE AND A VOICE CIRCUIT).

1 -- ENCRYPTED LANDLINE (AUTODIN)
2 -- NON-ENCRYPTED LANDLINE (TELETYPE)
3 -- DIGITAL RF CIRCUIT (HF/VHF/UHF) WITH A/J
4 -- DIGITAL RF CIRCUIT WITHOUT A/J PROTECTION
5 -- VOICE (TELEPHONE)
6 -- PERFECT LINK (NO DELAY, NO DEGRADATION)

NOTE: ALL COMM LINKS HAVE BEEN PRESET TO 0. ENTER THE
DESIRED VALUE AS THE CHOICE IS PRESENTED TO YOU
PLUS <CR> (IF NO COMM LINK EXISTS, TYPE <CR> ONLY).

THIS SECTION ACQUIRES THE CIRCUIT VALUES
AND STORES THE VALUE IN LINK<NUM> TO<NUM>

CNT = 1
KNT = 1
OPEN/WRTE OUTFILE DRAV:<CLARA>DATA1TMP

LOOPS:
1F CNT .EQ. KNT THN COTO SKIP2
1F ANSWER2 .EQS. "" THN ANSWER2 := 00000
LINK CNT TO KNT := ANSWER2
WRITE OUTFILE 'LINK' CNT TO KNT = 'ANSWER2'
WRITE OUTFILE "'ANSWER2"

SKIP2:
COUNT = COUNT + 1
IF COUNT .LE. PLAYCNT THEN GOTO LOOP2
COUNT = COUNT - 1
!
WRITE OUTFILE " "
WRITE OUTFILE " "
!
CLOSE OUTFILE
CLOSE OUTFILE
!
ASSIGN/USER MODE STATISTIC.SRT SYS$OUTPUT:
SORT/KEY=(POSITION:1,SIZE=20) DRAW:[CITAKE]DATA.TMP DRAW:[CITAKE]DOC.SRT
!
OPEN/WRITE OUTFILE DOCUMENT.DAT
WRITE OUTFILE " PLAYER NAMES AS OF '"F$TIME()'"
WRITE OUTFILE " "
!
CLOSE OUTFILE
!
SKIP1:
!
WRITE SYS$OUTPUT " YOU HAVE NAMED "COUNT" PLAYERS"
!
TYPE SYS$INPUT

IF YOU WISH EACH PLAYER TO HAVE A COMPLETE COMMUNICATION
SUITE (I.E., TO BE ABLE TO COMMUNICATE DIRECTLY WITH EVERY
OTHER PLAYER VIA ANY LINK), TYPE "ALL". OTHERWISE, TYPE <CR>.

INQUIRE CHOICE
IF CHOICE .EQS. "ALL" THEN GOTO SETVAL
TYPE SYS$INPUT

ENTER A SIX DIGIT NUMBER CONSISTING OF 0'S AND 1'S
TO INDICATE CIRCUIT AVAILABILITY BETWEEN PLAYER PAIRS
PLAYCNT := 'LPLAYCNT'
PLAYCNT := 'LPLAYCNT'
WRITE SYS$OUTPUT "THE NUMBER OF PLAYERS IS: "'PLAYCNT'"
WRITE SYS$INPUT

WHAT ARE THE (GAME) NAMES OF THE PLAYERS IN THE GAME?

OPEN/WHITE OUTFILE DRAO:[CLARK]DATA.TMP ! CREATE AND OUTPUT FILE
OPEN/WHITE OUTFILE DRAO:[CLARK]DATA2.TMP ! CREATE AND OUTPUT FILE
OPEN/WHITE OUTFILE DRAO:[CLARK]INTERFACE.DAT ! CREATE AND OUTPUT FILE
WRITE OUTFILE "" ""
WRITE OUTFILE "" ""
WRITE OUTFILE ""'PLAYCNT'

COUNT = 1

LOCP2:
  INQUIRE NAME "PLAYER'COUNT'"
  PLAYER'COUNT' := 'NAME'
  INQUIRE NAME "WHAT IS PLAYER'COUNT' VAX DIRECTORY NAME?"
  DIRECTORY_NAME'COUNT' := 'RNAME'

THE FOLLOWING PROCEDURE STORES THE PLAYER/LINK
ON FILE DRAO:[CLARK]DATA.TMP

WRITE OUTFILE ""'NAME'"
WRITE OUTFILE ""'PLAYER'COUNT': 'NAME' IN DIRECTORY 'RNAME'"
WRITE OUTFILE ""'NAME'"
WRITE OUTFILE ""'RNAME'"
IF NAME .EQS. "" THEN GOTO LOCP2
E = -4

1. 

S = E - 0.4

P = -4

1

\( \text{ENTRY1:} \)

\( \text{TYPE: SYSINPUT} \)

\( \text{THIS PORTION OF THE PROGRAM WILL ALLOW THE} \)

\( \text{CONTROLLER TO:} \)

- 1. DESIGN THE ORGANIZATIONAL STRUCTURE FOR THE GAME

- 2. DESIGN THE COMM NET SUPPORTING THE ORGANIZATION

\( \text{INQUIRY: LPLAYCNT "HOW MANY PLAYERS, INCLUDING THE CONTROLLER, ARE THERE?"} \)

\( \text{IF LPLAYCNT .LE. 2 OR LPLAYCNT .GE. 99999 THEN GOTO MENU} \)
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO STATUS_LOOP3

WRITE OUTFILE " "
WRITE OUTFILE " "

CNT = 1
STATUS_LOOP4:

READ STATUS_FILE RATE
WRITE OUTFILE " LOSS RATE FOR CIRCUIT "'CNR' " "RATEN"'
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO STATUS_LOOP4

WRITE OUTFILE " "
WRITE OUTFILE " "

CNT = 1
STATUS_LOOP5:

READ STATUS_FILE RATE
WRITE OUTFILE " GARNET RATE FOR CIRCUIT "'CNR' " "RATEN"'
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO STATUS_LOOP5

WRITE OUTFILE " "
WRITE OUTFILE " "

CNT = 1
STATUS_LOOP6:

READ STATUS_FILE RATE
```plaintext
STATUS_ICOP1:
  IF CNT .EQ. KNT THEN GOTO STAT_SKIP

  REAL STATUS_FILE LINK
  WRITE OUTFILE " LINK ' ' CNT ' ' KNT ----> ' ' LINK '

STAT_SKIP:
  KNT = KNT + 1
  IF KNT .LE. STAT_CNT THEN GOTO STATUS_LOOP1
  KNT = 1
  CNT = CNT + 1
  IF CNT .LE. STAT_CNT THEN GOTO STATUS_ICOP1

CLOSE STATUS_FILE
OPEN/READ STATUS_FILE [CLARA] PARAMETER.DAT

  WRITE OUTFILE " 
  WRITE OUTFILE 

  CNT = 1
STATUS_ICOP2:

  REAL STATUS_FILE RATE
  WRITE OUTFILE " ARRIVAL RATE FOR CIRCUIT ' ' CNT ' ' ----> ' ' RATE ' 
  CNT = CNT + 1
  IF CNT .LE. 5 THEN GOTO STATUS_LOOP2

  WRITE OUTFILE " 
  WRITE OUTFILE 

  CNT = 1
STATUS_ICOP2:

  REAL STATUS_FILE RATE
  WRITE OUTFILE " SERVICE RATE FOR CIRCUIT ' ' CNT ' ' ----> ' ' RATE ' 
```
GOTO ENTRYZ

SET_SVC_RATE_3:

SERVICE_RATE1 := 9.30
SERVICE_RATE2 := 9.15
SERVICE_RATE3 := 9.12
SERVICE_RATE4 := 9.09
SERVICE_RATE5 := 9.06

GOTO ENTRYZ

SET_PARAMETER_2:

TYPE: SYS$INPUT

ENTER THE "PROBABILITY" THAT A MESSAGES WILL
BE "GARbled" DURING TRANSMISSION. FIVE EQUALS
FIVE PERCENT. USE INTEGER VALUES.

TYPE "1" IF YOU WISH TO ENTER A SEPARATE PROBABILITY
FOR EACH CIRCUIT TYPE. (1-5)

TYPE "2" IF YOU WANT A STANDARD PROBABILITY FOR ALL
CIRCUITS.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES
CURRENTLY SET -- TYPE <CR>

ENTER THE MESSAGE GARBLE PROBABILITY FOR EACH CIRCUIT
FOR CIRCUIT TYPE [ ]:
! INCLUDE NORM_MED_HEAVY "NORMAL, MEDIUM OR HEAVY"
IF NORM_MED_HEAVY .EQS. "NORMAL" THEN A := 1.0
IF NORM_MED.HEAVY .EQS. "MEDIUM" THEN A := 2.0
IF NORM_MED.HEAVY .EQS. "HEAVY" THEN A := 3.0

ARRIVAL_RATE1 := 'A'
ARRIVAL_RATE2 := 'A'
ARRIVAL_RATE3 := 'A'
ARRIVAL_RATE4 := 'A'
ARRIVAL_RATE5 := 'A'

IF A .EQS. "1.0" THEN GOTO SET_SVC_RATE_1
IF A .EQS. "2.0" THEN GOTO SET_SVC_RATE_2
IF A .EQS. "3.0" THEN GOTO SET_SVC_RATE_3

GOTO ENTRY;

SET_SVC_RATE_1:
SERVICE_RATE1 := 3.10
SERVICE_RATE2 := 3.05
SERVICE_RATE3 := 3.04
SERVICE_RATE4 := 3.03
SERVICE_RATE5 := 3.02

GOTO ENTRY;

SET_SVC_RATE_2:
SERVICE_RATE1 := 6.40
SERVICE_RATE2 := 6.10
SERVICE_RATE3 := 6.08
SERVICE_RATE4 := 6.06
SERVICE_RATE5 := 6.02

!
CNT = 1

L00P4:
WHILE SYS$OUTPUT "FOR CIRCUIT TYPE [ 'CAT' ]"
INQUIRE/NOPUNCTUATION ARRIVAL_RATE,CNT, "ARRIVAL RATE = "
INQUIRE/NOPUNCTUATION SERVICE_RATE,CNT, "SERVICE RATE = "
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO L00P4

INQUIRE NULL "RETURN TO MODIFY MENU BY DEPRESSING <CR>"
GOTO ENTRY2

ENTRY4:
TYPE SYS$INPUT

THE PRE-ESTABLISHED RELATIONSHIP BETWEEN ARRIVAL
AND SERVICE RATES FOR A NORMAL ARRIVAL RATE IS
AS FOLLOWS:

FOR CIRCUIT TYPE 1: S = 3.10A
FOR CIRCUIT TYPE 2: S = 3.25A
FOR CIRCUIT TYPE 3: S = 3.04A
FOR CIRCUIT TYPE 4: S = 3.03A
FOR CIRCUIT TYPE 5: S = 3.02A

ENTER "NORMAL", "MEDIUM" OR "HEAVY" TO ESTABLISH
THE INITIAL MESSAGE ARRIVAL RATES.

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING
THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY".
VARY ONLY THE MESSAGE ARRIVAL RATE BY REQUESTING:

A. NORMAL TRAFFIC
B. MEDIUM TRAFFIC (TWICE THE NORMAL ARRIVAL RATE)
C. HEAVY TRAFFIC (THREE TIMES THE NORMAL RATE)

TYPE "1" TO INSERT SPECIFIC ARRIVAL AND SERVICE RATES.
TYPE "2" TO USE THE GENERAL RATES (NORMAL, MEDIUM, HEAVY)

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>.

INQUIRE ANSWER " ENTER 1, 2, OR <CR> "

IF ANSWER .EQS. "1" THEN GOTO ENTRY3
IF ANSWER .EQS. "2" THEN GOTO ENTRY4
IF ANSWER .EQS. " " THEN GOTO ENTRY2

GOTO SET_PARAMETER_1

ENTRY3:

TYPE SYS$INPUT

TO AVOID A QUEUE WHICH GROWS WITHOUT BOUND, INSURE
A/S < 1

NOTE: MESSAGE RATES ARE IN NUMBERS OF MESSAGES PER MINUTE. USE REAL NUMBERS 99.99 OR LESS.

SPECIFIC RATES:
1 -- ENCRYPTED LANDLINE (AUTODIN)
2 -- NON-ENCRYPTED LANDLINE (TELETYPE)
3 -- DIGITAL RF CIRCUIT (HF/VHF/UHF) WITH A/J
4 -- DIGITAL RF CIRCUIT WITHOUT A/J PROTECTION
5 -- VOICE (TELEPHONE)
6 -- PERFECT LINK (NO DELAY, NO DEGRADATION)

WHITE SYS$OUTPUT "12345" THE CURRENT LINK CODE IS :
WHITE SYS$OUTPUT LINK"INDEX1"TO"INDEX2"
WHITE SYS$OUTPUT

INQUIRE LINK"INDEX1"TO"INDEX2" "ENTER NEW LINK CODE"

LINK_CODE_CHANGED := "YES" ! LINK CODE CHANGE FLAG

GOTO ENTRY2

SET_PARAMETER_1:

TYPE SYS$INPUT

BASED ON THE QUEUING ALGORITHM FOR SINGLE SERVER
FACILITIES THE AVERAGE AMOUNT OF TRANSMISSION DELAY
FOR MESSAGES ADDRESSED TO A GIVEN FACILITY (WQ)
CAN BE EXPRESSED AS A FUNCTION OF THE AVERAGE
MESSAGE ARRIVAL RATE (A) AND THE AVERAGE MESSAGE
SERVICE RATE (S).

\[ WQ = \frac{A}{S(S-A)} \]

YOU MAY SPECIFY THE ACTUAL ARRIVAL RATES AND
SERVICE RATES FOR EACH CLASS OF COMMUNICATIONS
(1 THROUGH 5) OR YOU MAY RELY UPON A PRE-ESTABLISHED
SERVICE RATE AND ARRIVAL RATE RELATIONSHIP AND
$ !  CNT = 1
$ !  INDEX1 = 0
$ !  INDEX2 = 0
$ !  CHG_LINK_LOOP:
$ !
$ !    IF CHG_NAME1 .EQ. 'PLAY' CNT THEN INDEX1 = 'CNT'
$ !    IF CHG_NAME2 .EQ. 'PLAY' CNT THEN INDEX2 = 'CNT'
$ !    IF INDEX1 .NE. 0 .AND. INDEX2 .NE. 0 THEN GOTO OVER_CHG
$ !    CNT = CNT + 1
$ !    IF CNT .GT. 10 THEN GOTO INVALID_PLAYER_NAME
$ !    GOTO CHG_LINK_LOOP
$ !  INVALID_PLAYER_NAME:
$ !  TYPE SYS$INPUT

******** INVALID PLAYER NAME USED ********

$ !    IF INDEX1 .EQ. 0 THEN WRITE SYS$OUTPUT " "'CHG_NAME1'"
$ !    IF INDEX2 .EQ. 0 THEN WRITE SYS$OUTPUT " "'CHG_NAME2'"
$ !    GOTO ENTRY2
$ !  OVER_CHG:
$ !  TYPE SYS$INPUT

ENTER A SIX DIGIT NUMBER CONSISTING OF 0'S AND 1'S
TO ILLICIT CIRCUIT AVAILABILITY BETWEEN PLAYER PAIRS
IN THE POSITIONAL ORDER SHOWN BELOW (I.E., 100010
REPRESENTS AN ENCRYPTED LANDLINE AND A VOICE CIRCUIT).
2. REMOVE PLAYERS FROM THE GAME.

6. RETURN TO MAIN MENU. (GAME UNCHANGED, STATUS DISPLAYED)

# 7. IMPLEMENT CHANGES AND RETURN TO MAIN MENU.
(STATUS DISPLAYED)

#NOTE#: CHANGES DO NOT BECOME EFFECTIVE UNTIL ITEM #7 IS REQUESTED. THIS ALLOWS YOU TO RECOVER BY SIMPLY RETURNING TO THE MAIN MENU (ITEM #6)

INQUIRE PARAM "ENTER 0, 1, 2, 3, 4, 5, 6 OR ?"
IF PARAM .EQS. "0" THEN GOTO SET_PARAMETER_0
IF PARAM .EQS. "1" THEN GOTO SET_PARAMETER_1
IF PARAM .EQS. "2" THEN GOTO SET_PARAMETER_2
IF PARAM .EQS. "3" THEN GOTO SET_PARAMETER_3
IF PARAM .EQS. "4" THEN GOTO SET_PARAMETER_4
IF PARAM .EQS. "5" THEN GOTO SET_PARAMETER_5
IF PARAM .EQS. "6" THEN GOTO GAME_STATUS
IF PARAM .EQS. "?" THEN GOTO SAVE_MODIFICATIONS

GOTO ENTRY2
GOTO ENTRY2
SET_PARAMETER_0:

! TYPE SYS$INPUT

YOU WISH TO CHANGE THE COMMUNICATION LINK CODE BETWEEN TWO GAME PLAYERS.

INQUIRE CHG_NAME1 "CHANGE LINK CODE FROM PLAYER (NAME)"
INQUIRE CHG_NAME2 "TO PLAYER"

! DETERMINE THE LINK CODE
KNT = KNT + 1
IF KNT .LE. PLAYCNT THEN GOTO LOOP3
KNT = 1
CNT = CNT + 1
IF CNT .LE. PLAYCNT THEN GOTO LOOP3
CLOSE OUTFIL
CLOSE OUTFIL
APPEND DRA: [CLARK]DOC.SRT, DRA: [CLARK]DATA2.TMP, DRA:
[CLARK]DATA1.TMP DOCUMENT
INQUIRE NULL "RETURN TO MAIN MENU BY DEPRESSING <CR>"
GOTO MENU
ENTRY2:
TYPE SYS$INPUT

PARAMETER INITIALIZATION AND MODIFICATION SUBROUTINE

THIS PORTION OF THE PROGRAM ALLOWS THE CONTROLLER TO:

1. CHANGE COMMUNICATION LINK BETWEEN TWO PLAYERS.
2. SET MESSAGE ARRIVAL AND SERVICE RATES TO BE USED FOR EACH TYPE LINK.
3. SET PROBABILITY OF MESSAGE GARBLING.
4. SET MESSAGE LOSS RATES FOR EACH CIRCUIT TYPE.
5. SET RATE OF MESSAGE GARBLING.
PROBABILITY =

OK:

ENTER THE STANDARD PROBABILITY FOR ALL CIRCUITS

PROBABILITY =

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING
THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY"

1
1
1

ENQUIRE ANSWER " ENTER 1, 2, OR <CR> "
1
1
1

ENTRY21:
1
1
1

IF ANSWER .EQS. "1" THEN GOTO ENTRY21
1
1
1

IF ANSWER .EQS. "2" THEN GOTO ENTRY22
1
1
1

IF ANSWER .EQS. "3" THEN GOTO ENTRY2
1
1
1

ENTRY21:
1
1
1

CNT = 1
1
1
1

TYPE SYS$INPUT
1
1
1

ENTER THE MESSAGE GARBLE PROBABILITY FOR EACH CIRCUIT
1
1
1

PACKT_LOOP:
1
1
1

WRITE SYS$OUTPUT " FOR CIRCUIT TYPE [ "CNT" ]:
1
1
1

INQUIRE/NO PUNCTUATION $GARB'CNT' "PROBABILITY = 
1
1
1

CNT = CNT + 1
1
1
1

IF CNT .LE. 5 THEN GOTO PACKT_LOOP
1
1
1

GOTO ENTRY2
1
1
1

ENTRY22:
1
1
1

TYPE SYS$INPUT
1
1
1
ENTER THE STANDARD PROBABILITY FOR ALL CIRCUITS

INQUIRE/NO PUNCTUATION PGARB " PROBABILITY = "

! CAT = 1
PGARB_LOOP:
    PGARB'CNT' := 'PGARB'
    CAT = CAT + 1
    IF CNT .LE. 5 THEN GOTO PGARB_LOOP

GOTO ENTRY2

SET_PARAMETER_3:

TYPE SYS$INPUT

ENTER THE RATE FOR MESSAGES TO BE "LOST." FIVE EQUALS FIVE PERCENT. USE INTEGER VALUES.

ENTER "1" IF YOU WISH TO ENTER A SEPARATE LOSS RATE FOR EACH CIRCUIT TYPE. (1-5)

ENTER "2" IF YOU WANT A STANDARD RATE FOR ALL CIRCUITS.

NOTE: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>

ENTER THE LOSS RATE BY COMM CIRCUIT TYPE FOR CIRCUIT TYPE [ ]:

LOSS RATE =
OR:

ENTER THE STANDARD RATE FOR ALL CIRCUITS

LOSS RATE =

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING
THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY"

\[ \text{INQUIRE ANSWER " ENTER 1, 2, OR <CH> "} \]

\[ \text{IF ANSWER .EQS. "1" THEN GOTO ENTRY6} \]
\[ \text{IF ANSWER .EQS. "2" THEN GOTO ENTRY7} \]

\[ \text{GOTO ENTRY2} \]

ENTRY6:
\[ \text{CNT} = 1 \]

LOOP6:
\[ \text{WRITE SYS$OUTPUT " ENTER THE LOSS RATES FOR COMM CIRCUIT TYPE [ 'CNT' ]"} \]
\[ \text{INQUIRE/NOPUNCTUATION LOSS_RATE 'CNT' "LOSS RATE = "} \]
\[ \text{CNT} = \text{CNT} + 1 \]
\[ \text{IF CNT .LE. 5 THEN GOTO LOOP6} \]

GOTO ENTRY7

ENTRY7:
\[ \text{TYPE SYS$INPUT} \]

ENTER THE STANDARD LOSSRATE

\[ \text{INQUIRE/NOPUNCTUATION LOSSRATE " LOSS RATE = "} \]

!
CLT = 1
LOSS_LOOP:
  LOSS_RATE(CNT) := 'LOSSRATE'
  CNT = CNT + 1
  IF CNT .LE. 5 THEN GOTO LOSS_LOOP
  GOTO ENTRY2
ENTRY2:
ENTRY3:
  INQUIRE NULL "ENTER <CR> TO CONTINUE OR 1 FOR MAIN MENU"
  IF NULL .EQS. "1" THEN GOTO MENU
ENTRY4:
  SET PARAMETER_4:
  TYPE SYS INPUT
  ENTER THE RATE AT WHICH YOU WISH MESSAGES TO BE "GARBLED" DURING TRANSMISSION. FIVE EQUALS FIVE PERCENT. USE INTEGER VALUES.

  TYPE "1" IF YOU WISH TO ENTER A SEPARATE RATE FOR EACH CIRCUIT TYPE. (1-5)

  TYPE "2" IF YOU WANT A STANDARD RATE FOR ALL CIRCUITS.

NOTF: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>.

  ENTER THE MESSAGE GARBLE RATE FOR EACH CIRCUIT FOR CIRCUIT TYPE 1:

NOTF: IF YOU DO NOT WISH TO CHANGE VALUES CURRENTLY SET -- TYPE <CR>.

  ENTER THE MESSAGE GARBLE RATE FOR EACH CIRCUIT FOR CIRCUIT TYPE 2:
RATE -

ON:

ENTER THE STANDARD RATE FOR ALL CIRCUITS

RATE =

CHANGES TO THESE VALUES WHICH ARE DESIRED DURING THE GAME SHOULD BE MADE BY RE-ENTERING "MODIFY"

! INQUIRE ANSWER " ENTER 1, 2, OR <CR> "

! IF ANSWER .EQS. "1" THEN GOTO ENTRY9
! IF ANSWER .EQS. "2" THEN GOTO ENTRY10
! IF ANSWER .EQS. "3" THEN GOTO ENTRY2

ENTRY9:

CNT = 1
TYPE SYS$INPUT
ENTER THE MESSAGE GALLERY RATE FOR EACH CIRCUIT

ENTRY10:

CNT = 1
TYPE SYS$INPUT
ENTRY10:

TYPE SYS$INPUT
ENTER THE STANDARD RATE FOR ALL CIRCUITS

INQUIRE/NOPIUNCUTATION GARB_RATE " RATE = "

CNT = 1

GARB_LOOP:
  GARB_RATE,CNT := 'GARB_RATE'
  CNT := CNT + 1
  IF CNT .LE. 5 THEN GOTO GARB_LOOP

GOTO ENTRY2

SIT_PARAMETER_5:

ENTRY11:

TYPE SYS$INPUT

ENTER THE NUMBER OF PLAYERS AND THE NAME OF EACH
PLAYER WHO HAS BEEN "DESTROYED" OR WHO FOR SOME
REASON IS TO BE REMOVED FROM THE GAME AFTER GAME
START. ENTER THE NUMBER AND A <CR> AND THE NAME
OF EACH PLAYER WITH A <CR> FOLLOWING EACH NAME.

NOTE: IF YOU DO NOT WISH TO CHANGE THE VALUES
CURRENTLY SET -- TYPE <CR>.

INQUIRE/NOPIUNCUTATION REPLAYCNT -
"NUMBER OF PLAYER(S) TO BE REMOVED = "

IF REPLAYCNT .EQ. 0 THEN GOTO ENTRY2
  REPLAYER_CNT := 'REPLAYER_CNT'
  IF REPLAYER_CNT .EQ. 0 THEN GOTO ENTRY2

CNT = 1
LOOP1:

INQUIRE NAME

"PLAYER NAME"

NR PLAYER NAME 'Cnt' := 'NAME'

WRITE SYS$OUTPUT 'REMOVED PLAYER IS: ' 'NAME'

Cnt = Cnt + 1

IF CNT .LT. NRPLAYERRCNT THEN GOTO LOOP1

GOTO ENTRY2

THIS IS THE END OF THE MODIFY PORTION OF WARGAME.

MENU:

TYPE SYS$INPUT

WHAT PORTION OF THE WARGAME DO YOU WISH TO ACCESS?

BUILD (B)
MODIFY (M)
PLAY (P)
INITIATE FTM (I)
STATUS (S)
QUIT (Q)

GOTO LOOP1

SETVAL:

THIS SECTION ENABLES ALL CURCUIT LINK VALUES AND STORES THE VALUE IN LINK<NUM>TC<NUM>

CNT = 1
\$ 1
\$ LOOP4:
\$ TYPE SYS4INPUT

ENTER SELECTION:
1. SEND ANOTHER MESSAGE
2. RETURN TO COMMAND LEVEL
3. QUIT THE GAME

\$ INQUIRE SELECTION "ENTER SELECTION 1, 2, OR 3"
\$ IF SELECTION .EQ. 1 THEN GOTO LOOP1
\$ IF SELECTION .EQ. 2 THEN GOTO LOOP2
\$ IF SELECTION .EQ. 3 THEN GOTO LOOP3
\$ GOTO LOOP4
\$ LOOP2:
\$ STOPGAME := YES
\$ EXIT

*** RTCCG.COM ***
\$ SET NOCONTROL_Y
\$ SET NOVERIFY
\$ !
\$ COMM := "DONE"
\$ !
\$ OPT := 4MAIL7SFINDMSG7READMSG4QUIT4HELP
\$ TYPE SYS4INPUT

WELCOME TO THE TO THE NAVAL POSTGRADUATE SCHOOL C3 WARGAME
HERE IS A LIST OF ALLOWABLE COMMANDS AND THEIR DEFINITIONS.

\$
**THE MESSAGE ARE NOW SUBMITTED TO THE BATCH SYSTEM TO BE**
**APPROPRIATELY DELAYED AND FORWARDED.**

OPEN/READ INFILE MSGOUT.TXT
CNT = 1

SUBMIT_MORE:
GLOBAL_CNT = 'GLOBAL_CNT' + 1
IF GLOBAL_CNT .GE. 999 THEN GLOBAL_CNT = 1

READ/FNL OR FILE=FINISH INFILE TO PLAYER
READ/FNL OR FILE=FINISH INFILE MSG_FILE
READ/FNL OR FILE=FINISH INFILE DELAY_TIME

SUBMIT/NOIDENTITY DRAo:(Clara.CCC)SENDMSG.COM -
PARAMETERS=""SUBJ",""MSG_FILE",""TO_PLAYER",""FROM",""DELAY_TIME",""GLOBAL_CNT",""TO",""CIRCUIT_TYPE"")

CNT = CNT + 1 ! INCREMENT FILE VERSION NUMBER
GOTO SUBMIT_MORE

FINISH:
CLOSE INFILE
4 -- DIGITAL RF CIRCUIT WITHOUT A/J PROTECTION
2 -- VOICE (TELEPHONE)

! GOTO FIND_CIRCUIT_TYPE
! SKIP_TYPE:
! TEST FOR NUMERIC CIRCUIT TYPE.
! IF CIRCUIT_TYPE  LT. 0 .OR. CIRCUIT_TYPE  GT. 7 THEN GOTO FIND_CIRCUIT_TYPE

This procedure prompts for mailing information, invokes ed to allow
you to edit your text.
ALSO, this procedure initiates a subprocess to scramble
the message, delay the message according to its message path and
send the message on its way using the mail utility.

EDIT MESSAGE TEXT BY INVOKING THE EDITOR.

ASSIGN/USER_MODE SYS$COMMAND SYS$INPUT
edit/edt CMAE: [CLARK.CCC]MSG.TXT /O=message.txt

EDMAE: [CLARK.CCC]CKMSG.COM

IF SEVERITY .EQ. 2 THEN GOTO LOOP4 INACTIVE PLAYER OR MISSPELLED NAME
IF SEVERITY .EQ. 3 THEN GOTO LOOP4 INC ADDRESSSEE. SEND NO MESSAGE
CIRCUIT TYPES ARE DESCRIBED BELOW:

0 -- FIRST AVAILABLE CIRCUIT IN THE ORDER LISTED BELOW
1 -- "PERFECT LINK (NO DELAY, NO DEGRADATION)
2 -- ENCRYPTED LANDLINE (AMIDIN)
3 -- NON-ENCRYPTED LANDLINE (TELETYPE)
4 -- DIGITAL No CIRCUIT (Hf/Vhf/Uhf) WITH A/J
\$ ASSIGN/USER MODE SYSSCOMMAND SYSSINPUT
\$ RUN DRA:O:[CLARK.INOUTP]INOUTP
\$ TEST FOR THE BRANCH VALUE. THE FLAGS ARE SET IN THE
\$ INPUT/OUTPUT PROGRAM (A FORTRAN PROGRAM) BY THE USE
\$ OF THE LIBDO SYSTEM CALL.
\$ IF COMM .EQS. "YES" THEN DRA:O:[CLARK.CCC]RTCCC
\$ IF COMM .EQS. "DONE" THEN GOTO LOOP2
\$ IF INIMAIL .EQS. "YES" THEN DRA:O:[CLARK.CCC]ACTMAIL
\$ IF INIMAIL .EQS. "DONE" THEN GOTO LOOP2
\$ IF MTM_HELP .EQS. "YES" THEN DRA:O:[CLARK.CCC]MTMHELP
\$ IF MTM_HELP .EQS. "DONE" THEN GOTO LOOP2
\$ GOTO LOOP1
\$ LOOP2:
\$ RESET FLAGS TO NO AFTER EXECUTION OF EITHER SUBPROGRAM.
\$ INFAIL := "NO"
\$ MTMHELP := "NO"
\$ COMM := "NO"
\$ SEE DESCRIPTION OF INOUTP ABOVE
\$ ASSIGN/USER MODE SYSSCOMMAND SYSSINPUT
\$ RUN DRA:O:[CLARK.INOUTP]INOUTP1
\$ TEST FOR BRANCH FLAG.
\$ IF COMM .EQS. "YES" THEN DRA:O:[CLARK.CCC]RTCCC
**ERROR:**

```plaintext
WRITE SYS$OUTPUT "*** THE VALUE IS NOT ACCEPTABLE ***"
GOTO LOOP3
```

**RTSSS.COM**

```plaintext
**SET NOVERIFY**

**CLEAR THE MESSAGE TO PARAMETER AND THE MESSAGE SUBJECT**
**PARAMETER TO ACCEPT NEW VALUES**
TO := ""
SUBJ := ""

**DELETE COLOR.DAT.**! FILE CONTAINS RED OR BLUE. RESET IT.

**ON FIRST ENTRY TO THIS DCL, I SET MY INITIATE MAIL, HELP AND**
**STOP GAME FLAGS TO NO. THESE FLAGS ARE USED TO BRANCH**
**TO THE APPROPRIATE SECTION OF THIS DCL**

COMM := "NO"
INFAIL := "NO"
RTM_HELP := "NC"
STOPGAME := "NO"

**INOUTP OR THE INPUT/OUTPUT PROGRAM IS THE COMMUNICATION**
**INTERFACE TO THE MCCLINIC THEATER NOFE. INOUTP IS**
**EXECUTED ONLY ONCE (THE FIRST TIME A PLAYER ENTERS THE**
**GAME). ALL SUBSEQUENT ENTRIES ARE EXECUTED IN INOUTP1**
**WHICH IS A MODIFIED VERSION OF THE ORIGINAL INPUT/OUTPUT**
**PROGRAM.**
```
! CLOSE OUTFILE
!
!* IF LINK_CODE_CHANGED .NES. "YES" THEN GOTO GAME_STATUS
!
OPEN/WHITE OUTFILES DRAW:[CLARA]INTERFACE.DAT
WRITE OUTFILES PLAYCNT
CNT = 1
SAV_LOOP6:
   WRITE OUTFILES PLAYER'CNT'
   WRITE OUTFILES DIRECTORY_NAME'CNT'
   CNT = CLT + 1
   IF CNT .LE. PLAYCNT THEN GOTO SAV_LOOP6
!
   CNT = 1
   KNT = 1
SAV_LOOP7:
   IF CNT .EQ. KNT THEN GOTO SAV_SKIP3
!
   WRITE OUTFILES LINK'CNT'TO'KNT'
!
SAV_SKIP3:
   KNT = KNT + 1
   IF KNT .LE. PLAYCNT THEN GOTO SAV_LOOP7
   KNT = 1
   CNT = CNT + 1
   IF CNT .LE. PLAYCNT THEN GOTO SAV_LOOP7
!
CLOSE OUTFILES
!* LINK_CODE_CHANGED := "NO" ! RESET FLAG
!* GOTO GAME_STATUS
!
\$!
\$ write outfile ""service_rate1"
\$ write outfile ""service_rate2"
\$ write outfile ""service_rate3"
\$ write outfile ""service_rate4"
\$ write outfile ""service_rate5"
\$ write outfile ""loss_rate1"
\$ write outfile ""loss_rate2"
\$ write outfile ""loss_rate3"
\$ write outfile ""loss_rate4"
\$ write outfile ""loss_rate5"
\$ write outfile ""garble_rate1"
\$ write outfile ""garble_rate2"
\$ write outfile ""garble_rate3"
\$ write outfile ""garble_rate4"
\$ write outfile ""garble_rate5"
\$ write outfile ""pgarbl"
\$ write outfile ""pgarbl2"
\$ write outfile ""pgarbl3"
\$ write outfile ""pgarbl4"
\$ write outfile ""pgarbl5"
\$ write outfile ""rm_player_cnt"

1f rm_player_cnt .le. 0 then goto over5
CNT = .
outloop5:
    write outfile rm_player_name\'cnt'
    CNT = CNT + 1
1f CNT .le. rm_player_cnt then goto outloop5
over5:
GOTO SUBMIT_MORE
! FINISH:
! CLOSE INFILE
! GOTO NEXT
! QUIT:
! EXIT

*** ACTMAIL.COM ***

ASSIGN/USER_MODE SYS$COMMAND SYS$INPUT
MAIL
INMAIL ::= "DONE"
EXIT

*** MTMHELP ***

SET NOCONTROLL
! SET NOVERIFY
! MTM_HELP ::= "DONE"
SUPPLY?SEALIFT?SEND?TIME?
! TYPE SYS$INPUT

WARGAME COMMAND INTERPRER
Also, this procedure initiates a subprocess to scramble
the message, delay the message according to its message path and
send the message on its way using the mail utility.

EDIT MESSAGE TEXT BY INVOKING THE EDITOR.

ASSIGN/USER_MODE SYS$COMMAND SYS$INPUT
edit/edit DRAW:[CLARK.CCC]MSG.TXT /u=message.txt

@DRAW:[CLARK.CCC]SENDMSG.COM

IF SEVERITY EQ. 2 THEN GOTO NEXT !INACTIVE PLAYER OR MISPELLED NAME
IF SEVERITY EQ. 3 THEN GOTO NEXT !NO ADDRESS. NO MESSAGE SENT.

OPEN/READ INFILE MSGOUT.TAT
CNT = 1
SUBMIT/TO:

GLOBAL_CNT == 'GLOBAL_CNT' + 1
IF GLOBAL_CNT .GE. 999 THEN GLOBAL_CNT == 1

READ/ END OF FILE FINISH INFILE TO_PLAYER
READ/END OF FILE FINISH INFILE MSG_FILE
READ/END OF FILE FINISH INFILE LDELAY TIME

SUBMIT/NOIDENTIFY DRAW:[CLARK.CCC]SENDMSG.COM -
/PARAMETERS=('"SUBJ"', '"MSG_FILE"', '"TO_PLAYER"', -
'"FROM"', '"DELAY TIME"', '"GLOBAL_CNT"', '"TO"', '"CIRCUIT_TYPE"')

CNT = CNT + 1 ! INCREMENT FILE VERSION NUMBER
! INQUIRE/GLOBAL TO
! INQUIRE/GLOBAL FROM
! INQUIRE/GLOBAL SUBJ
!
! FINL_CIRCUIT_TYPE:
!
! INQUIRE/GLOBAL CIRCUIT_TYPE "SELECT CIRCUIT TYPE 0,1,2,3,4,5,6 OR ?"
!
! IF CIRCUIT_TYPE .NE. "?" THEN GOTO SKIP_TYPE
! TYPE SYS$INPUT

CIRCUIT TYPES ARE DESCRIBED BELOW:

0 -- FIRST AVAILABLE CIRCUIT IN THE ORDER LISTED BELOW
6 -- PERFECT LINK (NO DELAY, NO DEGRADATION)
1 -- ENCRYPTED LANDLINE (AUTODIN)
2 -- NON-ENCRYPTED LANDLINE (TELETYPE)
3 -- DIGITAL HF CIRCUIT (HF/VHF/UHF) WITH A/J
4 -- DIGITAL HF CIRCUIT WITHOUT A/J PROTECTION
5 -- VOICE (TELEPHONE)

! GOTO FINL_CIRCUIT_TYPE
! SaIP_TYPE:
! TEST FOR NUMERIC CIRCUIT TYPE.
! IF CIRCUIT_TYPE .LT. 0 .OR. CIRCUIT_TYPE .GT. 7 THEN GOTO FIND_CIRCUIT_TYPE
! ++
! mail.com
! |
! This procedure prompts for mailing information, invoke edit to allow
! you to edit your text.


```plaintext
$ CNT = 1
$ KNT = 1
$ INLOOP7:
  $  IF CNT .EQ. KNT THEN GOTO INSKIP3
  $  !
  $  READ INFILE3 LINK
  $  LINK 'CNT' TO 'KNT' := ''LINK''
  $ !
  $  INSKIP3:
  $  KNT = KNT + 1
  $  IF KNT .LE. PLAYCNT THEN GOTO INLOOP7
  $  KNT = 1
  $  CNT = CNT + 1
  $  IF CNT .LE. PLAYCNT THEN GOTO INLOOP7
  $!
  $ JUMP_LOOP2:
  $  CLOSE INFILE3
  $!
  $!

$ !******************************************************************************************
$ ! CLEARING THE MESSAGE TO PARAMETER AND THE MESSAGE SUBJECT
$ ! PARAMETER TO ACCEPT NEW VALUES
$ ! TO := ''
$ ! SUBJ := ''
$ ! TYPE SYS$INPUT ! SKIP DOWN TWO LINES

$ !
$ ! LOOP1:
$ ! REQUEST MESSAGE PARAMETERS
```
CNT = 1

INLOOP4A:

REAL INFILE PGARR 'CNT'
CNT = CNT + 1
IF CNT .LE. 5 THEN GOTO INLOOP4A

READ INFILE RM_PLAYER_CNT

RM_PLAYER_CNT := 'RM PLAYER_CNT'

IF RM_PLAYER_CNT .EQ. 0 THEN GOTO JUMP5

CNT = 1

INLOOP5:

REAL INFILE RM PLAYER NAME 'CNT'
CNT = CNT + 1

IF CNT .LE. RM PLAYER_CNT THEN GOTO INLOOP5

JUMP5:

JUMP_LOOP1:

CLOSE INFILE

OPEN/READ/ERROR=JUMP LOOP2 INFILE3 DRA0:[CLARK]INTERFACE.DAT
READ/END OF _FILE=JUMP LOOP2 INFILE3 PLAYCNT
PLAYCNT := 'PLAYCNT'
CNT = 1

INLOOP6:

REAL INFILE3 NAME
PLAYER 'CNT' := 'NAME'

READ INFILE3 RNAME
DIRECTORY_NAME 'CNT' := 'RNAME'

CNT = CNT + 1

IF CNT .LE. PLAYCNT THEN GOTO INLOOP6
**THIS SECTION READS THE CURRENT GAME STATUS WHICH MAY BE CHANGED**
**REAL-TIME BY THE CONTROLLER. THIS SECTION RE-ESTABLISHES THE GAME**
**PARAMETERS AND INTERFACES.**

```
***
INITIAL:
  CLOSE/ERROR=INITIAL1 INFILE
INITIAL1:
  OPEN/READ/ERROR=JUMP_LOOP1 INFILE [CLARK]PARAMETER.DAT
  CNT = 1
  INLOOP1:
    READ(END_OF_FILE=JUMP_LOOP1 INFILE ARRIVAL_RATE,'CNT'
    CNT = CNT + 1
    IF CNT .LE. 5 THEN GOTO INLOOP1
  CNT = 1
  INLOOP2:
    READ(INFILE SERVICE_RATE,'CNT'
    CNT = CNT + 1
    IF CNT .LE. 5 THEN GOTO INLOOP2
  CNT = 1
  INLOOP3:
    READ(INFILE LOSS_RATE,'CNT'
    CNT = CNT + 1
    IF CNT .LE. 5 THEN GOTO INLOOP3
  CNT = 1
  INLOOP4:
    READ(INFILE GARBLE_RATE,'CNT'
    CNT = CNT + 1
    IF CNT .LE. 5 THEN GOTO INLOOP4
```
§ 1
§ INVALID_COMMAND:
§   WRITE SYSS$OUTPUT "INVALID COMMAND"
§ 1
§ HELP:
§   TYPE SYSS$INPUT

The commands you can enter are:

MAIL  READ INCOMING MAIL
SENDMSG SEND A MESSAGE
READMSG READ ENTIRE MESSAGE FILE
HELP PRINTS THIS HELP MESSAGE
QUIT TO QUIT

ENTER CTRL/Y TO RESTART THIS SESSION

§ 1
§ GOTO 'PROMPT'
§ 1
§ MAIL:
§ 1
§ ASSIGN/USER_MODE SYSS$COMMAND SYSS$INPUT
§ MAIL
§ 1
§ GOTO NEXT
§ READMSG:
§ 1
§ TYPE MAIL.MAI
§ 1
§ TYPE SYSS$INPUT
§ GOTO NEXT
§ SENDMSG:
§ 1
§ 1
$ ! SET UP FOR INITIAL PROMPT
$ !
$ PROMPT := INIT0
$ GOTO HELP0/print initial help message
$ !
$ ! AFTER THE FIRST PROMPTING MESSAGE, USE THE PROMPT: NEXT
$ !
$ INIT0:
$ PROMPT := NEXT
$ !
$ ! MAIN COMMAND PARSING ROUTINE. THE ROUTINE COMPARES THE CURRENT
$ ! COMMAND AGAINST THE OPTIONS IN THE OPTION TABLE. WHEN IT FINDS
$ ! A MATCH, IT BRANCHES TO THE APPROPRIATE LABEL.
$ !
$ NEXT:
$ ON CONTROL_Y THEN GOTO NEXT ! CTRL/Y RESTS PROMPT
$ SET CONTROL_Y
$ OR WARNING THEN GOTO NEXT ! IF ANY, REST PROMPT
$ !
$ INQUIRE COMMAND "COMMANDER, WHAT IS YOUR COMMAND?"
$ IF COMMAND .EQ. "" THEN GOTO NEXT
$ COMMAND_SIZE = F$LENGTH(COMMAND) ! INPUT LENGTH
$ INDEX = 0
$ !
$ CHECK_NEXT:
$ OPTION_LENGTH := F$EXTRACT(INDEX,1,OPT)'
$ IF OPTION_LENGTH .EQ. "" THEN GOTO INVALID_COMMAND
$ IF OPTION_LENGTH .EQ. "" THEN GOTO INVALID_COMMAND
$ INDEX = INDEX + 1 ! ADVANCE INDEX
$ NEXT_COMMAND := F$EXTRACT(INDEX,OPTION_LENGTH,OPT)
$ F$EXTRACT(0,COMMAND_SIZE,NEXT_COMMAND)
$ IF NEXT_COMMAND .EQ. "COMMAND" THEN GOTO NEXT_COMMAND E
$ INDEX = INDEX + OPTION_LENGTH ! SET TO NEXT COMMAND
$ GOTO CHECK_NEXT
ENTER THE COMMAND NAME WITH WHICH YOU REQUIRE A DEFINITION.

! SET UP FOR INITIAL PROMPT
! BRIEF := 'YES'
! PROMPT := INIT
! GOTO HELP0
! PRINT INITIAL HELP MESSAGE
!
! AFTER THE FIRST PROMPTING MESSAGE, USE THE PROMPT: NEXT
!
! INIT:
! PROMPT := NEXT
!! MAIN COMMAND PARSING ROUTINE. THE ROUTINE COMPARES THE CURRENT
!! COMMAND AGAINST THE OPTIONS IN THE OPTION TABLE. WHEN IT FINDS
!! A MATCH, IT BRANCHES TO THE APPROPRIATE LABEL.
!! NEXT:
 ON CONTROL Y THEN GOTO NEXT  I CTRL/Y RESTS PROMPT
 SET CONTROL Y
 ON WARNING THEN GOTO NEXT  I IF ANY, REST PROMPT
 ! INQUIRE COMMAND "ENTER COMMAND"
 IF COMMAND EGS. "" THEN GOTO NEXT
 IF COMMAND EGS. "BRIEF" THEN BRIEF := "YES"
 IF COMMAND EGS. "VERBOSE" THEN BRIEF := "NO"
 IF COMMAND EGS. "BRIEF" THEN GOTO NEXT
 IF COMMAND EGS. "VERBOSE" THEN GOTO NEXT
 COMMAND_SIZE = `$LENGTH(COMAND)`  I INPUT LENGTH
 INDEX = 0
!
 CHECK_NEXT:
 OPTION_LENGTH := `$LENGTHINDEX(1, OPT)"`
IF OPTION_LENGTH .EQ. 0 THEN GOTO INVALID_COMMAND
IF OPTION_LENGTH .EQ. THEN GOTO INVALID_COMMAND
INDEX = INDEX + 1 ! ADVANCE INDEX
NEXT_COMMAND := '"$EXTRACT(INDEX,OPTION_LENGTH,OPT)"
$EXTRACT(index,COMMAND_SIZE,NEXT_COMMAND)
IF NEXT_COMMAND .EQ. '"COMMAND"' THEN GOTO 'NEXT_COMMAND'
INDEX = INDEX + OPTION_LENGTH ! SET TO NEXT COMMAND
GOTO CHECK_NEXT
!
INVALID_COMMAND:
WRITE SYS$OUTPUT " INVALID COMMAND"
!
HELP:
TYPE SYS$INPUT

The commands you can enter are:

AIR  CAUSE AN AIR ATTACK ON A SPECIFIED HEX
AIRLIFT  AIRLIFT A UNIT TO A SPECIFIED HEX
CANCEL  CANCEL ORDERS TO A SPECIFIED UNIT
COMM  SEND AND/OR RECEIVE MESSAGES ON COMMUNICATIONS NET
DCA  ASSIGN AN UNIT TO DEFENSIVE COUNTER ALERT
DESTROY  DESTROY BUILDINGS, BRIDGES ETC. AT COORDINATES
ESCORT  ASSIGN AIR ESCORT TO ANOTHER AIR UNIT
FIRE  FIRE ON A DESIGNATED HEX COORDINATE
INTEI  REQUEST INTELLIGENCE INFORMATION
LOGREP  REQUEST A LOGISTICS REPORT
MINES  INFLIT MINES AT A GIVEN HEX
MOVE  DIRECT UNIT TO MOVE TO A SPECIFIED HEX
RESUPPLY  TRANSFER SUPPLIES BETWEEN FRIENDLY UNITS
SEALIFT  MOVE UNIT ON NAVY SHIP
SEND  SEND MESSAGE TO OPPOSING FORCE
SITREP  REQUEST A SITUATION REPORT
THRESHOLD  SET/Determine a units WITHDRAWAL LEVEL
TIME REQUEST CURRENT BATTLE TIME
HELP PRINTS THIS HELP MESSAGE
DONE EXIT HELP ROUTINE

ENTER CTL/ALT/ 
VERBOSITY DETAILED DESCRIPTION OF COMMAND
BRIEF BRIEF DESCRIPTION OF COMMAND (DEFAULT)

$ 1
$ GOTO 'PROMPT'
$ !
$ AIRe: 
$ TYPE SYS$INPUT

*******************************************************************************************
*                                           *
*   AIR  ON COORDINATES                       |
*                                           *
*                                           |
*   |ROAD                                        |
*   |BRIDGE                                      |
*   |NUCLEAR                                     |
*   |CHEMICAL                                    |
*   |NINE                                        |
*                                           |
*******************************************************************************************

$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

This order causes an air attack on the specified hex consisting of the
ordered number of aircraft sorties (not unit sorties) from the specified
friendly unit. Unit ID must be of type AIRCRAFT, HELICOPTER, or AC CARRIER;
must be a friendly unit; must have sufficient fuel and ammunition; and must
not be engaged in combat. When coordinates follow the word FROM, all air units at that location participate in the attack. If no start time is specified, NOW is assumed. If the number of sorties specified is greater than the number of available aircraft in the unit, all available aircraft are sent out, return, are rearmed and refueled (time delay), and reattack the target. This process repeats until the number of sorties requested is satisfied. If the number of sorties is not specified, all aircraft available in the specified unit are sent out one one sortie. During the first 3 days of combat, 80 percent of the aircraft in a unit is available for action at any given time; after that, availability drops to 60 percent. If the words ROAD or BRIDGE appear in the order, the aircraft mission is to destroy roads, bridges, tunnels, etc., rather than combat units. If the words NUCLEAR or CHEMICAL appear in the order, this is to be a nuclear or chemical airstrike. The model checks to see if the requesting force has been granted nuclear or chemical permission by the game controller. If it has not been, then the order is ignored, and the requestor is so informed. If the word MINES appears in the order, the mission is to air emplace a minefield. Every air mission has a certain probability of detecting and reporting enemy units flown over. That probability is less at night or in adverse weather. Every enemy ground unit flown over has ground air defense weapons that have a chance of shooting down aircraft. The percent of aircraft shot down is less at night or in adverse weather. Ingress and egress flight paths are different. Aircraft making it to the target do a fixed percent damage to the target and slow up that target by 15 minutes if it were moving, firing artillery, or sending out air sorties.

EXAMPLES:

AIR ON 86 FROM #211 10 SORTIES
10 SORTIES AIR ON 86 FROM #211 START NOW
SURPRISE THEM. START 11:50 HOURS 10 SORTIES FROM #211 ON 86 AIR
AIR ON 86 ROADS FROM #211
AIR FROM HH70 ON F6F NUCLEAR
AIR FROM #211 ON F86 MINE

$ GOTO NEXT
$ AIRLIFT0:
$ TYPE SYS$INPUT

******************************************************************************
* * AIRLIFT  # UNIT ID ON # UNIT ID2 TO COORDINATES *
* * {STAY START |Time |
* * |Now |
* *
******************************************************************************

$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

The unit chosen to do the airlifting must be of type AIR FORCE OR helicopter. The unit to be airlifted must be at the same location and on the same side (Red or Blue) as the lifting unit. Navy units cannot be airlifted. The lifting unit must have sufficient POL to reach its destination and return. If all of the above criteria are met, the airlift order is approved and the lifted unit arrives at its specified destination 2 hours plus flight time after the airlift began. If, at the time of landing, the landing zone is occupied by one or more enemy units, the aircraft return to base without dropping off the lifted unit. Whatever damage is done to the lifting unit by air defenses and air-to-air engagements is also done to the unit being lifted until it is dropped off. If the word STAY appears in the order, the airlifting unit stays at the landing zone with the airlifted unit. If not,
the airlifting unit returns to its original base.

EXAMPLES:

AIRLIFT #101 ON #703 TO HH86
TO HH86 ON #703 #101 AIRLIFT
#101 TO HH86 ON #703 AIRLIFT START 11.50 HOURS
ORDER & SPACING DO NOT MATTER AIRLIFT ON #703 TO HH86 #101
AIRLIFT ON #703 #701 TO HH86 STAY

$GOTO NEXT
$ CANCEL:
$ TYPE SYS$INPUT

******************************************************************************

*                              *
*                  CANCEL    # UNIT    |ALL                               *
*                              *
******************************************************************************

$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

The CANCEL order tells the specified unit number to stop what it is
doing and to wait for a new order. If the word ALL is added to the CANCEL
order, the specified unit not only stops what it is currently doing, it
also scrubs all planned missions.
EXAMPLES:

CANCEL #701
#701 CANCEL ALL
ALL CANCEL #701
#701 CANCEL.

\$ GOTO NEXT
\$ DCA0:
\$ TYPE SYS$INPUT

******************************************************************************
* * DCA   # UNIT ID *
* *  
******************************************************************************

\$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
\$ TYPE SYS$INPUT

DESCRIPTION:

The specified unit number is assigned to Defensive Counter Alert.
All enemy aircraft passing into range of the specified air unit and detected
by a radar unit will be engaged by the DCA aircraft, unless they have already
been scrambled.

EXAMPLES:

DCA #712
#712 DCA
ASSIGN #712 TO DCA

$ GOTO NEXT
$ DESTROY:
$ TYPE SYS$INPUT

*****************************************************************************
* *
* DESTROY COORDINATES *
* *
*****************************************************************************

$ IF BRIEF .EQS. 'YES' THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

This order causes friendly units at the specified coordinates to
destroy all roads, bridges, tunnels, etc., to reduce trafficability through
the specified area. If no friendly units are present at the specified area,
the requesting commander will be so informed and the order ignored. If it is
desired to have a unit burn the bridges behind it, tell that unit to move
first, then tell it to destroy the bridge. Otherwise, the unit will execute
the orders in the sequence in which they are received—blow bridge then cross
river. This could make the simulated men in that unit quite angry (and wet).

EXAMPLES:

DESTROY H70
DESTROY BRIDGE H70
H70 ROAD DESTROY
§ GOTO NEXT
§ E S C O R T  
§ T Y P E S Y S $ I N P U T

***

* E S C O R T  # U N I T  I D  W I T H  # U N I T  I D *

***

§ I F  B R I D  " Y E S "  T H E N  G O T O  N E X T
§ T Y P E  S Y S $ I N P U T

D E S C R I P T I O N :

This order causes the specified air unit preceded by the word "WITH" to escort the other air unit until told otherwise. The presence of escort aircraft with an Air force unit helps protect the mission aircraft (and repair and attack) from other air interceptors and ground air defenses. Unescorted air units will jettison their ordnance and return to base if attacked by enemy fighters.

E X A M P L E S :

E S C O R T  # 7 4 3  W I T H  # 9 4 5

W I T H  # 9 4 5  E S C O R T  # 7 4 3

W I T H  # 9 4 5  # 7 4 3  E S C O R T

§ G O T O  N E X T
§ F I R E :
§ T Y P E  S Y S $ I N P U T
This command causes artillery, missile, or naval gunfire on the designated hex coordinates. The number of volleys refers to the number of unit volleys (not tube volleys) to be fired on the target. If the number of volleys is not specified, one volley is assumed. If the start time is not specified, START NOW is assumed. When coordinates follow the word FROM, all indirect fire units at that location participate in the fire mission. Every new fire request takes 15 minutes for communication delays and to plan the fire mission. The time between volleys on the same target is 3 minutes for tube artillery-type units, and 6 hours for rockets and missiles. If a new fire order for this unit is received before the old order is completed, the unit will start firing on the new target after the last order is completed. In order for a fire order to be executed, the unit specified must be friendly, indirect-firing type, non-moving, not engaged in ground combat, within range of its weapons to the target, and have sufficient ammunition on hand. Each volley impinging on the target damages all units in the specified hex and slows them up by 15 minutes. If the words NUCLEAR or CHEMICAL appear in the order, this is to be a nuclear or chemical mission. The model then checks whether or not the requesting force (Blue or Red) has been granted nuclear or chemical permission by the game controller. If permission has not been granted, the order is ignored and the requester is informed.
EXAMPLES:

FIRE ON Z50 FROM #115
ON Z50 FIRE 1 VOLLEY FROM #115 START 9.30 HOURS
FIRE 7 VOLLEYS FROM #116 ON Z50 HOLD IT DOWN I'M TRYING TO SLEEP
FIRE FROM AA45 ON Z50 CHEMICAL
START 17.31 HOURS TO FIRE ON Z50 FROM AA45

$ GOTO NEXT
$ INTEL0:
$ TYPE SYS$INPUT

******************************************************************************

******************************************************************************

$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

This order causes one or more human agents to be sent to the
specified coordinates to report on all enemy activity in that hex. When a
war gamer receives information from HUMINT, it is 1 hour old, and locations,
activities and strengths may have changed.

EXAMPLES:
INTEL AA?1

AA?1 INTL

YE OLDE CLOAK AND DAGGER INTL ON AA?1

\$ GOTO NEXT
\$ LOGREP:
\$ TYPE SYS$INPUT

******************************************************************************
*
*                            LOGREP
*
******************************************************************************

\$ IF HIFI .EQS. "YES" THEN GOTO NEXT
\$ TYPE SYS$INPUT

DESCRIPTION:

This order causes a complete logistics report to be generated on all friendly units. The unit number, number of people in that unit, number of tons of 12 classes of supplies (1 through 10 plus 3A and 5A), and the maximum carrying capacity of the unit are printed out for each friendly unit on the map.

EXAMPLES:

LOGREP

PLEASE GIVE ME A LOGREP
GOTO NEXT

MINE:

TYPE SYS$INPUT

******************************************************************************
*
*
*      FILE Coordinates
*
*
******************************************************************************

IF BRIEF .FCS. "YES" THEN GOTO NEXT

TYPE SYS$INPUT

DESCRIPTION:

This order causes a ground or Navy unit at the specified coordinates to emplace a minefield. There must be a friendly unit that is not engaged in combat, with at least one ton of class 5 supplies, in the specified hex or the program will state the reason the order cannot be executed. Artillery-emplaced minefields can be emplaced by contacting the controllers. Air and helicopter-emplaced minefields are ordered with the AIR command. Any unit (including a friendly unit) which enters a mined hex will suffer a 32% loss and a 2 to 4-hour delay while clearing the minefield. Any unit unfortunate enough to enter a minefield that is overwatched by enemy forces will have its combat strength halved.

EXAMPLES:

MINE HH40
HH40 MINE
MINE HH40 WHERE'S MY SHOVELY
NAVY UNITS CAN MINE 236 TOOL
§ COTO NEXT
§ MOVIN:
TYPE SYSSINPUT

******************************************************************************
* MOVE |Unit ID| TO Coordinates | AT | Speed *
* | FROM Coordinates |
* |
* START | Time |
|
* |
******************************************************************************

§ If CHLD.ECS. "YES" THEN COTO NEXT
§ TYPE SYSSINPUT

DESCRIPTION:

This order causes a unit to begin moving to the specified location if it is not out of FOL. If no speed is specified, the unit will move at its maximum speed. If no start time is specified, the move will start now. The unit will choose the fastest route (easiest terrain) from where it is to its destination. If another route is desired, the move must be broken up into a series of move statements. The second move can be entered before the first is completed. Movement is stopped whenever an enemy unit is contacted, and a warning message is given to both Red and Blue commanders. Movement after contact can be started again by issuing a new MOVE order. Movement is slowed by enemy air attacks; incoming artillery; minefields; nuclear, chemical, or biological contamination; terrain features; and adverse weather. Movement consumes FOL. When coordinates follow the word FROM, all units at that location begin moving to the specified destination.

EXAMPLES:
MOVE #101 TO #116
START NOW TO #116 AT 20 MPH MOVE #101
WALK SOFTLY AND CARRY A BIG STICK MOVE #101 TO #116
MOVE ALL UNITS FROM #116 TO 1161 START 11.90 HOURS
MOVE FROM #116 TO 1161

$ GOTO NEXT
$ RESUPPLY:
$ TYPE SYS$INPUT

******************************************************************************
*
RESUPPLY #Unit 1D FROM #Unit 1D Number TONS Class Number *
*
******************************************************************************

$ IF BRIEF .ECS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

Generally, the unit from which the supplies are coming is a trans-
portation unit. However, supplies can be transferred between any two friendly
units. If there is enough of the specified class of supplies in the giving
unit, the number of tons specified will be given to the receiving unit. If no
class of supply is specified, all classes will be transferred. If no tonnage
is specified, 99% of the specified class of supplies in the giving unit will be
transferred to the receiving unit. A warning message will be printed for each
unit that gets dangerously low (less than 12 hours at current consumption rate)
in any class of supply. If a unit runs out of POL in combat, its combat
strength is halved. If a unit runs out of AMMO in combat, it is destroyed.
So, close attention must be given to logistics problems. Since units must be
allocated in order to transfer supplies, attention must be given to resupplying
a unit before that unit runs completely out of any class of supplies. Supply
shipments may be delayed, damaged, or destroyed enroute due to enemy actions.

EXAMPLES:

RESUPPLY #102 30 TONS OTHER FROM #510
45 TONS POL TO RESUPPLY #103 FROM #511
FROM #512 90 TONS RESUPPLY #104
HOT FOOD AT LAST RESUPPLY #117 2 TONS OTHER FROM #513
RESUPPLY #104 FROM #512

$ GOTO NEXT
$ SEALLT:
$ TYPE SYS$INPUT

******************************************************************************
$ SEALLT #Unit ID ON #Unit ID TO Coordinates START 'Time'
$ NOW
******************************************************************************

$ IF BRHFE .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:
The unit chosen to do the sealifting must be of type NAVY and be adjacent to the unit to be sealifted. Both units must be on the same side (Red or Blue). Whatever damage is done to the NAVY unit enroute to its destination is also inflicted upon the unit being carried. The time of arrival at the specified destination is a function of the speed of the NAVY ships and the distances involved. If the specified destination is occupied by one or more enemy units by the time the sealift arrives, the commander will be so informed, and the lifted unit will stay on board and wait for a new SIALIFT order. NOTE: The destination specified in the order must be a land destination, NOT a water destination.

EXAMPLES:

SIALIFT #102 ON #504 TO Z30

#102 SIALIFT ON #504 TO Z30. THE MARINES HAVE LANDED.

START 22.70 HOURS TO Z30 #102 BY SIALIFT TO Z30

$ GOTO NEXT
$ SEND:
$ TYPE SYS$INPUT

******************************************************************************
*                                                                         *
*   SEND  Message                                                        *
*                                                                         *
******************************************************************************

$ IF BRIEF .EQS. 'YES' THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:
This order causes the message line to be sent to the opposing force's computer terminal, with a heading of FROM BLUE COMMANDER or FROM RED COMMANDER.

**EXAMPLES:**

SEND: YOU ARE SURROUNDED—DO YOU WISH TO SURRENDER?
SEND: NUTS!

`$ GOTO NEXT
$ SITREP:
$ TYPE SYS$INPUT

***************************************************************************
***************************************************************************
***************************************************************************

`$ IF BRIEF .EQS. "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

**DESCRIPTION:**

If no unit is specified, this order causes a printout of accurate data on all friendly units. Data such as the unit ID, name, location, percent strength, activity, destination, weather, and battle time are listed. If a unit number is specified, more detailed information on just that unit will be printed out. Of course, the model will not let the player check on enemy units with this order.

**EXAMPLES:**
SITREP
SIT REPORT
SITUATION
SITUATION REPORT
SITREP #12

$ GOTO NEXT
$ THRESHOLD:
$ TYPE SYS$INPUT

************************************************************************************************************
* *
* THRESHOLD |#Unit ID | Percent *
* |ALL |
* *

************************************************************************************************************

$ IF BRIEF EQU "YES" THEN GOTO NEXT
$ TYPE SYS$INPUT

DESCRIPTION:

The THRESHOLD concept allows commanders to determine whether a unit is to hold at all costs, avoid decisive engagement, or something in between these extremes. The percent specified in this order is the percent strength at which the specified unit will break contact with the enemy. If the word ALL is used in place of a specific unit number, all friendly units will break contact at that percent strength.

EXAMPLES:
THRESHOLD #101 Y5%  
THRESHOLD ALL 50%  
#102 THRESHOLD 0%  

$ GOTO NEXT  
$ TIME:  
$ TYPE SYS$INPUT

******************************************************************************
*                           *
*                          TIME                                      *
*                           *
******************************************************************************

$ IF BRIEF.ECS. "YES" THEN GOTO NEXT  
$ TYPE SYS$INPUT

DESCRIPTION:  
This order causes the program to print the current battle time.

EXAMPLES:  
TIME

WHAT TIME IS IT?
$ GOTO NEXT  
$ !  
$ COMMENT:  
$ TYPE SYS$INPUT
DESCRIPTION:

This order causes the program to activate the communications assets in your command. Allowing you to read incoming messages and/or send messages over available communications circuits.

NOTICE: The following options are also available to exercise at your convenience.

- READ MESSAGE
- SEND MESSAGE
- READ MAIL

EXAMPLES:

- COMM
- READ MESSAGE
- SEND MESSAGE
- READ MAIL

GOTO NEXT

!
$ DONE:
$!
$ TYPE SYS$INPUT

$ EXIT

*** CKMSG.COM ***

$ START_CKMSG:
$!
$ NOCIRCUIT0 := "YOU HAVE NO CIRCUITS TO"
$ NOCIRCUIT1 := "YOU HAVE NO ENCRYPTED LANDLINE TO"
$ NOCIRCUIT2 := "YOU HAVE NO NON-ENCRYPTED LANDLINES (AUTODIN) TO"
$ NOCIRCUIT3 := "YOU HAVE NO DIGITAL HF CIRCUITS (HF/VHF/UHF) WITH A/J TO"
$ NOCIRCUIT4 := "YOU HAVE NO DIGITAL HF CIRCUITS (HF/VHF/UHF) W/O A/J TO"
$ NOCIRCUIT5 := "YOU HAVE NO VOICE CIRCUIT TO"
$ NOCIRCUIT6 := "YOU DO NOT HAVE A PERFECT LINK TO"
$!
$ IF RM_PLAYER_CNT .EQ. 0 THEN GOTO NONE_REMOVED
RMCNT = 1
CK_REMOVED:
$!
$ IF " FROM" .EQS. RM_PLAYER_NAME RMCNT THEN GOTO NOT_A_PLAYER
RMCNT = RMCNT + 1
IF RMCNT .LE. RM_PLAYER_CNT THEN GOTO CK_REMOVED
$!
NONE_REMOVED:
$!
THIS STEP DETERMINES THE IDENTITY OF THE PLAYER Sending THE MESSAGE
IT MUST MATCH THE GAME USER/PLAYER TABLE
CNT = 1
INDEX = Ø

FROM_LOOP:

1 IF FROM .EQ. PLAYER'CNT' THEN INDEX = 'CNT'
   IF INDEX .NE. Ø THEN GOTO OVER_FROM
   CNT = CNT + 1
   IF CNT .GT. 'PLAYC' THEN GOTO NOT_A_PLAYER
   GOTO FROM_LOOP

NOT_A_PLAYER:
   WRITE SYS$OUTPUT " ATTENTION: ' FROM'
   TYPE SYS$INPUT

   DID YOU SPELL YOUR NAME CORRECTLY. IF NOT, RESUBMIT YOUR MESSAGE.

   1 IF HOWEVER, IT IS SPALLED CORRECTLY, THE FOLLOWING NOTE APPLIES:

   YOU ARE NO LONGER AN ACTIVE PLAYER IN THIS GAME!!!!!
   WAIT FOR FURTHER INSTRUCTIONS FROM THE CONTROLLER.

SEVERITY == 2

EXIT

OVER_FROM:

1 IF REAL_FROM .NE. DIRECTORY_NAME'CNT' THEN GOTO INVALID_FROM

INDEX1 = 'CIRCUIT_TYPE' - 1 ! DEFINE AN INDEX FOR THE EXTRACT LEXICAL FUNCTION TO DETERMINE LINK AVAILABILITY.

1 IF CIRCUIT_TYPE .EQ. Ø THEN INDEX1 = Ø ! USE AVAILABLE CIRCUIT
OPEN/ WRITE OUTFILE MAILING.DAT
OPEN/ WRITE OUTFILE1 MAILING.IMP
!
WHITE OUTFILE1 CIRCUIT_TYPE 1 STORE CIRCUIT TYPE FOR MSG HANDLER
!
LOCATE_LOOP1:
  CNT = 1
  NUMCNT = 0
  IF CNT .GT. 'PLAYCNT' THEN GOTO DONE
!
******************************************************************************
  INDEX2 = 'INDEX1'
!
  IF 'F$LOCATE(PLAYER1,TO)' .EQ.  'F$LENGTH(TO)' THEN GOTO LOCATE_LOOP
  IF PLAYER1 .EQ.  $ FROM THEN GOTO LOCATE_LOOP

TEST_NEXT:

  TEST_LINK :=  'F$EXTRACT('INDEX2',1,LINK,'INDEX',TO1)'
  IF 'TEST_LINK' .EQ.  $ ANL. CIRCUIT_TYPE .NE.  0 THEN GOTO NO_LINK
  IF 'TEST_LINK' .EQ.  1 THEN GOTO GOT_LINK
  INDEX2 = INDEX2 + 1
  IF INDEX2 .GT.  5 THEN GOTO NO_LINK
  IF INDEX2 .LT.  5 THEN GOTO TEST_NEXT

GOTO LOCATE_LOOP
!
GOT_LINK:

  WHITE OUTFILE LINK 'INDEX', TO1
  WHITE OUTFILE DIRECTORY NAME1 : MSG ADDRESSED TO THIS PLAYER
  WRITE SYS$OUTPUT "'PLAYER1' --- OK"
  NUMCNT = NUMCNT + 1
!
!
LOCATE_LOOP:

  INDEX2 = 'INDEX1'
  CNT = CNT + 1
  IF CNT .GT.  'PLAYCNT' THEN GOTO DONE

!
TEST_LOCATE := "$\$\$LOCATE(PLAYER\'\'CNT\',TO)\""
IF TEST_LOCATE .EQ. $\$LENGTH(TO) THEN GOTO LOCATE_LOOP
IF PLAYER\'CNT .EQ. FROM THEN GOTO LOCATE_LOOP

TEST_NEXT1:
TEST_LINK := "$\$\$\$EXTRACT(INDEX2,1,INDEX\"INDEX\'TO\'\'CNT\')\""
IF TEST_LINK .EQ. ANL. CIRCUIT_TYPE .NE. \"\" THEN GOTO NO_LINK
IF TEST_LINK .EQ. 1 THEN GOTO GOT_LINK1
INDEX2 = INDEX2 + 1
IF INDEX2 .EQ. 5 THEN GOTO NO_LINK
IF INDEX2 .EQ. 5 THEN GOTO TEST_NEXT1
GOTO LOCATE_LOOP

GOT_LINK1:
\WHILE OUTFILE LINE_INDEX\'TO\'CNT"
\WRITE OUTFILE DIRECTORY_NAME\'CNT\" ! MSG ADDRESSED TO THIS PLAYER
PRT_NAME := "PLAYER\'CNT"
PRT_NAME := PRT_NAME
\WRITE SYS\$OUTPUT "\"PRT_NAME\" --- OK"
NUMCNT = NUMCNT + 1
GOTO LOCATE_LOOP

DONE:
IF NUMCNT .EQ. 0 .OR. NUMCNT .EQS. \"\" THEN GOTO NO_ADDRESS
\WRITE OUTFILE1 NUMCNT
CLOSE OUTFILE
CLOSE OUTFILE1
APPEND MAILING.DAT MAILING.TMP
SUBMIT MAILER.COM

THE MESSAGE HANDLER ROUTINE IS A FORTRAN PROGRAM WHICH
COMPUTES THE DELAY TIME AND CALLS A MESSAGE GARBLER
ROUTE
RUN DBAO:CLARK.CCCMSGHAN

SEVERITY = 2
EXIT

NO_LINK:
WRITE SYS$OUTPUT 'NO CIRCUIT 'CIRCUIT_TYPE'
WRITE SYS$OUTPUT 'PLAYER 'CNT'
GOTO LOCATE_LOOP

NO_ADDRESSEE:
CLOSE OUTFILE
CLOSE OUTFILE
WRITE SYS$OUTPUT 'INVALID ADDRESSEE: 'TO'

SEVERITY = 3
EXIT

INVALID_FROM:
TYPE SYS$INPUT

********************

WHEN SENDING A MESSAGE YOU MUST USE YOUR CORRECT IDENTITY.
PLEASE USE YOUR CORRECT PLAYER NAME WHEN "FROM" IS REQUESTED.

********************

INQUIRE/GLOBAL FROM " RE-ENTER <FROM> (ENTER <CH> ONLY TO ABOPT )"


```$1
$IF FROM .MES. "" THEN GOTO START_CRM
$SEVERITY == 3
$TYPE SYS$INPUT

***** MESSAGE ABORTED *****

$EXIT
$1

*** SENDMSG.COM ***

$SET MESSAGE /NOTEXT/NCFACILITY/NOIDENTIFICATION/NOSEVERITY
$DELETE SENDMSG.LCG.*
$1
$ASSIGN DUMMY SYS$PRINT 1 PREVENTS JOB LOG PRINTOUT
$1
$1 ******I KNOW WHO THE MESSAGE IS FROM *******
$1 ******DETERMINE WHO THE MESSAGE IS GOING TO *******
$1 ******WHAT CIRCUIT IS AVAILABLE TO HIM *******
$1 ******WHAT IS THE LOSS RATE FOR THAT CIRCUIT *******
$1 ******WHAT IS THE GREATEST RATE FOR THAT CIRCUIT *******
$1 ******WHAT ARE THE ARRIVAL AND SERVICE RATES FOR THE CIRCUIT *******
$1
$SUB_HEAD0 := 'EXPEDITED -- FIRST AVAILABLE CIRCUIT'
$SUB_HEAD1 := 'ENCRYPTED LANDLINE WITH PRECEDENCE CONTROL'
$SUB_HEAD2 := 'NON-ENCRYPTED LANDLINE (AUTODIN)'
$SUB_HEAD3 := 'DIGITAL H/F CIRCUITS (HF/VHF/UHF) WITH A/J'
$SUB_HEAD4 := 'DIGITAL H/F CIRCUITS WITHOUT A/J'
$SUB_HEAD5 := 'VOICE'
$SUB_HEAD6 := 'PERFECT LINK'
$1
$IELFTE HEALER.TXT.'PC'
$1
```
OPEN/WRITE OUTFILE HEADER.TXT.'P6' 1 CREATE HEADER FOR MESSAGE
WRITE OUTFILE SUB HEAD.'P8'
WRITE OUTFILE ''
WRITE OUTFILE ''TIME()''
WRITE OUTFILE ''
WRITE OUTFILE ''
WRITE OUTFILE ''LINE ''P4''
WRITE OUTFILE ''TO ''P7''
WRITE OUTFILE ''
WRITE OUTFILE ''
WRITE OUTFILE ''BT''
!
CLOSE OUTFILE
!
FILNAM := 'P2'
ISTAT := '$$EXTRACT(5,1,FILNAM)'
!
COPY/REPLACE HEADER.TXT.'P6',MESSAGE.TXT,STAT.ISTAT.TXT SENDMSG.TXT.'P6'
!
SEND COPY OF ORIGINAL MESSAGE TO MTM MAIN GAME DIRECTORY
FOR LATER ANALYSIS
!
MAIL /SUBJECT:'''P1'' SENDMSG.TXT.'P6' CIANK 1 GAME DIRECTORY
!
IF '''P3''' .EQS. ''CONTROLLER'' .OR. '''P3''' .EQS. '' CONTROLLER'' THEN -
GOTO DONE
!
GENERATE FILE MESSAGE.TXT USING ABOVE PARAMETERS
!
SEND COPY OF GARNISHED MESSAGE TO MTM MAIN GAME DIRECTORY
FOR LATER ANALYSIS
!
!
COPY/REPLACE HEADER.TXT .'PG'.'F2' SENDMSG.TXT.'PG'

MAIL /SUBJECT:'''P1''' SENDMSG.TXT.'PG' CLARK 1 SEND GARBLED MSG TO GAME DIRECT

THE WAIT TIME WILL BE A PARAMETER
WAIT 'P2'
MAIL /SUBJECT:'''P1''' SENDMSG.TXT.'PG' 'P3'

DONE:

SET MESSAGE /TEXT/FACILITY/IDENTIFICATION/SEVERITY

END

*** INOUTP.FOR ***

INCLUDE 'COM1.FOR/List'
INCLUDE 'COM2.FOR/List'
INCLUDE 'COM3.FOR/List'
INCLUDE 'COM4.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM3.FOR/List'
INCLUDE 'CMTM4.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM1.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM2.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM62.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM63.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM64.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM65.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM19.FOR/List'
INCLUDE 'RDRAIL.FOR/List'
INCLUDE 'MYCALL.FOR/List'
INCLUDE 'LRA0:[CLARK.MTK]MTM19.FOR/List'
*** INOUTP1 ***

INCLUDE 'ACM1.FOR/LIST'
INCLUDE 'ACM5.FOR/LIST'
INCLUDE 'COP4.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTF3.FOR/LIST'
INCLUDE 'MTH4.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTF5.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTF6.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTF8.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTF9.FOR/LIST'
INCLUDE 'NDMAIL.FOR/LIST'
INCLUDE 'NMCALL.FOR/LIST'
INCLUDE 'DRA0:[CLARK.MTH]MTM19.FOR/LIST'

*** MSGHAN.FOR ***

INTEGER LMTYP, NUMADD
INTEGER*4 HRS, MINUTS, SECS, LIB$DO_COMMAND
INTEGER*4 SEED, ILOSS, IGFRAT, IPGAR
INTEGER*4 LTH, LTNUM, Seed
CHARACTER ADDR10, LNKCOD,E, ICODE1,FROM10,DATA1*8E,DATA2*80
CHARACTER NEWLTH*1, LETTER*1
REAL*4 FACTOR, FLOSS, URAN, AIR, SERV, DELAY, A, S, GRFRAT, PGAR
REAL*4 GARB,E, RANNUM,GARVAL
DIMENSION ICODE(6), FLOSS(6), IPLOSS(6), A(6), S(6),
IGFRAT(6), IGFRAT(6), IPGAR(6)
DIMENSION NEWLTH(6), GARB(6), LETTER(8E)
EQUIVALENCE (DATA2, NEWLTH(1))
EQUIVALENCE (LNKCOD, ICODE(1))
**FORTRAN Code**

```fortran
ELSE IF (1_CKET.EQ. 5) THEN
  OPEN(Unit=10, File='SEED.EAT', Status='UNKNOWN')
END IF (1_CKET.EQ.6) THEN
  OPEN(Unit=10, File='SEED.EAT', Status='UNKNOWN')
ENDIF

C READ(10,4,ERH=11,END=11)SEED
CLOSE(Unit=10)

GO TO 21

11 CONTINUE

C SEED = 16752
CLOSE(Unit=10)

21 CONTINUE

C OPEN(Unit=2, File='DRAW:CLARA PARAMETER.DAT', Status='OLD',
      IFREADONLY)
C
READ(2,1001) A(1), A(2), A(3), A(4), A(5)
READ(2,1001) S(1), S(2), S(3), S(4), S(5)
READ(2,1002) IPLSS(1), IPLSS(2), IPLCSS(3), IPLSS(4), IPLSS(5)
READ(2,1002) IGRHAT(1), IGRHAT(2), IGRHAT(3),
   IGRHAT(4), IGRHAT(5)
READ(2,1002) IPGARB(1), IPGARB(2), IPGARB(3), IPGARB(4), IPGARB(5)
1001 FORMAT(I/F/F/F/F/F)
1002 FORMAT(I/I/I/I/I)
C
REW 2
CLOSE(Unit=2)
C
LO 69 I=1,t
   PLOSS(1) = FLOAT(ILOSS(1))/100.8
```
GRHRAT(1) = FLOAT(IGRHRAT(1))/100.0
69 CONTINUE
C REAL IN THE LIST OF SUBSTITUTABLE CHARACTERS WHICH EXCLUDES CARRIAGE RETURN
C AND LINEFED, THOSE CHARACTERS ARE HANDLED SEPARATELY.
OPEN (UNIT=15, FILE='DRAO:\CLARK.CCC\LETTER.TXT', STATUS='OLD')
C THE NUMBER OF CHARACTERS CONTAINED IN LETTER() DETERMINES THE LOOP LIMIT
DO LTR=1,47
READ (15,1116) END=998, ERR=9996) LETTER (LTR)
CONTINUE
998 CLOSE (UNIT=15)
C OBTAIN LAST VALUE OF THE SEED USED IN THE GENERATION OF RANDOM NUMBERS,
C VALUE IS STORED SO THAT SEQUENCE IS NOT REPEATED EVERY TIME A LINE IS
C GARBLED
SEED7 = 16883
OPEN (UNIT=16, FILE='SEED.DAT', STATUS='OLD', ERR=1112)
READ (16, *) SEED7
CLOSE (UNIT=16)
1112 CONTINUE
C OPEN A MESSAGE ADDRESS FILE THAT IS TO CONTAIN THE
C GARBLED TEXT FILE NAME, THE DELAY TIME AND TO WHOM
C THE MSG IS ADDRESSED -- MESSAGE OUTFILE.
C OPEN (UNIT=12, FILE='MESSAGE.TXT', STATUS='NEW')
OPEN (UNIT=3, FILE='MESSAGE.TXT', STATUS='OLD', READONLY) 1 ORIG. MESSAGE FILE
C DETERMINE FOR EACH ADDRESS EE WHICH LINK TO BE USED
C
INK = 0
DO 500 I=1, NUMADD
INK = INK + 1
500 READ (1,1003) LNKCOD, ADDNAME
1003 FORMAT (A6/A)
IF (1_CKT .EQ. 0) THEN
  GC TO 2101
ELSE
  LNKTP = 1_CKT
ENDIF

2102 CONTINUE

C
C OPEN A FILE TO SAVE EACH MESSAGE PARAMETERS FOR LATER ANALYSIS
C

INSTAT = 1
IF ( INSTAT .EQ. 1 ) THEN
  OPEN(UNIT=17,FILE='STAT1.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
  OPEN(UNIT=4,FILE='GARB1.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT .EQ. 2 ) THEN
  OPEN(UNIT=17,FILE='STAT2.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
  OPEN(UNIT=4,FILE='GARB2.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT .EQ. 3 ) THEN
  OPEN(UNIT=17,FILE='STAT3.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
  OPEN(UNIT=4,FILE='GARB3.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT .EQ. 4 ) THEN
  OPEN(UNIT=17,FILE='STAT4.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
  OPEN(UNIT=4,FILE='GARBL4.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT .EQ. 5 ) THEN
  OPEN(UNIT=17,FILE='STAT5.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
  OPEN(UNIT=4,FILE='GARBL5.TXT',STATUS='NEW',
       CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT = 6 ) THEN
    OPEN (UNIT=17,FILE='STAT6.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
    OPEN (UNIT=4,FILE='GARB6.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT = 7 ) THEN
    OPEN (UNIT=17,FILE='STAT7.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
    OPEN (UNIT=4,FILE='GARB7.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT = 8 ) THEN
    OPEN (UNIT=17,FILE='STAT8.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
    OPEN (UNIT=4,FILE='GARB8.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
ELSE IF ( INSTAT = 9 ) THEN
    OPEN (UNIT=17,FILE='STAT9.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
    OPEN (UNIT=4,FILE='GARB9.TXT',STATUS='NEW',
        CARRIAGECONTROL='LIST')
ENDIF

DRAW A RANDOM NUMBER TO DETERMINE IF MESSAGE IS LOSS.
IF LOSS, GO GET THE NEXT ADDRESSEE.

URAN=HAN(S$H)
IF (URAN.LE.PLOSS(INKYP)) THEN
    WRITE(17,*) ' ',
    WRITE(17,*) ' IF URAN = ',URAN,' IS LESS THAN P(LOSS) = ',
    PLOSS(INKYP),' MESSAGE IS LOST
    WRITE(17,*) ' GARBLE RATE ',INKYP,' = ',GRRAT(INKYP)
    WRITE(17,*) ' LOSS RATE ',INKYP,' = ',PLOSS(INKYP)
    WRITE(17,*) ' LINK TYPE = ',INKYP
WRITE(14,*,') LINK CODE = ',LNKCOD
WRITE(14,*) I.CKT = ',I.CKT
CLOSE (UNIT=1?)
WRITE(12,*)'CONTROLLER'
WRITE(12,1216) 'GARF',INX,'.TXT'

C
GOTO 506
ENDIF

120 URAN=RAN(SEED)
CALL CALCULATE DELAY TIME BASE ON ARRIVAL RATE AND SERVICE RATE
OF THE CIRCUIT.
IF( URAN .LT. .91 OR URAN .GT. .99 ) GO TO 120
IF( LNKTYP .EQ. 6 ) THEN
  DELAY = .91
ELSE
  ARR = 1. / ( -1*(1./A(LNKTYP) ) * LOG10(URAN) )
  SERV = 1. / ( -1*(1./S(LNKTYP) ) * LOG10(URAN) )
  DELAY=( ARR / ( SERV * (SERV-ARR) ) ) / FACTOR
ENDIF

HRS = INT(DELAY)
MINUTS = INT(.90*(DELAY-FLOAT(HRS)))
SECS = INT(.90*.90*(DELAY-FLOAT(HRS)-FLOAT(MINUTS))/.90.)

C
C
C
C
FIXED GARFLE RATES FOR EACH CIRCUIT TYPE.
IF( LNKTYP .EQ. 6 ) THEN
  GARF=0.9
ELSE IF( LNKYP .EQ. 3 ) THEN
  PGB= FLOAT( PGBA(3) ) / 100.

ELSE IF( INATYP .EQ. 4 ) THEN
  PGB= FLOAT( PGBA(4) ) / 100.

ELSE IF( LNKYP .EQ. 3 ) THEN
  PGB= FLOAT( PGBA(3) ) / 100.

ELSE IF( LNKYP .EQ. 2 ) THEN
  PGB= FLOAT( PGBA(2) ) / 100.

ELSE IF( INATYP .EQ. 1 ) THEN
  PGB= FLOAT( PGBA(1) ) / 100.
ENDIF

C
INDEX = 1
C
130 CONTINUE
URAN= RAN(86)
C
C DETERMINE IF MESSAGE IS TO BE GARBLED
C
IF(URAN.LE.PGBA) THEN
  WRITE(1',*')
  WRITE(1',*')
  WRITE(1',*') 'MESSAGE GARBLED'
  WRITE(1',*')
  GO TO 1111
ENDIF

C
131 CONTINUE
READ(3,1004,END=2004,ERR=2004) DATA1
READ(4,1004) DATA1
GO TO 131
2004 CONTINUE
CLOSE(UNIT=4)
REWIND 2
CONTINUE
WHITE(12,+)ADDNAP
WHITE(14,16)"GARBL",INX,".TXT"

FORMAT DELAY TIME FOR DIGITAL COMMAND LANGUAGE COMMAND "WAIT"

IF (MINUTS .LT. 10 .AND. SECS .LT. 10) THEN
   WHITE(12,1620)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
   WHITE(17,*)"MESSAGE DELAY TIME FOLLOWS:
   WHITE(17,1620)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
ELSE IF (MINUTS .LT. 10 .AND. SECS .GE. 10) THEN
   WHITE(12,1621)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
   WHITE(17,*)"MESSAGE DELAY TIME FOLLOWS:
   WHITE(17,1621)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
ELSE IF (MINUTS .GE. 10 .AND. SECS .LT. 10) THEN
   WHITE(17,*)'
   WHITE(12,1622)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)"MESSAGE DELAY TIME FOLLOWS:
   WHITE(17,1622)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
ELSE
   WHITE(17,*)'
   WHITE(12,1623)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)"MESSAGE DELAY TIME FOLLOWS:
   WHITE(17,1623)HRS,':0',MINUTS,'':0',SECS,'':00'
   WHITE(17,*)'
ENDIF
SAVING STATISTICAL DATA

WHILE(17,*), IF URAN = ' URAN, LESS THAN PGATE, PGATE, GATE 11
WHILE(17,*), GATE RATE, LKTYPE, = GRB RATE(LKTYPE)
WHILE(17,*), LOSS RATE, LKTYPE, = FLOSS(LKTYPE)
WHILE(17,*), LK TYPE, LKTY P
WHILE(17,*), LK CODE, INK CODE
WHILE(17,*), I CFT, I CFT
WHILE(17,*), SERVICE RATE, SERV
WHILE(17,*), ARRIVAL RATE, ARR
WHILE(17,*), FACTOR, FACTOR
WHILE(17,*), LK TYPE, LKTY P
WHILE(17,*), DELAY, DELAY

CLOSE(UNIT=17)

CONTINUE
CLOSE(UNIT=5)
CLOSE(UNIT=12)
CLOSE(UNIT=2)
CLOSE(UNIT=1)

IF (1.CFT.EQ.0) THEN
  OPEN(UNIT=16, FILE='SERVX.DAT', STATUS='UNKNOWN')
ELSE IF (1.CFT.EQ.1) THEN
  OPEN(UNIT=16, FILE='SERV1.DAT', STATUS='UNKNOWN')
ELSE IF (1.CFT.EQ.2) THEN
  OPEN(UNIT=16, FILE='SERV2.DAT', STATUS='UNKNOWN')
ELSE IF (1.CFT.EQ.3) THEN
  OPEN(UNIT=16, FILE='SERV3.DAT', STATUS='UNKNOWN')
ELSE IF (1.CFT.EQ.4) THEN
  OPEN(UNIT=16, FILE='SERV4.DAT', STATUS='UNKNOWN')
ELSE IF (1.CFT.EQ.5) THEN
  OPEN(UNIT=16, FILE='SERV5.DAT', STATUS='UNKNOWN')
ELSE IF (I_C AT .EQ. 6 ) THEN
    OPEN (UNIT=16, FILE= 'S E C L AT ', STATUS= 'UNKNOWN ')
ENDIF

C
REWIND 16
WRITE (16, *) SEED
CLOSE (UNIT=16)
C
PLACE THE LAST VALUE OF SEED INTO 'SEED7.DAT'
OPEN (UNIT=16, FILE= 'S E D7.DAT ', STATUS= 'UNKNOWN ')
WRITE (16, *) SEED
CLOSE (UNIT=16)
GO TO 999
C
C THE I I B SE D OR I I B RUN COMANDS MAY BE USED HERE TO EXECUTE OTHER
PROCEDURES SUCH AS A STATISTICAL PACKAGE
999 STATUS = I I B SE D ( 'DUMMY := " DUMMY " ' )
C
7101 CONTINUE
C
C
IF ( I CODE (6) .EQ. '1' ) THEN
    LNKTYP = 6
ELSE IF ( I CODE (1) .EQ. '1' ) THEN
    LNKTYP = 1
ELSE IF ( I CODE (2) .EQ. '1' ) THEN
    LNKTYP = 2
ELSE IF ( I CODE (3) .EQ. '1' ) THEN
    LNKTYP = 3
ELSE IF ( I CODE (4) .EQ. '1' ) THEN
    LNKTYP = 4
ELSE IF ( I CODE (5) .EQ. '1' ) THEN
    LNKTYP = 5
ELSE
    GO TO 500
ENDIF
GO TO 2102

1111 CONTINUE

C ******************************************************************************
C CARRIB PGMULF
C ******************************************************************************
C This subroutine takes a single line of text and changes a percentage of the
C characters by randomly replacing them with other characters as well as
C carriage return and line feed.
C
C DICTIONARY
C 
C  variance: the percentage of garbling to applied against the text line
C (value will remain constant for the entire message).
C  inline: the input character line (max length = 8W).
C  link: the type of circuit the message is being transmitted over
C      (each link will have a pre-determined percent garbling)
C  offline: the output line with garbled text
C
C 151 continue
C     read(5,10K4,end=156)datak
C
C Line garbling is accomplished a single character at a time with the
C modification taking place on 'inline/offline' via 'newlin' equivalence.
C Rate of garbling set by 'variance()' for a particular link.
C do 910 I=1,8W
C     rannum=ran(seed?)
C     if(rannum.le.grbrat(inline/j))go to 910
C     rannum=ran(steady)
C     if(rannum.gt.grbrat(inline/j))go to 910
C 910 continue
c If ltrnum equals 58 limited will be substituted.
   if(ltrnum.eq.58) then
     newltr(i)=char(10)
   go to 910
   endif

c If ltrnum equals 59 limited is substituted for the previous character
   and carriage return for the current character.
   if(ltrnum.eq.59) then
     newltr(1-1)=char(10)
     newltr(i)=char(13)
   go to 910
   endif

c If ltrnum is less than 58 the new value is taken from letter(ltrnum).
   newltr(i)=letter(ltrnum)

910 continue

c Final output line is now created through data2 and newltr equivalence.
   write(4,1004) data2
   go to 151

9999 write(6,99e)
150 go to 162
1110 format(a1)
99e format(iz,'input error in letter.txt')
1004 format(a)
1006 format(iz/a10)
1010 format(az,11,4)
1021 format(iz,az,11,az,az,11,az)
1022 format(iz,az,az,11,az,az,az)
1023 format(az,az,az,az)
end
LIST OF REFERENCES


5. Lindsay, G.F., Associate Professor, Naval Postgraduate School, Unpublished Class Notes, O.A.4603, 1982.
<table>
<thead>
<tr>
<th>No.</th>
<th>Copies</th>
<th>Name</th>
<th>Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>Defense Technical Information Center</td>
<td>Cameron Station, Alexandria, Virginia 22314</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>Library, Code 0142</td>
<td>Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>Assoc Professor S.H. Parry, Code 5Epa</td>
<td>Department of Operations Research, Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>Assoc Professor F.R. Richards, Code 55Rh</td>
<td>Department of Operations Research, Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>LtCol J.W. Johnson, Code 39</td>
<td>Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>6</td>
<td>2</td>
<td>Fred McClintic</td>
<td>Wargaming Division, U.S. Army War College, Carlisle Barracks, Pennsylvania 17013</td>
</tr>
<tr>
<td>7</td>
<td>1</td>
<td>Professor M.G. Sovereign, Code 552o</td>
<td>Chairman, C3 Academic Group, Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>Professor J.M. Wozencraft, Code 74</td>
<td>Department of Electrical Engineering, Naval Postgraduate School, Monterey, California 93940</td>
</tr>
<tr>
<td>9</td>
<td>1</td>
<td>Major Robert L. Rapp</td>
<td>Tactical Communications Division, HQ Tactical Air Command, Langley AFB, Virginia 23665</td>
</tr>
</tbody>
</table>
10. Captain Licyd N. Clark
   EQ Strategic Air Command/XPFC
   Offutt AFB, Nebraska 68133

11. Captain Larry D. Piess
   C3 Division
   Rome Air Development Center
   Griffiss AFB, New York 13441

12. Captain Brian Van Crwann
    AFIT/CISR
    Wright-Patterson AFB, Ohio 45433

13. Joseph Fearney
    Defense Information System Program
    Jet Propulsion Laboratory
    4800 Oak Grove Drive
    Pasadena, California 91109

14. Commanding General
    U.S. Readiness Command
    Attn: ASG-LTC Wayne Anderson
    Mac Dill AFB, Florida 33618

15. Commander
    U.S. Army Signal Center, Fort Gordon
    Attn: DCD-Major Bruce Eisentrout
    Fort Gordon, Georgia 30905

16. Roland & Associates
    500 Sloat Avenue
    Monterey, California 93940