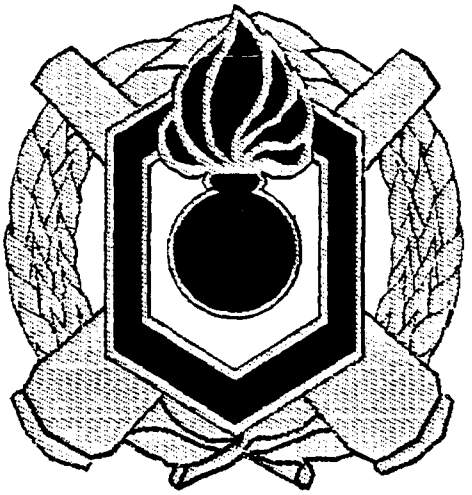
(2)

SA-HB-8801
ADSM 18-L62-LAT-ZZZ-MM-2604

PROGRAM DOCUMENTATION OF THE PLANT JOB SCHEDULING MODEL SYSTEM



SYSTEMS ANALYSIS OFFICE
AMSMC-SA
30 NOVEMBER 1987

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS: None
2a. SECURITY CLASSIFICATION AUTHORITY: N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): SA-HB-8801	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: Systems Analysis Office; 6b. OFFICE SYMBOL (If applicable): AMSMC-SA	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (City, State, and ZIP Code): U.S. Army Armament, Munitions and Chemical Command, Rock Island, IL 61299-6000	7b. ADDRESS (City, State, and ZIP Code)
8a. NAME OF FUNDING/SPONSORING ORGANIZATION: ODCSRDA; 8b. OFFICE SYMBOL (If applicable)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (City, State, and ZIP Code): HQDA, Washington, DC 20310-0653	10. SOURCE OF FUNDING NUMBERS: PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE (Include Security Classification)
Program Documentation of the Plant Job Scheduling Model System

12. PERSONAL AUTHOR(S) Dunn, Michael A.; Ireland-Dunn, Dorothy L.; Kall, Edward A.; Stiles, George E.; Trier, Norman H.; Wells, Lanny D.; and Mijares, Gerry R. (ASNC-ARI-TW)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (Year, Month, Day)	15. PAGE COUNT
Handbook	FROM May 87 TO Nov 87	871130	

16. SUPPLEMENTARY NOTATION
Inquiries may be forwarded to Commander, U.S. Army Armament, Munitions and Chemical Command, ATTN: AMSMC-SA, Rock Island, IL 61299-6000 AUTOVON: 793-5262

17. COSATI CODES			18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Ammunition, Budgeting, DBMS Automated System, POM, Planning, Programming, Workload, Readiness, Production, Scheduling.
15	05		

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

The PJSM system was developed at the request of the Vice Chief of Staff of the Army (VCSA) for use by AMCCOM and higher headquarters elements to provide an automated approach to ammunition planning, programming and budgeting, an analytical tool for decision making, and documentation of the POM and budget. This system consists of two models, a data base and generators for data screens, graphics, and reports. The Ammunition Scheduling Model is used to make initial computer runs and the Post Processor Model for subsequent "what if" evaluations. Accessing the programs for model runs and the input and output of data is done with a menu system designed to be "user friendly" for the functional users. Documentation of the PJSM system was a prerequisite to its transition from the developer (Systems Analysis Office) to the Ammunition Program Management Office for functional control and the Information Management Directorate for program maintenance.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION	
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS		
22a. NAME OF RESPONSIBLE INDIVIDUAL: Mr. Roger Powell	22b. TELEPHONE (Include Area Code): (309) 782-5980/5891	22c. OFFICE SYMBOL: AMSMC-SAL

DD Form 1473, JUN 86 *Previous editions are obsolete.*

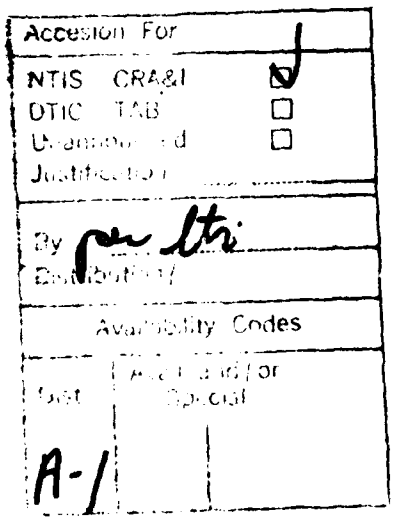
SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

DISPOSITION

Destroy this report when no longer needed. Do not return it to the originator.

DISCLAIMER

The findings in this report are not to be construed as an official Department of the Army position.



SA-HB-8801

PROGRAM DOCUMENTATION OF THE PLANT JOB SCHEDULING MODEL SYSTEM

Michael A. Dunn

Dorothy L. Ireland-Dunn

Edward A. Kall

Gerry R. Mijares

George E. Stiles

Norman H. Trier

Lanny D. Wells

November 1987

U.S. Army Armament, Munitions and Chemical Command
Systems Analysis Office
Rock Island, Illinois 61299-6000

In Appreciation

to

LINDA LENSLY

whose

word processing skills

and

exemplary performance

were responsible

for the

timely

completion

of this

documentation

TABLE OF CONTENTS

PAGE

SECTION 1. GENERAL 1-1

SECTION 2. SYSTEM DESCRIPTION 2-1

SECTION 3. ENVIRONMENT 3-1

SECTION 4. SYSTEM MAINTENANCE 4-1

SECTION 5. PROGRAM MAINTENANCE PROCEDURES 5-1

5.1 PJSM Master Menu Program (JSM1.CPL) 5-1

5.2 PJSM Table Export Program (ARCHIVE.CPL) 5-26

5.3 ORACLE Precompile, Compile, Link, and Execute Program (BIND32IX.CPL) 5-30

5.4 Change Revision (CREV.FOR) 5-33

5.5 PJSM Data Base Maintenance Program (PJSM_MAINT.CPL) 5-36

5.6 Data Base Consistency Check (BASECHK.FOR) 5-54

5.7 ORACLE User Validation Program (VALID.CPL) 5-62

5.8 Batch CPL Subprograms Interfacing JSM1.CPL (BATCH CPL PGMS) 5-65

5.9 Item Descriptors Input Screen (JSM_1.INP) 5-68

5.10 Item Planning Parameters Input Screen (JSM_2.INP) 5-74

5.11 Item Production Capacity/Staffing Input Screen (JSM_3.INP) 5-80

5.12 General Plant Data Input Screen (JSM_4.INP) 5-86

5.13 Primary Component Information Input Screen (JSM_5.INP) 5-92

PAGE

5.14 Yearly Consumption Requirements Input Screen (JSM_6.INP) 5-98

5.15 Production Project Specification Input Screen (JSM_7.INP) 5-104

5.16 Army Requirements Stratification Input Screen (JSM_8.INP) 5-111

5.17 Other Customer Requirements-Order Input Screen (JSM_9.INP) 5-117

5.18 Secondary Component Information Input Screen (JSM_10.INP) 5-123

5.19 Line Descriptors Input Screen (JSM_11.INP) 5-129

5.20 Update or Display Army Requirements Input Screen (JSM_12.INP) 5-134

5.21 Update or Display Other Service Requirements Input Screen (JSM_13.INP) 5-140

5.22 DODIC/Project Cross-Reference Input Screen (JSM_14.INP) 5-146

5.23 RAMP Item Input Screen (JSM_15.INP) 5-151

5.24 RAMP Production Input Screen (JSM_16.INP) 5-157

5.25 JSM Simulation Guidelines Input Screen (GUID_1.INP) 5-163

5.26 RDAISA (Army Requirements) Input Program (PKG.FOR) 5-169

5.27 ICAPP Unit Prices (Including Army's) and Other Customer Requirement Input Program (ICAPP.FOR) ... 5-175

5.28 Alternate Priority Scheme Generator (APS.FOR) 5-179

5.29 Individual Item Data Report (IIDR.FOR) 5-182

5.30 Plant Item Data Report (PIDR.FOR) 5-186

PAGE

5.31 Data Extract for Ammunition Executive Management System (AEMS.FOR) 5-190

5.32 Increased Workload Impact Report (IWIR.FOR) 5-194

5.33 Ammunition Scheduling Model (PS.FOR) 5-200

5.34 Plant Job Scheduling Model Post Processor (JSMPP) 5-223

5.35 Build Pointer Arrays Program (For PS and PPJSM) (BUILD_PTR.FOR) 5-240

5.36 Requirement Summarization Program (TREQ.FOR) 5-245

5.37 Army Acquisition Report (RPT1.FOR) 5-250

5.38 Workload Summary Report by Item (by Plant and Line) (RPT2.FOR) 5-253

5.39 Plant Workload Utilization Detailed Report (RPT3.FOR) 5-256

5.40 Plant Workload Summary Report (RPT4.FOR) 5-259

5.41 Analysis Worksheet Report (RPT5.FOR) 5-262

5.42 Program Development Incremental Package (PDIP) Summary Report (RPT6.FOR) 5-268

5.43 Compare Program (RPT7.FOR) 5-272

5.44 Summary of Training and Test Items not Achieving 100 Percent (RPT8.FOR) 5-276

5.45 Readiness Report (RPT9.FOR) 5-279

5.46 Other Service Shortfall Report (RPT10.FOR) 5-282

5.47 PDIP Package Structure by Item (RPT11.FOR) 5-285

5.48 PDIP Cost Report (RPT12.FOR) 5-288

5.49 Work-years Available to Increase Plant Utilization Report (RPT13.FOR) 5-292

		PAGE
5.50	Secondary Item Status Report (RPT14.FOR)	5-297
5.51	Summary of Package Requirement/Filled (RPT15.FOR)	5-302
5.52	Conventional Ammunition Acquisition Plan (RPT16.FOR)	5-306
5.53	Army Requirement Allocation Output Screen (RESULT1.INP)	5-310
5.54	Other Customer Requirement Allocation Output Screen (RESULT2.INP)	5-316
5.55	Production Summary Output Screen (RESULT3.INP) ...	5-322
5.56	Item Summary Output Screen (RESULT4.INP)	5-328
5.57	Post Processor Change Screen (RSLT_CHG.INP)	5-334
5.58	Plant Job Scheduling Model Workloading Program (PJSMWL)	5-339
5.59	Plant Job Scheduling Model Requirement Program (PJSMRQ)	5-344
5.60	Plant Job Scheduling Model Production Program (PJSMPD)	5-347
5.61	Plant Job Scheduling Model Dollar Program (PJSMDL)	5-351
5.62	Plant Job Scheduling Model Data Base Program (PJSMDB)	5-356
5.63	Plant Job Scheduling Model Charting (PJSMCHT) ...	5-365
5.64	Plant Job Scheduling Model Common (PJSMCOM)	5-368
5.65	Post Post Processor for the Plant Job Scheduling Model (POST-POST)	5-382
ANNEX A	DOCUMENTED PROGRAMS	A-1
ANNEX B	DATA BASE TABLES	B-1

PAGE

ANNEX C DECISION FLOW PROCESS FOR PRODUCTION SCHEDULING MODEL C-1

ANNEX D FILE FORMATS D-1

ANNEX E SUBROUTINE CROSS-REFERENCE F-1

FIGURES

2.2-1. Current System Structure of the Plant Job Scheduling Model 2-3

5.9-1-1. Example of Item Descriptors Input Screen Filled with Information 5-71

5.10.1-1. Example of Item Planning Parameters Input Screen Filled with Information 5-77

5.11.1-1. Example of Item Production Capacity/Staffing Input Screen Filled with Information 5-83

5.12.1-1. Example of General Plant Data Input Screen Filled with Information 5-90

5.13.1-1. Example of Primary Component Information Input Screen Filled with Information 5-95

5.14.1-1. Example of Yearly Consumption Requirements Input Screen Filled with Information 5-101

5.15.1-1. Example of Production Project Specification Input Screen Filled with Information 5-108

5.16.1-1. Example of Army Requirements Stratification Input Screen Filled with Information 5-114

5.17.1-1. Example of Other Customer Requirements-Orders Input Screen Filled with Information 5-120

5.18.1-1. Example of Secondary Component Information Input Screen Filled with Information 5-126

5.19.1-1. Example of Line Descriptors Input Screen Filled with Information 5-131

PAGE

5.20.1-1. Example of Update or Display Army Requirements Input Screen Filled with Information 5-137

5.21.1-1. Example of Update or Display Other Service Requirements Input Screen Filled with Information 5-143

5.22.1-1. Example of DODIC/Project Cross Reference Input Screen Filled with Information 5-148

5.23.1-1. Example of RAMP Item Input Screen Filled with Information 5-155

5.24.1-1. Example of RAMP Production Input Screen Filled with Information 5-160

5.25.1-1. Example of JSM Simulation Guidelines Input Screen Filled with Information 5-167

5.33.1-1. Subroutine Calls for the Production Scheduling Model 5-205

5.34.1-1. Example of Post-Processor Change Screen Filled with Information 5-224

5.53.1-1. Example of Army Requirement Allocation Output Screen Filled with Information 5-306

5.54.1-1. Example of Other Customer Requirement Allocation Output Screen Filled with Information 5-312

5.55.1-1. Example of Production Summary Output Screen Filled with Information 5-318

5.56.1-1. Example of Item Summary Output Screen Filled with Information 5-324

SECTION 1. GENERAL

1.1 Purpose of Program Maintenance Manual. This Program Maintenance Manual for the Plant Job Scheduling Model (PJSM) System, with a System Identifier Code of LAT, will provide the maintenance programmer personnel with the information necessary to effectively maintain the system.

1.2 Project References. This system was developed at the request of the Vice Chief of Staff of the Army (VCSA) for use by AMCCOM to provide an automated approach to ammunition programming and budgeting activities. The system will provide a data base of record for ammunition programming and budgeting, analytical capabilities for decisionmaking, and POM and budget documentation. The following references are applicable:

a. Functional Description, PJSM System, ADSM 18-L62-LAT-ZZZ-FD-ATA, 2 Feb 87; unclassified.

b. SA-TN-8701, U.S. Army Armament, Munitions, and Chemical Command; A Management Decision Tool for Ammunition Acquisition (The Ammunition Plant Job Scheduling Model); Systems Analysis Office; May 87; unclassified.

c. TB 18-111, Technical Documentation; U.S. Army Information Software Support Systems Command; 22 Apr 83; unclassified.

d. TB 18-103, Technical Documentation; Software Design and Development; U.S. Army Information Software Support Systems Command; 3 Jan 83; unclassified.

e. DOD 7935.1-STD, Department of Defense Automated Data Systems (ADS) Documentation Standards; Office of the Assistant Secretary of Defense (Comptroller); 24 Apr 84; unclassified.

f. ORACLE User and Reference Manuals; ORACLE Corporation, Menlo Park, CA; unclassified.

1.3 Terms and Abbreviations.

AAO	Authorized Acquisition Objective
AAP	Army Ammunition Plant
AEMS	Ammunition Executive Management System
AMCCOM	Armament, Munitions and Chemical Command

AMC	Army Materiel Command
APMO	Ammunition Program Management Office
COB	Command Operating Budget
COR	Contracting Officer Representative
DA	Department of the Army
DCSOPS	Deputy Chief of Staff for Operations and Plans
DCSRDA	Deputy Chief of Staff for Research, Development and Acquisition
DDN	Defense Data Network
DODAC	Department of Defense Ammunition Code
DODIC	Department of Defense Identification Code
DOS	Days of Supply
DSACS	Defense Standard Ammunition Computer System
DSS	Decision Support System
FMS	Foreign Military Sales
FSC	Federal Supply Class
GAO	Government Accounting Office
HQ	Headquarters
HQDA	Headquarters of the Department of the Army
IAF	ORACLE Interactive Application Facility
IAG	ORACLE Interactive Application Generator
IAP	ORACLE Interactive Application Processor
ICAPP	Integrated Conventional Ammunition Procurement Plan
IRD	Industrial Readiness Directorate

ISN	Item Sequence Number
JCS	Joint Chiefs of Staff
JSPD	Joint Strategic Planning Document
MCA	Military Construction Army
MPQ	Minimum Procurement Quantity
NMF	New Material Fielding
NSN	National Stock Number
ODCSOPS	Office of the DCSOPS
ODCSRDA	Office of the DCSRDA
OSD	Office of the Secretary of Defense
PARR	Program Analysis and Resource Review
PBD	Program Budget Decision
PBMA	Production Base Modernization Agency
PD	Production Directorate
PDE	Pricing Division
PDIP	Program Development Incremental Package
PDM	Program Decision Memorandum
PJSM	Plant Job Scheduling Model
POM	Program Objective Memorandum
PPBS	Planning, Programming, and Budgeting System
RAMP	The 2 years preceeding start of the POM period
RCN	Revision Control Number
R&D	Research and Development
RDAISA	Research, Development and Acquisition Information Systems Agency

RIA	Rock Island Arsenal
SAO	Systems Analysis Office
SCN	Sequence Control Number
SSN	Standard Study Number
TAMIS	Training Ammunition Management Information Systems
TOA	Total Obligation Authority
UFI	ORACLE's User Friendly Interface
VCSA	Vice Chief of Staff of Army
WARS	Worldwide Ammunition Reporting System

SECTION 2. SYSTEM DESCRIPTION

2.1 System Application. The Plant Job Scheduling Model (PJSM) system will provide management with a decision tool to integrate functional objectives and planning, programming, and budgeting guidance, with HQDA training and combat requirements, other DOI customer needs and production base capabilities, to result in an achievable multi-year ammunition program.

a. This sytem will maintain a data base of all ammunition items which are available for production during the POM years. Production facilities must be available during the POM period. Workload considerations are restricted to GOCO AAPs and GOGO arsenals/activities. Commercially produced items are reported by production capacities. Component breakout is restricted to GOCO materiel and commercial materiel considered a limiter (pacer). Other DOD customer requirements compete for GOCO/GOGO facilities and resources, and are considered priority orders.

b. Major functions of the sytem are the ammunition scheduling model, a user interface encompassing input and output functions, data base management and data storage, report generation, display graphics, and intersystem telecommunications facilities.

(1) The ammunition scheduling model represents the main analytical capability of the system. A rule based decision model is used to provide a management tool to estimate yearly production quantities and a mix of ammunition to achieve POM/budgeting goals.

(2) User interface will be provided through a master menu selection which provides a set of hierarchical screens to lead user through the input process and transform data into information portrayed on output screens.

(3) The Data Base Management System (DBMS) will manage, store, and retrieve all data used by the system.

(4) Report generation facilities will provide the capability to present understandable tabular data in a formatted report.

(5) Graphics facility will be interactive, flexible, comprehensive, and maintainable. Displays will include stacked and grouped bar charts and pie charts.

(6) Telecommunications and networking capabilities of the system will electronically link the PJSM system to other systems; i.e., RDAISA data base, DSACS system, and the AEMS system.

(7) The 'what if' capability will compute impacts of changes to existing conventional ammunition programs and budgets based on varied options, and quickly evaluate effects of the changes.

2.2 System Organization. Figure 2.2-1, titled Current System Plant Job Scheduling Model, delineates the system structure. A list of programs and files documenting the system is included in Annex A. The interrelationship between the structure chart and programs and files follows:

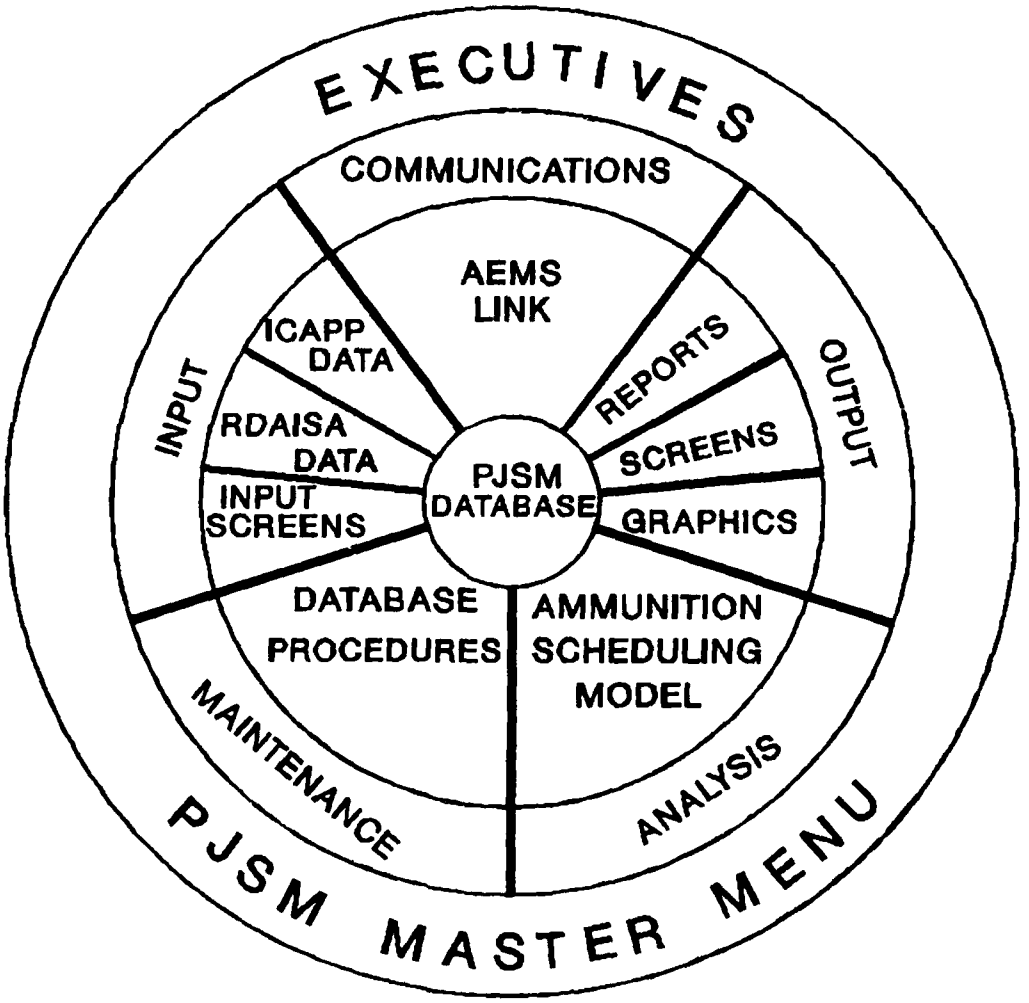
a. The PJSM data base includes the table names and descriptions displayed below. Detailed data are presented in Annex B.

DATA BASE TABLES

Table Name	Description
REVTAB	Revision Table
ITEM	Cross Reference Table
SERVICE	Service Table
PACKAGE	Package Table
FAMILY	Family Table
PLANT	Plant Table
STAFF	Staff Table
LINE	Line Table
GOALS	Goals Table
PFSTAB	PDIP Funding Sequence Table
ICT	Item Component Table
PRODT	Production Table
PROJECT	Project Table
PRODT_FY	Production Change Table
ITEM_DATA	Item Data Table
RAMP_ITEM	RAMP Years Item Table
RAMP_PROD	RAMP Years Production Table
REQTS_ARMY	Army Requirements Table
REQTS_OTHER	Other Requirements Table
TREQ	Total Requirements Table
REASON_CODE	Reason Code Table
RESULT1	Army Requirements Output Table
RESULT2	Other Requirements Output Table
RESULT3	Production Output Table
RESULT4	Item Output Table
RESULT5	Summary by Package and Family
RESULT6	Summary by Plant and Line

CURRENT SYSTEM
PLANT JOB SCHEDULING MODEL

DATA CENTERED - MULTI-LEVEL , INTERACTIVE



COMPLICATED STRUCTURE - EASY TO USE

Figure 2.2-1. Current System Structure of the Plant Job Scheduling Model.

b. The PJSM Master Menu is available through a utility program by that name.

c. Seventeen input screens are generated by input programs. The RDAISA and ICAPP programs provide a means to automatically load requirements and unit prices from other established data bases.

d. A communications link is available through a file titled Data Extract for AEMS.

e. Three input report generators provide a detailed review of all input data in the data base.

f. Four screens are available to interactively review the results of a study run.

g. Graphics output reports emanate from eight graphics programs.

h. Analyses are performed on the results obtained with the Ammunition Scheduling and Post Processor Models.

i. Systems maintenance is conducted through utilization of interactive utility programs.

j. Sample reports are contained in Appendix B of the Functional Description, PJSM System, ADSM 18-L62-LAT-ZZZ-FD-ATA, (reference 1.2a).

2.3 Security. The PJSM system displays information that is unclassified.

SECTION 3. ENVIRONMENT

3.1 Equipment Environment. A successful implementation of the PJSM system is currently operating on the AMCCOM PRIME 9955 Mod 2 minicomputer. It is anticipated that this equipment will allow for future enhancements.

3.2 Support Software. The operating software used on the PRIME 9955 Mod 2 is PRIMOS Release 20.0.4.

3.3 Data Base. The data base used is ORACLE, which is a vendor (ORACLE CORP) supported Relational Data Base Management System.

3.3.1 General Characteristics. The general characteristics of the ORACLE data base are:

a. All programs in the PJSM utilize the data base.

b. All data are considered dynamic and can be updated at random intervals.

c. The data base is housed on the PRIME 9955 Mod 2 minicomputer.

d. There are no limitations on the use of this data base by the programs in the system.

3.3.2 Organization and Detailed Descriptions. Section 2 lists the 27 data base tables that are in the PJSM system. The first 19 tables are input tables; table 20 is an intermediate table which summarizes the requirements; table 21 is an information table; and tables 22 - 27 are output tables. The composition of each table is given in Annex B. Annex C is a narrative description of the decision flow process for the production scheduling model. File formats may be found in Annex D and the decision flow process for the post processor model is described narratively in Annex E.

SECTION 4. SYSTEM MAINTENANCE

4.1 Conventions. Conventions commonly used for the Plant Job Scheduling Model (PJSM) System are listed below. Those unique to a program are described in Section 5.

a. The use of FORTRAN 77.

b. Host variables received directly from ORACLE tables end with a '$' character to help differentiate them from similarily named ORACLE columns.

4.2 Verification Procedures. Verification to check the performance of the PJSM system is usually done by a manual inspection, often using the ORACLE UFI. UFI is a program written by the ORACLE Corporation for accessing an ORACLE data base in an ad hoc manner. All of the SQL statements can be issued in UFI and there are a number of additional UFI commands that can be used for formatting UFI output.

4.3 Error Conditions. Errors are printed in a COMO or output file.

4.4 Maintenance Programs. The CPL programs used to maintain the system are listed below. These programs are described in Section 5.

PROGRAM	LOCATION	PURPOSE
JSM1.CPL	SYSSA>SAXPGM>JSM1.CPL	JSM Master Menu
ARCHIVE.CPL	SYSSA>SAXPGM>#PGM	Creates backup of JSM data tables
VALID.CPL	SYSSA>SAXPGM>VALID.CPL	External program for access by JSM1.CPL and VALID.CPL
BATCH CPL SUBPGMS	SYSSA>SAXPGM	Batch CPL subprograms interfacing JSM1.CPL
PJSM_MAINT.CPL	SYSSA>SAXPGM>#DBA> PJSM_MAINT.CPL	JSM system-level monitoring utilities
BIND32IX.CPL	SYSSA>SAXPGM>#PS> BIND32IX.CPL	Precompiles, compiles, and loads F77 programs

4.5 Maintenance Procedures. The PJSM_MAINT.CPL program functions as the administrator of the data base and is located in SYSSA>SAXPGM>#DBA>PJSM_MAINT.CPL.

SECTION 5. PROGRAM MAINTENANCE PROCEDURES

5.1 Program. PJSM Master Menu Program (JSM1.CPL)

5.1.1 Program Description. The PJSM Master Menu Program is written in CPL (PRIME's Command Processing Language) interfacing it with the PRIME ORACLE Data Base Management System using SQL.

a. Identification - The source code for the PJSM Master Menu Program program is stored in a file called JSM1.CPL in the directory SAXPGM.

b. Functions - This program is used for two purposes:

(1) Allow users access to the PJSM data on the PRIME ORACLE Data Base Management System by means of UFI and IAP.

(2) Allow users access to various FORTRAN programs which run in batch mode by means of a job statement.

c. Input - Input is done by interactive prompting from menus.

d. Processing of Subroutines -

(1) Subroutine: Main

(a) Description: Calls the JSM Master Menu.

(b) Screen:

JSM MASTER MENU

OPTION NO.	FUNCTION
1	Set up or modify study parameters
2	Develop or alter the ammunition program
3	Review or update input data
4	Review output results
5	Review graphics outputs
6	Other data base procedures
7	Interface with AEMS
8	Utility Menu
9	Select printer destination
10	Check job status
X	Logoff from PRIME

Printer destination is currently %DISK%

(c) Subroutines accessed:

(1) Set_Term_Type

(2) Enter_PJSM_Userid

(3) Main_Opt_01

(4) Main_Opt_02

(5) Main_Opt_03

(6) Main_Opt_04

(7) Main_Opt_05

(8) Main_Opt_06

(9) Printer_Dest_Menu

(10) Utility_Menu

NOTE: Users from AMSMC-AP or programmers have access to option '50' which will allow the user access to the PRIMOS OK-level.

(2) Subroutine: Main _Opt_01

(a) Description: Calls the Review or Update Guidelines Menu.

(b) Screen:

REVIEW OR UPDATE GUIDELINES

OPTION NO.	FUNCTION
1	Guidelines
2	PDIP funding sequence
R	Return to previous menu
X	Logoff

(c) Subroutines accessed:

(1) Sub_1.1

(2) Sub_1.2

(3) Subroutine: Sub_1.1

(a) Description: Calls the Guidelines Screen.

(b) ORACLE Filename: GUID_1.FRM

(c) Location: SAXPGM>*INP

(d) Execution Mode: View/Update ORACLE Screen

(4) Subroutine: Sub_1.2

(a) Description: Calls the PDIP Funding Sequence Screen.

(b) ORACLE Filename: PACK3.FRM

(c) Location: SAXPGM>*INP

(d) Execution Mode: View/Update ORACLE Screen

(5) Subroutine: Main_Opt_02

(a) Description: Calls the Ammunition Program Menu.

(b) Screen:

AMMUNITION PROGRAM MENU

OPTION NO.	FUNCTION
1	Run Production Scheduling program
2	Screen for review or update of output results
3	Run program to modify output results
R	Return to previous menu
X	Logoff

(c) Subroutines accessed:

(1) Sub_2.1

(2) Sub_2.2

(3) Sub_2.3

(6) Subroutine: Sub_2.1

(a) Description: Calls the Run Production Scheduling Program.

(b) Input Variables: USER, PSWD, RCN, TREQ.FLAG, PS.FLAG, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: PS.CPL, PS.FOR.

(e) Location: SAXPGM>$PS.

(f) Execution Mode: Batch processing.

(7) Subroutine: Sub_2.2

(a) Description: Calls the screen for review or update of output results.

(b) ORACLE Filename: RSLT_CHG.FRM.

(c) Location: SAXPGM>$INP.

(d) Execution Mode: View/update ORACLE screen.

(8) Subroutine: Sub_2.3

(a) Description: Calls the run program to modify Output Results Program.

(b) Input Variables: USER, PSWD, RCN, RES_CHANGE, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: JSM.RERUN.CPL, JSM.RERUN.FOR

(e) Location: SAXPGM>$PGM.

(f) Execution Mode: Batch processing.

(9) Subroutine: Main_Opt_03

(a) Description: Calls the Review or Update Input Data Menu.

(b) Screen:

REVIEW OR UPDATE INPUT DATA

OPTION NO.	FUNCTION
1	Screens for review or update
2	Generate reports
3	Spool reports
4	Update requirements with new RDAISA data
5	Update unit prices and other customer requirements with new ICAPP data
R	Return to Main Menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_3.1

(2) Sub_3.2

(3) Sub_3.3 (Not yet available)

(4) Sub_3.4

(5) Sub_3.5

(10) Subroutine: Sub_3.1

(a) Description: Calls the IAP screens for review and/or update to Input Data Menu.

(b) Screen:

SCREENS FOR REVIEW AND/OR UPDATE TO INPUT DATA

OPTION NO.	FUNCTION
1	Item descriptors
2	Item planning parameters
3	Item production capacity/staffing
4	General plant data
5	Primary component information

OPTION NO.	FUNCTION
6	Yearly consumption requirements
7	Production project specifications
8	Army requirements stratification
9	Other customer requirements-orders
10	Secondary component information
11	Line descriptors
12	Update or display Army requirements
13	Update or display Other Service requirements
14	DODIC/project cross-reference
15	RAMP item
16	RAMP production
R	Return to previous menu
X	Logoff from PRIME

(c) ORACLE Filenames:

JSM_1.FRM	JSM_2.FRM	JSM_3.FRM	JSM_4.FRM	JSM_5.FRM
JSM_6.FRM	JSM_7.FRM	JSM_8.FRM	JSM_9.FRM	JSM_10.FRM
JSM_11.FRM	JSM_12.FRM	JSM_13.FRM	JSM_14.FRM	JSM_15.FRM
JSM_16.FRM				

(d) Location: SAXPGM>$INP

(11) Subroutine: Sub_3.2

(a) Description: Calls the reports for Input Data Menu.

(b) Screen:

REPORTS FOR INPUT DATA

OPTION NO.	FUNCTION
1	Reports for input data by item
2	Reports for input data by table
3	Reports for input data by revision
R	Returns to previous menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_3.2.1

(2) Sub_3.2.2

(3) Sub_3.3.3 (Not yet available)

(12) Subroutine: Sub_3.2.1

(a) Description: Calls the reports for Input Data by Item program.

(b) Input Variables: USER, PSWD, RCN1, RCN2, IID1, IID2, PID1, PID2, DISK.

(c) Source Lanuages: CPL, FORTRAN.

(d) ORACLE Filename: IDP.CPL, IIDR.FOR, PDIR.FOR.

(e) Location: SAXPGM>$PGM.

(f) Execution Mode: Batch processing.

(13) Subroutine: Sub_3.2.2

(a) Description: Calls the Create Input Data Report by Table Name Menu. This subroutine, given the Table Name, will call the LISTT external program.

(b) Screen:

CREATE INPUT DATA REPORT BY TABLE NAME

Available ORACLE tables are:

FAMILY	PACKAGE	RAMP_ITEM	RESULT2	SERVICE
GOALS	PFSTAB	RAMP_PROD	RESULT3	STAFF
ICT	PLANT	REASON_CODE	RESULT4	TREQ
ITEM	PRODT	REQTS_ARMY	RESULT5	
ITEM_DATA	PROD_FY	REQTS_OTHER	RESULT6	
LINE	PROJECT	RESULT1	REVTAB	

(c) Input Variables: COLUMN, TABLE_NAME, USER, PSWD, RCN, UFD, DISK.

(d) Source Languages: CPL.

(e) ORACLE Filename: LISTT.CPL.

(f) Location: SAXPGM>*PGM.

(g) Execution Mode: Batch processing.

(14) Subroutine: Sub_3.4

(a) Description: Calls the Update Requirements with New RDAISA Data.

(b) Input Variables: USER, PSWD, FY, NEW, FILE1, FILE2, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: PKG.CPL, PKG.FOR.

(e) Location: SAXPGM>*PGM.

(f) Execution Mode: Batch processing.

(15) Subroutine: Sub_3.5

(a) Description: Calls the update unit price and other customer requirements.

(b) Input Variables: USER, PSWD, FY, LOC, YRS, ICAPP.FLAG, FILE1, FILE2, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: ICAPP.CPL, ICAPP.FOR.

(e) Location: SAXPGM>*PGM.

(f) Execution Mode: Batch processing.

(16) Subroutine: Main_Opt_04

(a) Description: Calls the Review Output Data Menu.

(b) Screen:

REVIEW OUTPUT DATA

OPTION NO.	FUNCTION
1	Screens for review output data
2	Generate reports
3	Spool reports
R	Return to previous menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_4.1

(2) Sub_4.2

(3) Sub_4.3

(17) Subroutine: Sub_4.1

(a) Description: Calls the IAP screens for reviewing output data menu.

(b) Screen:

SCREENS FOR REVIEWING OUTPUT DATA

OPTION NO.	FUNCTION
1	Army requirement allocation
2	Other customer requirement allocation
3	Production summary
4	Item summary
R	Return to previous menu
X	Logoff from PRIME

(c) ORACLE Filename: RESULT1.FRM, RESULT2.FRM, RESULT3.FRM, RESULT4.FRM.

(d) Location: SAXPGM>*INP.

(e) Execution Mode: View ORACLE screen.

(18) Subroutine: Sub_4.2

(a) Description: Calls the Generate Output Reports Menu. This subroutine calls the external program NRPT for the selection number chosen.

(b) Screen:

GENERATE OUTPUT REPORTS

OPTION NO.	FUNCTION
1	ARMY P-1 report
2	Workload summary report by item
3	Plant workload utilization
4	Plant workload summary
5	Item analysis worksheet
6	PDIP summary report
7	Comparison report
8	Items not acheiving 100%
9	Readiness factor
10	Other service buys by item - exception report
11	PDIP package structure by item
12	PDIP cost report
13	Additional workload required to reach 90% baseline
14	Secondary item status report
15	Summary of package requirements/filled
16	Conventional ammunition acquisition plan
R	Return to previous menu
X	Logoff from PRIME

(c) Input Variables: USER, PSWD, RPT, RCN1, RCN2, OPT, DISK.

(d) Source Languages: CPL, FORTRAN, RPT.

(e) ORACLE Filename: NRPT.CPL, RPT'#1'.FOR, RPT'#1'.RPT, where #1 is the menu option number for reports 1 - 16.

(f) Location: SAXPGM>#RPT.

(g) Execution Mode: Batch processing.

(19) Subroutine: Sub_4.3

(a) Description: Calls the Spool Output Reports Menu. This subroutine spools the external file RPT.*.RCN for the selection number chosen. These files are located in SAXJSM>*OUT.

(b) Screen:

SPOOL OUTPUT REPORTS

OPTION NO.	FUNCTION
1	ARMY P-1 report
2	Workload summary report by item
3	Plant workload utilization
4	Plant workload summary
5	Item analysis worksheet
6	PDIP summary report
7	Comparison report
8	Items not acheiving 100%
9	Readiness factor
10	Other service buys by item - exception report
11	PDIP package structure by item
12	PDIP cost report
13	Additional workload required to reach 90% baseline
14	Secondary item status report
15	Summary of package requirements/filled
16	Conventional ammunition acquisition plan
R	Return to previous menu
X	Logoff from PRIME

(c) Input Variables: USER, PSWD, RPT, RCN1, RCN2, OPT, DISK.

(d) ORACLE Filename: RPT'*1'.RCN'*2', where *1 is the menu option number for reports 1 - 6, 8 - 16 and *2 is the RCN or RPT7.'*2'_VS_'*3', where *2 is the RCN for report 7 and *3 is the alternative RCN.

(e) Location: SAXJSM>*OUT.

(f) Execution Mode: Spooling text file.

(20) Subroutine: Main_Opt_05

(a) Description: Calls the Graphs Menu.

(b) Screen:

GRAPHS

OPTION NO.	FUNCTION
1	Financial charts
2	Workload charts
R	Return to previous menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_5.1 (Not yet available)

(2) Sub_5.2 (Not yet available)

(21) Subroutine: Main_Opt_06

(a) Description: Calls the Other Data Base Procedures Menu.

(b) Screen:

OTHER DATA BASE PROCEDURES

OPTION NO.	FUNCTION
1	Change temporary DODAC to new DODAC
2	Delete DODIC from the data base
3	Run data base integrity check
4	Copy revisions from one study to next
5	Make revisions a permanent part of baseline
6	Archive data base
7	Delete a study run from the data base
8	Delete output reports for a study run
9	Dump input tables for a study run
10	Run alternate priority program
R	Return to previous menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_6.1

(2) Sub_6.2

(3) Sub_6.3

(4) Sub_6.4

(5) Sub_6.5

(6) Sub_6.6

(7) Sub_6.7

(8) Sub_6.8

(9) Sub_6.9

(10) Sub_6.10

(22) Subroutine: Sub_6.1

(a) Description: Changes temporary DODAC to new DODAC. This program accesses Oracle to modify the following tables: ICT, ITEM, PRODT, PRODT_FY, ITEM_DATA, RAMP_ITEM, RAMP_PROD, REQTS_ARMY, REQTS_OTHER, TREQ, RESULT1, RESULT2, RESULT3, RESULT4.

(b) Input Variables: OLD_FSC, OLD_DODIC, NEW_FSC, NEW_DODIC.

(b) Execution Mode: On-line ORACLE processing.

(23) Subroutine: Sub_6.2

(a) Description: Deletes a DODIC from the data base. All information for that DODIC is stored in a file named 'DODIC'.LIS in SAXPGM>#DEL. After that, any reference to the PJSM tables to that DODIC is removed from the data base.

(b) Input Variables: Fsc, DODIC.

(c) ORACLE Filename: 'DODIC'.LIS, where DODIC is the name of a DODIC which was removed from the data base.

(d) Location: SAXPGM>#DEL.

(e) Execution Mode: On-line ORACLE processing.

(24) Subroutine: Sub_6.3

(a) Description: Runs the Data Base Integrity Check Program.

(b) Input Variables: USER, PSWD, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: BASECHK.CPL, BASECHK.FOR

(e) Location: SAXPGM>*PGM.

(f) Execution Mode: Batch processing.

(25) Subroutine: Sub_6.4

(a) Description: Copy revisions from one study to next. This program copies information from a given RCN to another RCN for one/all of the following tables: GOALS, ITEM_DATA, PFSTAB, PRODT_FY, REQTS_ARMY, REQTS_OTHER.

(b) Input Variables: USER, PSWD, TNAME, OLD_RCN, NEW_RCN, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: CREV.CPL, CREV.FOR

(e) Location: SAXPGM>*PGM.

(f) Execution Mode: On-line ORACLE processing and batch processing.

(26) Subroutine: Sub_6.5

(a) Description: Calls the Run Extract Program. This program copies information from a given RCN to the baseline, RCN = 0, for one/or all of the following tables: GOALS, ITEM_DATA, PFSTAB, PRODT_FY, REQTS_ARMY, REQTS_OTHER.

(b) Input Variables: USER, PSWD, TNAME, OLD_RCN, DISK.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: CREV.CPL, CREV.FOR.

(e) Location: SAXPGM>*PGM.

(f) Execution Mode: Batch processing.

(27) Subroutine: Sub_6.6

(a) Description: Archive Data Base. This program takes the data from a table and makes a backup copy of it under a new name for a PJSM user. This program also makes a backup copy of all the PJSM tables, so that it may be captured by the daily tape backup procedure done by the computer operators.

(b) Tables operated on: REVTAB, ITEM, SERVICE, FAMILY, PLANT, STAFF, LINE, GOALS, PFSTAB, ICT, PRODT, PROJECT, PRODT_FY, ITEM_DATA, RAMP_ITEM, RAMP_PROD, REQTS_ARMY, REQTS_OTHER, TREQ, PACKAGE; and all or none of RESULT1, RESULT2, RESULT3, RESULT4, RESULT5, RESULT6.

(c) Input Variables: USER, PSWD, D_FLAG, R_FLAG, RCN, TNAME, DISK, SUB_UFD.

(d) Source Languages: CPL.

(e) ORACLE Filename: ARCHIVE.CPL.

(f) Execution Mode: Batch processing.

(g) Data Filename: 'Table_Name1'_SAVE.DMP or 'Table_Name2'_RCN#.DMP, where Table_Name1 is a non-RESULT table, Table_Name2 is a RESULT table, and # is a RCN number.

(h) Program Location: SAXPGM>$PGM.

(i) Data Destination: SAXJSM>$DMP.[today's date] for PJSM user SAXPGM>$DAILY_BKUP for daily tape backup.

NOTE: This program is to be modified only by the Data Base Administrator. An RCN = 666 returns user to the previous menu after an invalid response.

(28) Subroutine: Sub_6.7

(a) Description: Delete a study run from the data base. This program gives the user the ability to delete from ORACLE all data for a given RCN, all the RESULT tables (RESULT 1 - 6), or from any of these tables: REVTAB, ITEM, SERVICE, FAMILY, PLANT, STAFF, LINE, GOALS, PFSTAB, ICT, PRODT, PROJECT, PRODT_FY, ITEM_DATA, RAMP_ITEM, RAMP_PROD, REQTS_ARMY, REQTS_OTHER, TREQ, PACKAGE; and all or none of RESULT1, RESULT2, RESULT3, RESULT4, RESULT5, RESULT6.

(b) Input Variables: RCN, R_FLAG, I_FLAG, TNAME.

(c) Source Languages: UFI.

(d) Execution Mode: On-line ORACLE processing.

(29) Subroutine: Sub_6.8

(a) Description: Delete output reports for a study run. This program deletes all the reports for a given study (RCN) for all of the RESULT tables.

(b) Input Variables: RCN.

(c) Source Languages: PRIMOS.

(d) ORACLE Filename: P?T1.RCN*, RPT2.RCN*, RPT3.RCN*, RPT4.RCN*, RPT5.RCN*, RPT6.RCN*.

(e) Location: SAXPGM>*OUT.

(f) Execution Mode: On-line PRIMOS processing.

(30) Subroutine: Sub_6.9

(a) Description: Dump input tables for a study run. A copy of the input tables (REVTAB, STAFF, GOALS, PFSTAB, PRODT_FY, ITEM_DATA, modified REQTS_ARMY, REQTS_OTHER) is created, spooled, and then deleted.

(b) Input Variables: RCN.

(c) Source Languages: UFI.

(d) ORACLE Filename: DMP.RCN'RCN*'

(e) Location: SAXPGM>*OUT.

(f) Execution Mode: On-line ORACLE processing.

(31) Subroutine: Sub_6.10

(a) Description: Run Alternate Priority program.

(b) Input Variables: USER, PSWD.

(c) Source Languages: CPL, FORTRAN.

(d) ORACLE Filename: APS.CPL, APS.FOR.

(e) Location: SAXPGM>#PGM.

(f) Execution Mode: Batch processing.

(32) Subroutine: Main_Opt_07

(a) Description: Calls the AEMS Menu. This procedure spools reports (SSNDODAC.DOC, DATACOMP.DAT, or SSNDELTA.TXT) in UFD DCSRDA or calls Subroutine Sub_7.2.

(b) Screen:

AEMS MENU

OPTION NO.	FUNCTION
1	Spool DCSRDA's SSN/DODAC cross reference list
2	Run extract program
3	Spool output of the extract program
4	Spool DCSRDA's comparison text
R	Return to previous menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Sub_7.1

(2) Sub_7.2

(3) Sub_7.3

(4) Sub_7.4

(33) Subroutine: Sub_7.1

(a) Description: Calls the Spool DCSRDA's SSN/DODAC cross reference list.

(b) Program Input Variables: DISK.

(c) ORACLE Filename: SSNDODAC.DOC.

(d) Location: DCSRDA.

(e) Execution Mode: Spooling text file.

(34) Subroutine: Sub_7.2

(a) Description: Calls the Run Extract Program.

(b) Input Variables: USER, PSWD, BFY, NFY, DISK.

(c) Source Languages: CPL, FORTRAN

(d) ORACLE Filename: AEMS.CPL, AEMS.FOR.

(e) Location: SAXPGM>#PGM.

(35) Subroutine: Sub_7.3

(a) Description: Calls the spool output of the extract program.

(b) Input Variables: DISK.

(c) ORACLE Filename: DATACOMP.DAT.

(d) Location: DCSRDA.

(e) Execution Mode: Spooling text file.

(36) Subroutine: Sub_7.4

(a) Description: Calls the Spool DCSRDA's Comparison Text Program.

(b) Input Variables: DISK.

(c) ORACLE Filename: SSNDELTA.TXT.

(d) Location: DCSRDA.

(e) Execution Mode: Spooling text file.

(37) Subroutine: Utility_Menu

(a) Description: Calls the Utility Menu.

(b) Screen:

JOB SCHEDULING MODEL UTILITY MENU

OPTION NO.	FUNCTION
1	UFI Utility
2	IAP screens
3	Output screens
4	Precompile/run F77
5	PRIMOS commands
6	Reset terminal type
7	Full screen editor
R	Return to Main Menu
X	Logoff from PRIME

(c) Subroutines accessed:

(1) Run_UFI

(2) IAP_Screens

(3) Precomp_Run

(4) PRIMOS_Commands

(5) Reset_Terminal_Type

(6) Full_Screen

(38) Subroutine: Main_Opt_09

(a) Description: Calls the List of Available Printer Locations Menu. This subroutine initializes the variable disk so it may be used to establish the output destination.

(b) Screen:

LIST OF AVAILABLE PRINTER LOCATIONS

SYSSA	SYS104	SYS110	SYSSEQ	SYSSEN
SYS056	SYS090	SYS103	SYS062	SYS390
SYS103	SYS062	SYS390	SYSCAP	SYS350
SYSDP	SYS060	SYSMS	SYSDDN	SYSENG

(c) Execution Mode: On-line PRIMOS CPL routine.

(39) Subroutine: Main_Opt_10

(a) Description: Allows user to check the status of any PRIME batch jobs that he/she has submitted.

(b) Variables Accessed: JOBID.

(c) Execution Mode: On-line PRIMOS CPL routine.

e. Processing of Programs -

(1) Program: Run_UFI

(a) Description: This routine allows user to call up the ORACLE command UFI.

(b) Source Languages: ORACLE.

(c) Execution Mode: On-line PRIME ORACLE routine.

(2) Program: IAP_Screens

(a) Description: This routine allows user to call up any of the screens in SAXPGM>*INP using ORACLE's IAP.

(b) Input Variables: SCREEN.

(c) Source Languages: ORACLE.

(d) Location: SAXPGM>*INP.

(e) Execution Mode: On-line PRIME ORACLE routine.

(3) Program: Precomp_Run

(a) Description: This routine allows user to pre-compile a FORTRAN/ORACLE program.

(b) Input Variables: FILE, USER, PSWD

(c) Source Languages: CPL.

(d) ORACLE Filename: BIND32IX.CPL.

(e) Location: SAXPGM>*PS.

(f) Execution Mode: On-line PRIMOS CPL routine.

(4) Program: PRIMOS_Commands

(a) Description: This routine allows user access to PRIMOS and then back to the JSM Menu system if a blank line is returned.

(b) Execution Mode: On-line PRIMOS CPL routine.

(5) Program: Full_Screen

(a) Description: This routine allows user access to AEDIT, a full-screen editor available on PRIME.

(b) Execution Mode: On-line PRIMOS CPL routine.

(6) Program: Break_Handler

(a) Description: The default definition of the break key has been disabled. This was done so that the PJSM user cannot return to PRIMOS when the break key is pressed. Everytime the break key is pressed, a message will appear on the screen indicating that the break key has been disallowed. If the user depresses the break key a third time he/she is automatically logged off of PRIME. This logged out information is stored in the directory SAXPGM>#DBA>#LOGIN.

(b) Variables Accessed: Break_Count.

(c) Execution Mode: On-line PRIMOS CPL routine.

NOTE: This routine is accessed as an On-unit on every subroutine in this program.

(7) Program: Error_Handler

(a) Description: This program displays a message on the screen to call AMSMC-AP when an error condition would occur. These errors might occur if there is a system-fault or if the PJSM has ran out of disk space.

(b) Execution Mode: On-line PRIMOS CPL routine.

NOTE: This routine is accessed as an On-unit on every subroutine in this program.

(8) Program: Set_User

(a) Description: This routine queries the user for his/her PJSM userid. This is stored in a variable called 'USER'. The variables 'USER' and 'PSWD' are later sent to a CPL program called 'VALID.CPL' located at SAXPGM in order to verify if this user has a valid ORACLE account. See program 'ENTER_PJSM_USERID'.

(b) Variables Accessed: User.

(c) Execution Mode: On-line PRIMOS CPL routine.

(9) Program: Set_PSWD

(a) Description: This routine queries the user for his/her PJSM password. This is stored in a variable called 'PSWD'. The variables 'USER' and 'PSWD' are later sent to a CPL program called 'VALID.CPL' located at SAXPGM in order to verify if this user has a valid ORACLE account. See program 'ENTER_PJSM_USERID'.

(b) Variables Accessed: PSWD.

(c) Execution Mode: On-line PRIMOS CPL routine.

(10) Program: Set_Term_Type

(a) Description: This routine queries the user for the type of terminal he/she is using. This is stored in a variable called 'TERM_TYPE'. This information is needed by PRIME ORACLE's IAF so that the PJSM data-entry and verification screens can operate for different terminal types.

(b) Variables Accessed: Term_Type.

(c) Execution Mode: On-line PRIMOS CPL routine.

(d) Valid Terminal Types:

HDS	Human Design Systems terminal emulator
VT100	Dec VT100 terminal emulator
VT100_UL	Alternate Dec VT100 terminal emulator
PST100	PRIME PST100 terminal emulator
PT200	PRIME PT200 terminal emulator
PT45	PRIME PT45 terminal emulator
V500	Visual 500 terminal emulator/Dec VT52 terminal emulator
PC-ORACLE	PC communicating with ORACLE ORALINK communications package
XTALK	PC communicating with Crosstalk communications package

NOTE: If a user enters 'HELP' or 'H', a list of allowable terminal types would appear on the terminal screen.

(11) Program: Enter_Pjsm_Userid

(a) Description: This routine calls up an external program which checks if he/she is using a valid ORACLE Userid/Password. The external program is SAXPGM>VALID.CPL.

(b) Variables Accessed:

(1) Term_Type

(2) User

(3) PSWD

(4) Sub_UFD

(c) Execution Mode: On-line PRIMOS CPL routine.

(12) Program: Pause

(a) Description: Displays message on terminal when a job has been submitted to the batch queue.

(b) Program referred: Sub_2.1, Sub_2.3, Sub_3.2.1, Sub_4.2, Sub_6.3, Sub_6.6, Sub_6.10, Sub_6.11, Sub_7.2.

(c) Execution Mode: On-line PRIMOS CPL routine.

(13) Program: Pause1

(a) Description: This program enters a programmed loop which suspends any useful programming for approximatedly two seconds. This little utility is used by other programs so that a PJSM user may see output on his/her screen before the screen is cleared of any information when going to the previous menu.

(b) Program referred: Sub_5.1, Sub_5.2, Main_Opt_10.

(c) Variables Accessed:

(1) Time

(2) New_Time

(d) Execution Mode: On-line PRIMOS CPL routine.

f. Output - N/A

g. Security - The PJSM Master Menu Program works with information which is unclassified.

h. Interfaces - This CPL program intefaces with the PJSM tables on PRIME ORACLE through UFI and with other CPL programs.

i. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) Data tables:

REVTAB	ITEM	SERVICE	PACKAGE	FAMILY
PLANT	STAFF	LINES	GOALS	PFSTAB
ICT	PRODT	PROJECT	PROD_FY	ITEM_DATA
RAMP_ITEM	RAMP_PROD	REQTS_ARMY	REQTS_OTHER	TREQ

(b) Result tables:

RESULT1	RESULT2	RESULT3
RESULT4	RESULT5	RESULT6

5.1.2 Conventions. N/A

5.1.3 Verification Procedures. N/A

5.1.4 Error Conditions. N/A

5.1.5 Listings. Program listing is located in <SYSSA>SAXPGM>JSM1.CPL. A hard copy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.1	3. PROGRAM NAME JSMI.CPL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.2 Program. PJSM Table Export Program (ARCHIVE.CPL)

5.2.1 Program Description. The PJSM Table Export Program is written in CPL, interfacing it with the PRIME ORACLE Data Base Management System using SQL.

a. Identification - The source code for the PJSM Table Export Program is stored in a file called ARCHIVE.CPL located in the directory SAXPGM>#PGM.

b. Functions - This program is used for two purposes:

(1) A PJSM user can export a PJSM table for a given study into a directory called SAXJSM>#DMP.[today's date], where today's date is YYMMDD.

(2) The backup of all the PJSM tables on a daily basis. This is done by exporting all the tables.

c. Input -

(1) Through JSM menu, user automatically submits the following batch information:

(a) ORACLE identification (USER).

(b) Password (PSWD).

(c) Data table flag (D_FLAG).

(d) Result table flag (R_FLAG).

(e) RCN.

(f) Tablename (TNAME).

(g) Destination for reports to be spooled (DISK).

(h) Subdirectory location where export is to be stored (SUB-UFD).

(i) Daily backup flag.

Setup for PJSM user (Batch job) - referenced by SAXPGM>#JSM1.CPL - Job SAXPGM>#PGM>ARCHIVE -HO SAXJSM>%SUB_UFD% -QUEUE Q2 -ARGS %USER% %PSWD% %D_FLAG% %R_FLAG% %RCN% %TNAME% %DISK% %SUB_UFD% N.

(2) Through the daily backup program, this program is called a SYSTEM phantom that runs at 2200 daily.

Setup for Daily Backup (Phantom) - referenced by SYSTEM>PROCEDURE-UFD>SQ.CPL -PH SAXPGM>#PGM>ARCHIVE %USER% %PSWD% Y Y 99 ALL SYSSA SAXPGM Y.

d. Processing - A copy of the table requested is made. These tables are exported and then dropped. A PJSM user is given the ability of exporting all of the RESULT tables for a given RCN and/or the ability of exporting TREQ for a given RCN.

e. Output - N/A.

f. Security - The PJSM Table Export Program works with information which is unclassified.

g. Interfaces - This CPL program interfaces with the PJSM tables on PRIME ORACLE through UFI.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(a) From data tables:

REVTAB	ITEM	SERVICE	PACKAGE	FAMILY
PLANT	STAFF	LINES	GOALS	PFSTAB
ICT	PRODT	PROJECT	PROD_FY	ITEM_DATA
RAMP_ITEM	RAMP_PROD	REQTS_ARMY	REQTS_OTHER	TREQ

(b) From result tables:

RESULT1	RESULT2	RESULT3
RESULT4	RESULT5	RESULT6

5.2.2 Conventions. For daily backup and for PJSM user access, the data tables are exported under the file name 'TABLENAME'_SAVE. For PJSM user access, TREQ and the RESULT tables are saved under the file name 'TABLENAME'_RCN'RCN_NUMBER'. For daily backup the RESULT tables are saved under the file name 'TABLENAME'_RCN99.

5.2.3 Verification Procedures. Verification of the program can be accomplished by viewing the COMO files for the following procedures:

a. For a PJSM user: SAXJSM>#DMP.[today's date]ARCHIVE.COMO.

b. For daily backup: SAXPGM>#DAILY_BKUP>ARCHIVE.COMO.

5.2.4 Error Conditions. Errors will show up in the COMO files referenced in the Verification Procedures section.

5.2.5. Listings. Program listing is located in <SYSSA>SAXPGM>$PGM>ARCHIVE.CPL. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.2	3. PROGRAM NAME ARCHIVE.CPL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.3 Program. ORACLE Precompile, Compile, Link, and Execute Program (BIND32IX.CPL)

5.3.1 Program Description. BIND32IX.CPL is written in CPL interfacing it with the PRIME's ORACLE Data Base Management System using SQL.

a. Identification - The source code for the ORACLE Precompile, Compile, Link, and Execute Program is stored in the file GIND32IX.CPL located in the directory SAXPGM>$PS.

b. Functions - This CPL program is used for precompiling, compiling, linking, and executing FORTRAN 77 programs with embedded PRIME ORACLE coding.

c. Input -

(1) Program - Name of FORTRAN Program.

(2) User - (ORACLE userid).

(3) PSWD - (ORACLE password).

(4) Cursors - (Maximum Open ORACLE Cursors, default is 16).

d. Processing - BIND32IX.CPL is executed as a one-line command string with the following parameters (Program, User, PSWD, and Cursors). This program will precompile, compile, link, and then execute a newly created FORTRAN-ORACLE program. This program will also check if the program has been modified recently; if so, it will be precompiled, compiled, linked, and then executed. Existing programs not recently modified are just executed.

e. Output - A COMO file named 'Program'.COMO where Program is the name of the FORTRAN program. Any error messages will be listed in this file.

f. Security - BIND32IX.CPL works with information which is unclassified.

g. Interfaces - This CPL program interfaces with PRIME ORACLE through UFI.

h. Tables - N/A.

5.3.2. Conventions. N/A.

5.3.3 Verification Procedures. Error messages will show up on the terminal. A COMO file is created to save the error messages.

5.3.4 Error Conditions. N/A.

5.3.5 Listings. Program listing is located in <SYSSA>SAXPGM>#PS>BIND32IX.CPL. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.3	3. PROGRAM NAME BIND32IX.CPL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.4 Program. Change Revision (CREV.FOR)

5.4.1 Program Description. The Change Revision program is written in structured FORTRAN 77. The source code is located in a file named CREV.FOR which is input to the ORACLE precompiler that creates the CREV.F77 file. This file in turn is compiled with the F77 compiler and loaded with the BIND utility to produce the executable program CREV.RUN. The following paragraphs describe the CREV program:

a. Identification - Source code for this program is in the CREV.FOR file. Its executable equivalent is CREV.RUN.

b. Functions - The CREV program allows the user to copy data from one revision control number to another for the following tables: GOALS, ITEM.DATA, PFSTAB, PRODT.FY, STAFF, REQTS.ARMY, REQTS.OTHER.

c. Input -

(1) Interactive: User inputs ORACLE identification and password, responds to old RCN and new RCN prompts, and enters list of tables to update.

(2) Input by program:

(a) GOALS: All columns read for update.

(b) ITEM.DATA: All columns read for update.

(c) PFSTAB: All columns read for update.

(d) PRODT.FY: All columns read for update.

(e) STAFF: All columns read for update.

(f) REQTS.ARMY: All columns read for update.

(g) REQTS.OTHER: All columns read for update.

d. Processing -

(1) User input: Reads user ID, password, old RCN, new RCN, and list of tables for update.

(2) For each table listed by the user:

(a) Read rows in table where RCN = old RCN.

(b) Search table for row with row-id equal to row-id just read and RCN equal to user input new RCN.

(c) If a row is found, update all columns with data from old RCN row.

(d) If a row is not found, insert a row in the table with the appropriate data and RCN.

(e) Delete rows in table where RCN = old RCN.

(3) It should be noted that for the REQTS.ARMY and REQTS.OTHER table, the only update or inserts allowed are changes to the baseline.

(a) Read the quantity and change values from the table for items where the CHANGE column is not null.

(b) Update the table, setting the QUANTITY column equal to the quantity just read plus the change. Set the CHANGE column to null.

e. Output - Update or insert to tables.

f. Security - No unique security considerations.

g. Interfaces - This program does not require interfacing with other programs. PMO users may execute this program at will through the JSM Master Menu.

h. Tables - See Annex B.

5.4.2 Conventions. This program has been written with the convention that all local variables end with a '#' character to distinguish them from similarly named ORACLE columns.

5.4.3 Verification. Verification will be done manually using ORACLE's UFI to directly access the tables.

5.4.4 Error Conditions. Errors encountered will be documented in the COMO file produced at each execution.

5.4.5 Listings. Program listing is located in <SYSSA>SAXPGM>#PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.4	3. PROGRAM NAME CREV.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.5 Program. PJSM Data Base Maintenance Program (PJSM_MAINT.CPL)

5.5.1 Program Description. The PJSM Data Base Maintenance Program is written in CPL interfacing it with the PRIME ORACLE Data Base Management System using SQL. This program was written due to the fact that keying in the data base maintenance commands for PRIME ORACLE were very time-consuming. It was easier to write this program to include all of the known maintenance operations and for future maintenance operations.

a. Identification - The source code for the PJSM Data Base Maintenance Program program is stored in a file called PJSM_MAINT.CPL in the directory SAXPGM>$DBA.

b. Functions - This program is used for two purposes:

(1) Allow users access to the PJSM data on the PRIME ORACLE Data Base Management System by means of UFI and IAP.

(2) Allow users access to various FORTRAN programs which run in batch mode by means of a job statement.

c. Input - Done by interactive prompting from menus.

d. Processing - This is the general purpose menu-driven program to allow users access to various data base maintenance options for the PJSM:

(1) Program: Main

(a) Description: Initializes variables before calling the DBA_Master_Menu sub-routine.

(b) Variables Accessed:

(1) Break_Count

(2) TSCC

(3) Disk

(4) Table_Original = SAXPGM>$TABLES_ORIGINAL

(5) Table_Scratch = SAXPGM>$TABLES_SCRATCH

(6) UFD = SAXPGM

(7) Sub_UFD = SAXPGM>$DBA

(8) Login_Time

(9) Term_Type

(c) Subroutines accessed:

(1) Set_Term_Type

(2) Enter_PJSM_Userid

(3) Verify_If_DBA_or_PGMR

(4) Printer_Dest_Menu

(5) DBA_Master_Menu

(2) Program: DBA_Master_Menu

(a) Description: Main Menu which indicates which program a user may access given the user is the creator or an ordinary user.

(b) Variables Accessed:

(1) User /* contains PJSM userid*/

(2) OPT_NBR

(c) Screen:

JSM DATA BASE PROGRAMMERS MENU [Today's Date]

OPTION NUMBER	FUNCTION
1	Regrant access to all PJSM tables Menu
2	Call Table Maintenance Procedures Menu
3	Create Table Maintenance Reports Menu
4	Grant user authority to access PJSM
5	Revoke user authority to access PJSM
6	Rebuilt Scratch Tables Directory for PJSM
P	Select Printer Destination Menu
U	Reset PJSM Userid
S	Spool Table Maintenance Reports Menu
R	Set terminal type
X	Logoff from PRIME

(d) Subroutines accessed:

OPTION NUMBER	
1	Grant_Access_Menu
2	Maintenance_Menu
3	Create_Reports_Menu
4	View_Report_Menu
5	Spool_Report_Menu
6	Grant_User_Access_Menu
7	Revoke_User_Access_Menu
8	Rebuild_Create_Tab_Scr_Dir
D	Ods -> External Program created by ORACLE
M	JSM_Info -> External Program maintainable by DBA
P	Printer_Dest_Menu
R	Set_Term_Type
U	Enter_PJSM_Userid

NOTE: users from AMSMC-AP or programmers have access to option '50' which will allow the user access to the PRIMOS OK-level.

(3) Program: Create_Reports_Menu

(a) Description: Creates reports.

(b) Variables Accessed: RPT_OPT_NBR

(c) Screen:

JSM CREATE REPORTS MENU [Today's Date]

OPTION NUMBER	FUNCTION
1	Create Table Access Report
2	Create Column Access Report
3	Create Table Space Allocation Report
4	Create PJSM User Directory Report
5	*** All of the above ***
P	Select printer destination
R	Return to previous menu
S	Spool out report
X	Logoff from PRIME

Printer destination is currently [SYSTEM ID]

(d) Subroutines accessed:

OPTION NUMBER	
1	Sys_Grant
2	Col_Grant
3	Space_Allocation
4	User_Directory
5	
P	Printer_Dest_Menu
S	Spool_Report_Menu

NOTE: Option '50' Returns the user to the previous menu.

(4) Program: Spool_Report_Menu

(a) Description: Spool reports.

(b) Variables Accessed:

(1) SP_OPT_NBR

(2) Disk /*Destination for Routing Reports*/

(c) Screen:

JSM SPOOL LIST MENU [Today's Date]

OPTION NUMBER	FUNCTION
1	Spool Table Access Report
2	Spool Column Access Report
3	Spool Table Space Allocation Report
4	Spool PJSM User Directory Report
5	*** All of the above ***
R	Return to previous menu
P	Select printer destination
S	*** Not accessible ***
X	Logoff from PRIME

Printer destination is currently [SYSTEM ID]

(d) Reports spooled:

OPTION NUMBER	
1	Sys.Grant
2	Syscolauth.lis
3	Space.Allocation
4	PJSM_User_Directory

(e) Subroutine Accessed:

OPTION NUMBER	
P	Printer_Dest_Menu

NOTE: Option '50' returns the user to the previous menu.

(5) Program: View_Report_Menu

(a) Description: View reports

(b) Variables Accessed:

(1) VW_OPT_NBR

(2) Disk /*Destination for Routing Reports*/

(c) Screen:

JSM VIEW REPORT MENU [Today's Date]

OPTION NUMBER	FUNCTION
1	View Table Access Report
2	View Column Access Report
3	View Table Space Allocation Report
4	View PJSM User Directory Report
5	*** All of the above ***
R	Return to previous menu
P	Select printer destination
S	*** Not accessible ***
X	Logoff from PRIME

Printer destination is currently [SYSTEM ID]

(d) Reports viewed:

OPTION NUMBER	
1	Sys.Grant
2	Syscolauth.lis
3	Space.Allocation
4	PJSM_User_Directory

(e) Subroutine Accessed:

OPTION NUMBER	
P	Printer_Dest_Menu

NOTE: Option '50' returns the user to the previous menu.

(6) Program: Grant_Access_Menu

(a) Description: Executes the Grants User Access to PJSM Tables program.

(b) Variables Accessed:

(1) GRANT_OPT_NBR

(2) All_Flag /* Y -> accesses all tables
/* N -> accesses a table (User-selected)

(c) Screen:

JSM ACCESS GRANT MENU [Today's Date]

OPTION NUMBER	FUNCTION
1	Grant group table access
2	Grant unique table access
R	Return to previous menu
P	Select printer destination
S	Spool out report
X	Logoff from PRIME

Printer destination is currently [SYSTEM ID]

(d) Subroutine Accessed:

OPTION NUMBER	
1,2	Grant_User_Table_Access
P	Printer_Dest_Menu
S	Spool_Report_Menu

NOTE: Option '50' returns the user to the previous menu.

(7) Program: Maintenance_Menu

(a) Description: Calls menu to execute space compression on a PJSM Table and regrants access to that table.

(b) Variables Accessed:

(1) MAINT_OPT_NBR

(2) All_Flag /* N -> accesses a table (User-selected)

(c) Screen:

WELCOME TO THE JOB SCHEDULING MODEL (JSM) SYSTEM [Today's Date]
Maintenance Menu

OPTION NUMBER	FUNCTION
1	Rebuild JSM tables
R	Return to previous menu
P	Select printer destination
S	Spool out report
X	Logoff from PRIME

Printer destination is currently [SYSTEM ID]

(d) Subroutine Accessed:

OPTION NUMBER	
1	Rebuild_Tables_Menu, Grant_User_Table_Access
P	Printer_Dest_Menu
S	Spool_Report_Menu

NOTE: Option '50' returns the user to the previous menu.

(8) Program: Grant_User_Access_Menu

(a) Description: Grant user access to the PJSM system level views.

(b) Variables Accessed:

User1	User whose views are given access to
User	PJSM ORACLE Userid
PSWD	PJSM ORACLE Password

(c) Views Accessed:

Wells.PJSM_User_Catalog	PJSM User Information
Wells.JSM_Colauth	PJSM User Column Update Information
Wells.JSM_Tabauth	PJSM User Table Access Rights Information
Wells.JSM_Taballoc	PJSM Table Space Allocated Information

(9) Program: Revoke_User_Access_Menu

(a) Description: Revokes user access to the PJSM system level views

(b) Variables Accessed:

User1	User whose views are revoked to
User	PJSM ORACLE Userid
PSWD	PJSM ORACLE Password

(c) Views Accessed:

Wells.PJSM_User_Catalog	PJSM User Information
Wells.JSM_Colauth	PJSM User Column Update Information
Wells.JSM_Tabauth	PJSM User Table Access Rights Information
Wells.JSM_Taballoc	PJSM Table Space Allocated Information

(10) Program: Rebuild_Tables_Menu

(a) Description: Executes space compression on a PJSM table and regrants access to that table.

(b) Variables Accessed:

Table_Scratch
Table_Original
Flag_Old
Flag_New
Table_Name
Old_Tab_Exists
User
PSWD

(c) Screen:

RESUILD JSM TABLES [Today's Date]

Available ORACLE tables are:

FAMILY	PACKAGE	RAMP_ITEM	RESULT2	SERVICE
GOALS	PFSTAB	RAMP_PROD	RESULT3	STAFF
ICT	PLANT	REASON_CODE	RESULT4	TREQ
ITEM	PRODT	REQTS_ARMY	RESULT5	
ITEM_DATA	PRODT_FY	REQTS_OTHER	RESULT6	
LINE	PROJECT	RESULT1	REVTAB	

******* (Enter Q to Quit) *******

(d) ORACLE Table Definition Location:

(1) Current Table Definition from 'Table_Original' Directory.

(2) Old Table Definition from 'Table_Scratch' Directory.

(3) Current Table Index Definition from 'Table_Original' Directory.

(4) Old Table Index Definition from 'Table_Scratch' Directory.

NOTE:

Current Table Definition is as follows: 'table_name'.UFI
Current Table Index Definition is as follows: 'table_name'_INDX.UFI
Old Table Definition is as follows: 'table_name'_OLD.UFI
Old Table Index Definitionn is as follows: 'table_name'_Old_INDX.UFI

where 'table_name' is the name of a PJSM table.

(11) Program: Grant_User_Table_Access

(a) Description: Recreates the Grant Definitions for a single PJSM Table or all PJSM Tables. Access rights of select, alter, index, insert, update, or select can be given to the PJSM users.

(b) Variables Accessed:

(1) User

(2) PSWD

(3) All_Flag

(4) Table_Name

(5) Current_Table

(6) X

(c) Tables where users are given access to:

FAMILY	PACKAGE	RAMP_ITEM	RESULT2	SERVICE
GOALS	PFSTAB	RAMP_PROD	RESULT3	STAFF
ICT	PLANT	REASON_CODE	RESULT4	TREQ
ITEM	PRODT	REQTS_ARMY	RESULT5	
ITEM_DATA	PRODT_FY	REQTS_OTHER	RESULT6	
LINE	PROJECT	RESULT1	REVTAB	

(12) Program: Printer_Dest_Menu

(a) Description: Selects PRIME Computer System Destination to route printouts

(b) Variables Accessed: Disk

(c) Screen:

LIST OF AVAILABLE PRINTER LOCATIONS [Today's Date]

SYSSA	SYS104	SYS110	SYSSEQ	SYSSEN	SYS056	SYS090
SYS103	SYS062	SYS390	SYS103	SYS062	SYS390	SYSCAP
SYS350	SYSDP	SYS060	SYSMS	SYSDDN	SYSENG	IME9

(13) Program: Sys_Grant

(a) Description: Creates report indicating the rights granted for users to the PJSM tables. Rights granted are select, update, insert, delete, or alter.

(b) Variables Accessed:

(1) User

(2) PSWD

(c) Report Destination: SAXPGM>$DBA>$REPORTS>SYS.GRANT

(d) Views Accessed:

Wells.JSM_Tabauth PJSM User Table Access Rights Information

NOTE: This is a ORACLE view which is maintainable by the ORACLE Data Base Administrator.

(14) Program: Col_Grant

(a) Description: Creates report indicating the update rights granted for users to individual columns in the PJSM tables.

(b) Variables Accessed:

(1) User

(2) PSWD

(3) Report Destination:

SAXPGM>$DBA>$REPORTS>SYSCOLAUTH.LIS

(c) Views Accessed:

Wells.JSM_Colauth PJSM User Column Update Information

NOTE: This is a ORACLE View which is maintainable by the ORACLE Data Base Administrator.

(15) Program: Space_Allocation

(a) Description: Creates report indicating the space used in terms of 2048 byte blocks by the PJSM tables. An extent of 16 is set by the ORACLE Data Base Administrator as a limit for the maximum size for a table. If limit is reached, then perform the Space Compression routine in the Maintenance Screen. If this routine fails, then the DBA must enlarge the space allocated for that table, and possibly enlarge the file where this table is stored (this is a major effort).

(b) Variables Accessed:

(1) User

(2) PSWD

(c) Report Destination:

SAXPGM>$DBA>$REPORTS>SPACE.ALLOCATION

(d) Views Accessed:

Wells.JSM_Taballoc PJSM Table Space Allocated Information

NOTE: This is a ORACLE view which is maintainable by the ORACLE Data Base Administrator.

(16) Program: User_Directory

(a) Description: Creates report indicating the views, tables, and/or synonyms that a PJSM user has in his account.

(b) Variables Accessed:

(1) User

(2) PSWD

(c) Report Destination:

SAXPGM>$DBA>$REPORTS>PJSM_USER_DIRECTORY.LIS

(d) Views Accessed:

Wells.PJSM_User_Catalog PJSM User Directory Information

NOTE: This is a ORACLE view which is maintainable by the ORACLE Data Base Administrator.

(17) Program: Rebuild_Create_Tab_Scr_Dir

(a) Description: Creates the table and index definitions for the backup tables required for the daily backup process for PSJM. These definition files are stored in SAXPGM>$TABLES_ORIGINAL and SAXPGM>$TABLES_SCRATCH.

(b) Variables Accessed:

(1) Table_Original

(2) Table_Scratch

NOTE: When a PJSM Table is altered, its corresponding table and index must be altered in SAXPGM>$TABLES_ORIGINAL.

(18) Program: Break_Handler

(a) Description: The default definition of the break key has been disabled. This was done so that the PJSM user can not return to PRIMOS when the break key is pressed. Everytime the break key is pressed, a message will appear on the screen indicating that the break key has been disallowed. If the user depresses the break key a third time he/she is automatically logged off of PRIME. This logged out information is stored in the directory SAXPGM>$DBA>$LOGIN.

(b) Variables Accessed: Break_Count

(c) Execution Mode: On-line PRIMOS CPL routine

NOTE: This routine is accessed as an On-unit on every subroutine in this program.

(19) Program: Set_User

(a) Description: This routine queries the user for his/her PJSM userid.

(b) Variables Accessed: User

(20) Program: Set_PSWD

(a) Description: This routine queries the user for his/her PJSM password.

(b) Variables Accessed: PSWD

(21) Program: Set_Term_Type

(a) Description: This routine queries the user for the type of terminal he/she is using. This is stored in a variable called 'TERM_TYPE'. This information is needed by this program in order that the terminal can clear its screen properly.

(b) Variables Accessed: Term_Type

(c) Execution Mode: On-line PRIMOS CPL routine

(d) Valid Terminal Types:

HDS	Human Design Systems terminal emulator
VT100	Dec VT100 terminal emulator
VT100_UL	Alternate Dec VT100 terminal emulator
PST100	PRIME PST100 terminal emulator
PT200	PRIME PT200 terminal emulator
PT45	PRIME PT45 terminal emulator
V500	Visual 500 terminal emulator/Dec VT52 terminal emulator
PC-ORACLE	PC communicating with ORACLE ORALINK communications package
XTALK	PC communicating with Crosstalk communications package

NOTE: If a user enters 'HELP' or 'H', a list of allowable terminal types would appear on the terminal screen.

(22) Program: Enter_PJSM_Userid

(a) Description: This routine calls up an external program which checks if he/she is using a valid ORACLE Userid/Password. The external program is SAXPGM>VALID.CPL.

(b) Variables Accessed:

(1) Term_Type

(2) User

(3) PSWD

(4) Sub_UFD

(5) Flag_old

(6) Flag_new

(7) User_temp

(23) Program: Verify_If_DBA_or_PGMR

(a) Description: This routine checks if the user is a DBA or a programmer by checking a column DBA_Flag in a DBA maintainable view called PJSM_User for D (DBA) or P (Programmer). If the user is not listed as either D or P in PJSM_User, he/she will be logged out of this program.

(b) Variables Accessed:

(1) DBA_or_PGMR_List

(2) A

(3) Max_DBA_or_PGMR

(4) User

(5) PSWD

(c) Views Accessed:

Wells.PJSM_Users PJSM User Information

e. Descriptions of all PJSM ORACLE Views:

(1) WELLS.PJSM_USERS

NO.	SIZE	CSIZE	TYPE	NAME	DESCRIPTION
1	10	1	1 character	USER1	ORACLE Userid
2	10	1	1 character	PSWD1	ORACLE Password
3	20	1	1 character	DEPT	Department
4	20	1	1 character	DEPT_HEAD	Department Head
5	14	75	12 date data type	ENTRY_DATE	Date User Given PJSM Access
6	1	1	1 character	DBA_FLAG	(DBA/Programmer/User)
7	1	1	1 character	SYN	N/A
8	10	1	1 character	GRANTOR	N/A

(2) WELLS.PJSM_USERS_CATALOG

NO.	SIZE	CSIZE	TYPE	NAME	DESCRIPTION
1	10	1	1 character	USER1	ORACLE Userid
2	30	1	1 character	CREATOR	Table Creator
3	30	1	1 character	TNAME	Table Name
4	7	1	1 character	TABLETYPE	Table Type
5	22	40	2 numeric	CLUSTERID	N/A
6	22	40	2 numeric	LOGBLK	N/A
7	22	40	2 numeric	REQBLK	N/A
8	10	1	1 character	IXCOMP	N/A

(3) WELLS.JSM_TABALLOC

NO.	SIZE	CSIZE	TYPE	NAME	DESCRIPTION
1	30	1	1 character	OWNER	Table Owner
2	30	1	1 character	NAME	Table Name
3	22	40	2 numeric	D_BLKS	No. of Data Table Blocks
4	22	40	2 numeric	D_EXTS	No. of Data Table Extents
5	22	40	2 numeric	I_BLKS	No. of Index Blocks
6	22	40	2 numeric	I_EXTS	No. of Index Extents

(4) WELLS.JSM_COLAUTH

NO.	SIZE	CSIZE	TYPE	NAME	DESCRIPTION
1	30	1	1 character	GRANTOR	Table Owner
2	30	1	1 character	GRANTEE	Table User
3	30	1	1 character	TNAME	Table Name
4	30	1	1 character	COLNAME	Column Name
5	1	1	1 character	UPD	(*=Updateable Column)

f. Output - N/A

g. Security - The PJSM Data Base Maintenance Program works with information which is unclassified.

h. Interfaces - This CPL program intefaces with the PJSM tables on PRIME ORACLE through UFI and with other CPL programs.

i. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) Data table:

REVTAB	ITEM	SERVICE	PACKAGE	FAMILY
PLANT	STAFF	LINES	GOALS	PFSTAB
ICT	PRODT	PROJECT	PROD_FY	ITEM_DATA
RAMP_ITEM	RAMP_PROD	REQTS_ARMY	REQTS_OTHER	TREQ

(2) Result table:

RESULT1	RESULT2	RESULT3
RESULT4	RESULT5	RESULT6

5.5.2 Conventions. N/A

5.5.3 Verification Procedures. N/A

5.5.4 Error Conditions. N/A

5.5.5 Listings. Program listing is located in <SYSSA>SAXPGM>$DBA>PJSM_MAINT.CPL. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.5	PJSM MAINT.CPL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.6 Program. Data Base Consistency Check (BASECHK.FOR)

5.6.1 Program Description. The BASECHK.FOR program is written in structured FORTRAN 77. Source code for the program is the BASECHK.FOR file. This file is input to the ORACLE precompiler to build the BASECHK.F77 file which is compiled and then loaded with the BIND utility to produce the executable program BASECHK.RUN. The following paragraphs describe the BASECHK.FOR program:

a. Identification - Source code is located in the BASECHK.FOR file and its executable equivalent is BASECHK.RUN. Users will have access to this program only through the JSM Master Menu.

b. Functions - The BASECHK.FOR program reads virtually all input tables and checks for valid DODICs, package levels, plants, lines, checks requirements, and production data as well. This is the data base integrity checker and it is run through the JSM Master Menu as a batch job.

c. Input -

(1) Interactive: User enters ORACLE identification and password.

(2) By program:

(a) From ITEM table: Read DODIC, use, family, new materiel fielding code, sub-family.

(b) From ITEM_DATA: Read DODIC, priority, POM cost, FY, unit price.

(c) From ICT: Select DODIC and component.

(d) From PRODT: Select DODIC, plant, line, R185.

(e) From PRODT_FY: Select DODIC, plant, line, R185.

(f) From RAMP_ITEM: Select DODIC, plant, line.

(g) From RAMP_PROD: Select DODIC, plant, line.

(h) From REQTS_ARMY: Select DODIC, FY, package level, quantity.

(i) From REQTS_OTHER: Select DODIC, FY, service, unit price, quantity.

(j) From REVTAB: Select beginning FY. next FY.

(k) TREQ: Select DODIC.

(l) FAMILY: Read family codes:

(m) LINE: Read line-plant cross reference.

(n) PLANT: Read plant list.

(o) STAFF: Read plant list.

(p) SERVICE: Read list of services.

d. Processing - The program is broken into five major subroutines: DODIC_MATCH, ITEM, PRODX, REQTS, and AREQTS. Each will be covered individually, in detail, in the following paragraphs. The user responds to ORACLE identification and password prompts as in most other programs.

(1) Main: Write report header. Also include a blank line (this is accomplished with a PRIME call to TONL). Begin calling subrountines.

(2) Subroutine: DODIC-MATCH. Its function is to check the ICT, PRODT, PRODT_FY, RAMP_ITEM, RAMP_PROD, REQTS_ARMY, REQTS_OTHER, and TREQ tables for invalid DODICs. The same logic applies for all of the mentioned tables.

(a) Read the ITEM table for list of DODICs.

(b) Read the ITEM_DATA table for list of DODICs.

(c) Declare and open a cursor to return all DODICs from one of the above mentioned tables. In each case, a separate subroutine - MATCH - is called which matches the current DODIC with the list of DODICs read from the ITEM and ITEM_DATA tables. MATCH returns an item number which, if non-zero, implies the current DODIC was found. In each case, keep track of the number of DODICs not in ITEM table, and the number of DODICs not found in ITEM_DATA table. Write these to output report.

(d) For the ICT table, all components are also checked.

(3) Subroutine: ITEM. Its function is to compare the DODICs in the ITEM and ITEM_DATA tables for consistency. The ITEM_DATA table is also checked for no priority items which initiates an error; and the ITEM table is checked for items that have no requirement, are not in family 7, and are not a component. If all of the above occur, an error message is initiated.

(a) Declare and open a cursor that will read the ITEM table and return the DODIC and family.

(b) For each record returned:

(1) Search the ITEM_DATA table for the DODIC, if not found write ouput error message.

(2) Search the FAMILY table for the family code returned; if not found, write output error message.

(3) Read the Army requirements from the REQTS_ARMY table, the requirements from the REQTS_OTHER table, and the component status from the ICT table.

(4) If both requirements are zero, the item is not a component, and the item is not non-AAO, then write an error message.

(5) Keep track of the number of DODICs with each of the above conditions. Also keep a list of items that are in 5.6.1d(3)(b)(4).

(c) When finished write out list of items in 5.6.1d(3)(b)(4), and the number of DODICs not found in ITEM_DATA, with invalid family code, and total entries read.

(d) Declare and open a cursor that will return the DODIC, FY, priority, and POM cost from the ITEM_DATA table.

(e) For each entry returned:

(1) If the priority is null, write an error message.

(2) Read the ITEM table for the family and sub-family codes associated with the DODIC.

(3) If family = 7 and sub-family = 1 and the POM cost = 0, then save item in list to be output later; if the POM cost is greater than zero, write an error message. If neither of the two major conditionals occur then read the PRODT and PRODT_FY tables to see if the item exists. If it does not, then add the item to a list for output later.

(f) When finished reading ITEM_DATA table, write out the lists of DODICs saved based on the prior conditions. Also write out totals of DODICs falling into the variuos conditionals.

(4) Subroutine: PRODX. This subroutine will check the PRODT, PRODT_FY and RAMP_PROD for valid plant codes, line numbers, and DODICs.

(a) Read the LINE table for list of valid plants and lines.

(b) For each of the three tables:

(1) Declare and open an cursor that will return the DODIC, plant, and line.

(2) For each entry returned:

(a) Check line against list of valid lines; if not found write an error message.

(b) Search the PLANT, STAFF, and LINE tables for the plant. If not found in any one of these tables, write an error message.

(c) Keep a running total of the number of DODICs with invalid lines or plants.

(3) When completed, write out numbers tallied previously.

(5) Subroutine: REQTS. This subroutine will check the REQTS_OTHER table for valid service codes. It will also check REQTS_OTHER and REQTS_ARMY for inconsistent unit prices and requirements.

(a) Read SERVICE table for list of valid services.

(b) For the REQTS_OTHER table, declare and open a cursor to return the DODIC, FY, service code, unit price, and quantity.

(c) Check service code against the list of valid DODICs. If not found write an error message.

(d) If the quantity is greater than zero and the unit price is zero, continue; otherwise, search the ICT table to determine if it is a component. If it is not a component, the item must exist in the PRODT_FY or PRODT tables where line is turned on; i.e., avail>0. If neither is true, add the DODIC to a list of items to be output later.

(e) When completed with REQTS_OTHER table write out the list of items that have no requirements, the list of items which are components, have no requirements and no production data, and the number of entries read, entries with invalid service codes, entries with inconsistent unit prices, and the number of entries with inconsistent requirements.

(f) For the REQTS_ARMY table, declare and open a cursor that will return the DODIC, FY, package level, and quantity.

(g) If the quantity is greater than zero, then search ITEM_DATA table for the unit price. If the unit price is zero or the item is not found write an error message.

(h) If the quantity equals zero, then search the ICT table to determine if the item is a component. If it is not a component, then add the DODIC to a list of items to be printed out later.

(i) If the DODIC is a component, then it should have data in either PRODT or PRODT_FY with line availability greater than zero. Search these two tables for the DODIC with line availability greater than zero. If it is not found, add the DODIC to a list of items to be printed out later.

(j) Loop through all entries in the REQTS_ARMY table.

(k) Write out:

(1) List of DODICs with no requirements.

(2) List of DODICs which are components, have no requirement, and no production data.

(l) Write out:

(1) Number entries read.

(2) Number entries with inconsistent unit prices.

(3) Number entries with inconsistent requirements.

(m) End subroutine.

(6) Subroutine: AREQTS. This subroutine will check the package levels across years for consistenty.

(a) Read REVTAB for the beginning FY and number of FYs where RCN = 0.

(b) Declare and open a cursor that will return the FY, package level, and quantity from REQTS_ARMY.

(c) Read in all packages and all pertinent FYs for a particular DODIC.

(d) For each DODIC, test packages across FYs:

(1) If package 10 is greater than 0, then package 5 must be greater than zero.

(2) If package 11 is greater than 0, then package 6 must be greater than zero.

(3) If package 15 is greater than 0, then package 13 must be greater than zero.

(4) If the item is a new materiel fielding item, then packages 3-7 must be non-zero.

(5) If the item is a training item, then packages 5 and 6 must be greater than zero, and packages 7, 8, 9, 13, 14, and 17 must equal zero.

(6) If the item is a war reserve item, then packages 6, 11, 12, and 16 must be zero.

(7) If any of packages 17, 14, 9, 8, or 7 has a quantity equal to zero, then all lower package levels (in the 17, 14, 9, 8, and 7 list) must equal zero as well.

(8) If any package level has a quantity greater than zero in any FY, then it must also have quantity greater than zero in all succeeding years.

(9) The requirement for a package must not rise or fall by more than 50 percent from the first year to the last year.

(e) If any of the above fail, a descriptive error message is written and the total of each is retained.

(f) Loop through all items.

(g) When finished, write out the totals for each of the possible error conditions outlined previously.

(h) End subroutine.

(7) End program.

e. Output - BASECHK.COMO

f. Interfaces - This program requires no special interfacing with any other program. The functional user can execute the program at any time but only through the JSM Master Menu.

g. Tables - See Annex B.

5.6.2 Conventions. This program was written with the convention that local variables end with a '$' character to help differentiate them from similarly named ORACLE columns.

5.6.3 Verification Procedures. Verification must be done by the functional user using the UFI utility to manually access the data base.

5.6.4 Error Conditions. All errors encountered will be documented in the COMO file created at each execution of the program.

5.6.5 Listings. Program listing is located in <SYSSA>PGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.6	3. PROGRAM NAME BASECHK.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.7 Program. ORACLE User Validation Program (VALID.CPL)

5.7.1 Program Description. The ORACLE User Validation Program is written in CPL interfacing it with the PRIME Data Base Management System using SQL.

a. Identification - The source code for the ORACLE User Validation program is stored in a file called VALID.CPL located in the directory SAXPGM.

b. Functions - This CPL program is used as an external program for access by other CPL programs (SAXPGM>JSM1.CPL and SAXPGM>$DBA>VALID.CPL).

c. Input -

(1) Through SAXPGM>JSM1.CPL:

(a) User - ORACLE userid.

(b) PSWD - ORACLE password.

(c) TSCC - Terminal Screen Clear Code.

(d) UFD - Directory of log file; i.e., SAXJSM.

(e) Login_Time - Start Time.

(2) Through SAXPGM>$DBA>VALID.CPL.

(a) User - ORACLE userid.

(b) PSWD - ORACLE password.

(c) TSCC - Terminal Screen Clear Code.

(d) Sub-UFD - Directory of log file; i.e., SAXPGM>$DBA.

(e) Login_Time - Start Time.

d. Processing - The ORACLE User Validation Program is used as a means of checking if a user trying to gain access to ORACLE is actually a valid user. The user is given three attempts to gain access into ORACLE. If successful, the PRIME userid, ORACLE userid, time of entry, and time logged are recorded in a log file. If the user is unsuccessful, either by entering three invalid ORACLE userids or hitting the break key three times, the PRIME userid, ORACLE userid, time of entry, and time logged off are recorded in a log file after which the user is logged off of PRIME.

e. Output -

(1) USER2 - Valid ORACLE Userid.

(2) PSWD2 - Valid ORACLE Password.

f. Security - The ORACLE User Validation Program works with information which is unclassified.

g. Interfaces - This CPL program interfaces with PRIME ORACLE through UFI.

h. Tables - N/A.

5.7.2 Conventions. N/A.

5.7.3 Verification Procedures. N/A.

5.7.4 Error Conditions. N/A.

5.7.5 Listings. Program listing is located in <SYSSA>SAXPGM>VALID.CPL. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.7	3. PROGRAM NAME VALID.CPL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.8 Program. Batch CPL Subprograms Interfacing JSM1.CPL (BATCH CPL PGMS)

5.8.1 Program Description. The following is a description of programs written in Command Procedure Language (CPL) which interface with SAXPGM>JSM1.CPL.

a. Identification -

CPL PROGRAM	LOCATION (* = SAXPGM)	USAGE
AEMS	*>#PGM	Runs AEMS Program
APS	*>#PGM	Runs APS Program
BASECHK	*>#PGM	Runs BASECHK Program
CREV	*>#PGM	Runs CREV Program
ICAPP	*>#PGM	Runs ICAPP Program
IDP	*>#PGM	Runs IIDR & PIDR Programs
IWIR	*>#PGM	Runs IWIR Program
JSM.RERUN	*>#PGM	Runs JSM.RERUN Program
LISTT	*>#PGM	Creates Individual Extract Reports
PKG	*>#PGM	Runs PKG Program
PS	*>#PS	Runs TREQ, BUILD_PTR, & PS Programs
NRPT	*>#RPT	Runs RPT1 - RPT16 Programs

b. Functions - These programs were written to allow FORTRAN 77 programs interfacing with PRIME ORACLE to be called in a Batch mode from SAXPGM>JSM.CPL

c. Input - See the program documentation for the SAXPGM>JSM1.CPL.

d. Processing - A batch program is called up by SAXPGM>JSM1.CPL with its associated input parameters.

e. Output - N/A.

f. Security - These programs work with information which is unclassified.

g. Interfaces - These CPL programs interface with FORTRAN 77 programs with the same name.

h. Tables - N/A.

5.8.2 Conventions. N/A.

5.8.3 Verification Procedures. A COMO file is created within each batch program in order to check the status of the program while it is running.

5.8.4 Error Conditions. N/A.

5.8.5 Listings.

CPL PROGRAM	LOCATION
AEMS	SAXPGM>#PGM
APS	SAXPGM>#PGM
BASECHK	SAXPGM>#PGM
CREV	SAXPGM>#PGM
ICAPP	SAXPGM>#PGM
IDP	SAXPGM>#PGM
IWIR	SAXPGM>#PGM
JSM.RERUN	SAXPGM>#PGM
LISTT	SAXPGM>#PGM
PKG	SAXPGM>#PGM
PS	SAXPGM>#PS
NRPT	SAXPGM>#RPT

A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.8	3. PROGRAM NAME BATCH CPL SUBPGMS	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.9 Program. Item Descriptors Input Screen (JSM_1.INP)

5.9.1 Program Descritpion. The Item Descriptors screen is written using ORACLE Interactive Appplication Facility (IAF). An ORACLE process called Interactive Application Generator (IAG) is used to create the form necessary to produce a screen. The ORACLE Interactive Application Processor (IAP) is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 1.

a. Identification - The source code for the Item Descriptors screen is identified by a file labeled JSM_1.INP. After being compiled through ORACLE IAG, a run file is created named JSM_1.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Item Descriptors, retrieves general item information from the PJSM data base table titled ITEM. The purpose of this screen is to review or input item data and update all but the FSC and DODIC. To execute a query, a user needs to know one of five identifiers of an item. If a user does not have that information, execution of a query will result in items appearing on the screen, one at a time, in DODIC order.

c. Input -

(1) Through JSM Master Menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base. From ITEM table:

(a) DODIC --- CHARACTER (4).

(b) FSC --- CHARACTER (4).

(c) SSN --- CHARACTER (6).

(d) NSN --- CHARACTER (16).

(e) ISN --- CHARACTER (5).

(f) Nomenclature --- CHARACTER (48).

(g) Family Code --- INTEGER (1).

(h) Sub Family Code --- INTEGER (1).

(i) Whether or not an item is new materiel. Fielding (NMF) --- CHARACTER (1).

(j) Unit of Measure (UOM) --- CHARACTER (4).

(k) Use Code --- CHARACTER (1).

d. Processing -

(1) The Item Descriptors screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause, which is by DODIC then FAMILY. If a query is executed with no given information, the order in which items will be returned to the screen is by DODIC. If a particular family is the value being queried, the items will appear in DODIC order within that family.

(b) Further limitation dictates that a user can only answer with a 'THOU' or 'EACH' when asked for the unit of measure and a 'Y' or 'N' when asked whether or not the item is a new materiel fielding item.

(c) The user is unable to delete any item through this screen. To delete an item, a procedure is available on the JSM Master Menu to authorized users only.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the table being used in the Item Descriptors screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) When updating value, replace old value in ITEM table with new value as inserted to the screen, based on key identifier (DODIC.

(6) When inserting a new item (record), enter screen values into ITEM table. When inserting a new record, the following fields are mandatory:

(a) DODIC.

(b) FSC.

(c) Nomenclature.

(d) Family Code.

(e) Sub Family Code.

(f) New Materiel Fielding (NMF).

(g) Unit of Measure (UOM).

(h) Use Code.

e. Output - Filled screen (see figure 5.9.1-1).

f. Security - The Item Descriptors screen displays information which unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following table in the PJSM data base:

ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

5.9.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.9.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

Item Descriptors

DODIC: D563

FSC: 1320

SSN: E28100

NSN: 1320-00-126-7339

Nomenclature: PROJ 155MM HE ICM M483A1 W/O/FUZE

Family: 3

Sub-Family: 1

MMF: N

Unit of Measure: THOU

Use Code: W

Figure 5.9.1-1. Example of Item Descriptors Input Screen Filled with Information

5.9.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know that the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.9.5 Listings. Program listing is located in <SYSSA>SAXPGM>#INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.9	3. PROGRAM NAME JSM 1.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.10 Program. Item Planning Parameters Input Screen (JSM_2.INP)

5.10.1 Program Description. The Item Planning Parameters screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 2.

a. Identification - The source code for the Item Planning Parameters screen is identified by a file labeled JSM_2.INP. After being compiled through ORACLE IAG, a run file is created named JSM_2.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Item Planning Parameters, retrieves item information from two PJSM data base tables titled, ITEM and ITEM_DATA. The purpose of this screen is to review, input, or update specified model parameters that can be set up for individual items. The general item information that appears in block 1 at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed, either through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From ITEM_DATA table:

(1) FY --- INTEGER (2).

(2) RCN --- INTEGER (2).

(3) Unit Price (UNIT_PRICE) --- NUMBER (F12.4).

(4) Priority (PRI) --- INTEGER (3).

(5) Inventory Protect (DRAWDN) --- INTEGER (3).

(6) Readiness Buildup (BUILDUP) --- INTEGER (3).

(7) Non AAO Costs (POM_COST) --- NUMBER (F6.1).

(8) Maximum Inventory Allowed (MIA) --- INTEGER (10).

d. Processing -

(1) The Item Planning Parameters screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information. These rules include the use of a predelete and preselect statement, in addition to answering 'No' to the update field prompt.

(b) The second block, which relates to the ITEM_DATE table, also uses a default WHERE and ORDER BY clause which is by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Item Planning Parameters screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the ITEM_DATA table.

(6) Updating and inserting new information about an item can only be accomplished in block 2 of the Item Planning Parameters screen. The screen automatically checks to see if there is an active RCN. If not, a message will state that to the user. If there is an active RCN, the screen will take all of the information that can be updated/inserted for that DODIC and place the new data into the maximum RCN that is active. A user is unable to update baseline data which are stored under RCN zero. The user cannot delete information that is being retrieved from the ITEM_DATA table. When inserting a new record, the following fields are mandatory:

(a) FY.

(b) Priority (PRI).

e. Output - Filled screen (see figure 5.10.1-1).

f. Security - The Item Planning Parameters screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) ITEM_DATA - contains essential guideline information about each item for a particular RCN. If an item does not have specific goals, it will be defaulted to general rules governing all items which are found in the GOALS table.

Item Planning Parameters

FSC: 1315 DODIC: C785 SSN: E73400 NSN: 1315- ISN: 03266

Nomenclature: CTG 120MM TPCSDS-T M865

FY	RCN	Unit Price	Priority	Inventory Protect	Readiness Buildup	Non-AAO Costs	Maximum Inventory	Percent of Training
88	0	597.7144	2					
89	0	591.6899	3					
90	0	742.1899	5					
90	1	581.5966	5					
91	0	724.61	16					
91	1	576.3041	16					
92	0	716.3	16					
93	0	574.7094	16					

Figure 5.10.1-1. Example of Item Planning Parameters Input Screen Filled with Information

5.10.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.10.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.10.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.10.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.10	3. PROGRAM NAME JSM 2.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.11 Program. Item Production Capacity/Staffing Input Screen (JSM_3.INP)

5.11.1 Program Description. The Item Production Capacity/Staffing screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 3.

a. Identification - The source code for the Item Production Capacity/Staffing screen is identified by a file labled JSM_3.INP. After being compiled through ORACLE IAG, a run file is created named JSM_3.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Item Production Capacity/ Staffing, retrieves item information from two PJSM data base tables titled, ITEM and PRODT. The purpose of this screen is to review, input, or update specific production information about each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted here. These functions have to be performed either through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From PRODT table:

(1) Plant (PLANT) --- CHARACTER (2).

(2) Line (LINE) --- INTEGER (3).

(3) 1-8-5 Production Rate (R185) --- INTEGER (9).

(4) 1-8-5 Staffing (S185) --- INTEGER (4).

(5) 2-8-5 Production Rate (R285) --- INTEGER (9).

(6) 2-8-5 Staffing (S285) --- INTEGER (4).

(7) Maximum Procurement Quantity (MPQ) --- INTEGER (7).

(8) Shifts Available (AVAIL) --- INTEGER (1).

d. Processing -

(1) The Item Production Capacity/Staffing screen incorporates a few unique rules.

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information. These rules include the user of a predelete and preselect statement, in addition to answering no to the update field prompt.

(b) The second block will not allow the user to delete information that is being retrieved from the PRODT table. In order to delete this information, the user would have to contact the PMO.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Item Production Capacity/Staffing screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the PRODT table.

(6) Updating and inserting new information about an item can only be accomplished in block 2, of the Item Production Capacity/ Staffing screen. The screen automatically checks to see if there is an active RCN. If not, a message will state that to the user. If there is an active RCN, the screen will take all of the information that can be updated/inserted for that DODIC and place the new data into the maximum RCN that is active. A user is unable to update baseline data which are stored under RCN zero. The user cannot delete information that is being retrieved from the PRODT table.

When inserting a new record, the following fields are mandatory:

(a) Plant (PLANT).

(b) Line (LINE).

(c) 1-8-5 Production Rate (R185).

(d) 1-8-5 Staffing (S185).

(e) 2-8-5 Production Rate (R285).

(f) 2-8-5 Staffing (S285).

(g) Shifts Available (AVAIL).

e. Output - Filled screen (see figure 5.11.1-1).

f. Security - The Item Production Capacity/Staffing screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

Item Production Capacity/Staffing

FSC: 1320 DODIC: D563 SSN: E28100

Nomenclature: PROJ 15MM HE ICM M483A1 W/O/FUZE

Plant	Line	1-8-5 Rate	1-8-5 Staff	2-8-5 Rate	2-8-5 Staff	MPQ	Shift Availability
LS	31	18000	218	36000	436		0
MI	80	18000	342	36000	684		1
KS	132	18000	169	36000	337		0
MS	350	10000	446	20000	892		1

Figure 5.11.1-1. Example of Item Production Capacity/Staffing Input Screen Filled with Information

(2) PRODT - contains eeesential production information about each item. This table contains the baseline production data for all items in the system.

5.11.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same value were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of haveing WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.11.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.11.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply mean that the user does not have access to that table.

5.11.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.11	3. PROGRAM NAME JSM 3.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.12 Program. General Plant Data Input Screen (JSM_4.INP)

5.12.1 Program Description. The General Plant Data screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 4.

a. Identification - The source code for the General Plant Data screen is identified by a file labeled JSM_4.INP. After being compiled through ORACLE IAG, a run file is created named JSM_4.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, General Plant Data, retrieves plant information from two PJSM data base tables titled, PLANT and STAFF. The purpose of this screen is to review, input, or update specific staffing information about each plant. The general plant information that appears at the top of this screen cannot be deleted here. Once specific data have been entered in the second half of the screen, they too cannot be deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From PLANT table:

(1) Plant Name (NAME) --- CHARACTER (15).

(2) Plant Code (PLANT) --- CHARACTER (2).

(3) Type of Plant (TYPE) --- CHARACTER (4).

(4) Production Overhead Factor (PROD_OH_FAC) --- NUMBER (F6.4).

(5) Non-Production Overhead Factor (NON_PROD_OH_FAC) --- NUMBER (F6.4).

(b) From STAFF table:

(1) FY --- INTEGER (2).

(2) RCN --- INTEGER (3).

(3) Direct Labor Employees (DIRECT) --- INTEGER (5).

(4) OMA Overhead Employees (OMA_OH) --- INTEGER (5).

(5) Percent of the Workload Attempting to be Achieved (GOAL) --- INTEGER (3).

d. Processing -

(1) The General Plant Data screen incorporates a few unique rules:

(a) In the first block, which pertains to the PLANT table, this screen uses a default WHERE and ORDER BY clause which is by NAME. If a query is executed with no given information, plants will be returned to the screen alphabetically. The first block also uses a rule that will not allow the user to delete general plant information.

(b) The second block, which relates to the STAFF table, also uses a default WHERE and ORDER BY clause which is by FY. The data which relate to the PLANT in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the table being used in the General Plant Data screen.

(4) From PJSM data base, PLANT table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the 2-digit plant code fo the plant. The information in block 2 is retrieved from the STAFF table.

(6) Updating and inserting new information about a plant can be accomplished in block 1 and 2.

(a) When a user updates or inserts information in block 1, the data are directly placed into the PLANT table.

(b) When a user attempts to update or insert new information in block 2, the screen automatically checks to see if there is an active RCN. If not, a message will state that to the user. If there is an active RCN, the screen will take all of the information that can be updated/inserted for that PLANT and place the new data into the maximum RCN that is active. Block 2 also has formulas programmed in the sequel statement for the direct labor field. These are executed once a value for the direct labor is input to the screen. The first formula multiples the production overhead factor and the direct labor and places the rounded value into the total production overhead field. The second formula multiplies the production overhead factor and the direct labor, adds that to the OMA and direct labor, and multiplies that total by the non-production overhead factor. The rounded total is then inserted into the total non-production overhead field. Once a query is executed, the screen will add all appropriate fields and figure the total staff available. The user cannot delete information that is being retrieved from the STAFF table.

(c) When inserting a new record, the following fields are mandatory:

(1) Plant Name (NAME).

(2) Plant Code (PLANT).

(3) Type of Plant (TYPE).

(4) Production Overhead Factor (PROD_OH_FAC).

(5) Non-Production Overhead Factor (NON_PROD_OH_FAC).

(6) FY.

(7) Direct Labor Employees (DIRECT).

(8) OMA Overhead Employees (OMA_OH).

(9) Percent of the Workload Attempting to be Achieved (GOAL).

e. Output - Filled screen (see figure 5.12.1-1).

f. Security - The General Plant Data screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will have access only through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) PLANT - contains general information for each plant. This general information includes the production overhead factors and the non-production overhead factors.

(2) STAFF - contains specific staffing data about each plant. This table contains the baseline staffing data for all plants in the system.

5.12.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.12.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.12.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.12.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

General Plant Total Staffing Data

Plant Name: IOWA 2 digit code: IA Type: GOCO

Overhead Factors: (1.93) (.18)

FY	RCN	Direct Labor	Prod Oh	Non Prod Oh	OMA	Total Staffing	Goal
88	0	650	1255	344	9	2258	100
89	0	650	1255	344	9	2258	100
90	0	650	1255	344	9	2258	100
91	0	650	1255	344	9	2258	100
92	0	650	1255	344	9	2258	100
93	0	650	1255	344	9	2258	100

Figure 5.12.1-1. Example of General Plant Data Input Screen Filled with Information

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DUSOPS.		30 Nov 87
2. PROGRAM ID 5.12	3. PROGRAM NAME JSM 4.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.13 Program. Primary Component Information Input Screen (JSM_5.INP)

5.13.1 Program Description. The Primary Component Information screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 5.

a. Identification - The source code for the Primary Component Information screen is identified by a file labeled JSM_5.INP. After being compiled through ORACLE IAG, a run file is created named JSM_5.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Primary Component Information, retrieves primary component relationships between items as well as their usage factors. This information comes from two PJSM data base tables titled, ITEM and ICT. The purpose of this screen is to review, input, or update specific component information about each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From ICT table:

(1) Component DODIC (COMP) --- CHARACTER (4).

(2) Usage Factor (PF) --- NUMBER (F11.6).

(c) In block 2, the FSC and Nomenclature for the components are retrieved from the ITEM table.

d. Processing -

(1) The Primary Component Information screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information.

(b) The second block, which relates to the ICT table, also uses a default WHERE and ORDER BY clause which is by type. Only the primary components related to the DODIC in block 1 will be displayed.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Primary Component Information screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the ICT table.

(6) Updating and inserting new information about an item can only be accomplished in block 2, or the ICT table. When executed, the screen will replace the old value in the ICT table

with the new value the user has input. The user cannot delete information that is being retrieved from the ICT table.

When inserting a new record, the following fields are mandatory:

(a) Component DODIC (COMP).

(b) Usage Factor (PF).

e. Output - Filled screen (see figure 5.13.1-1).

f. Security - The Primary Component Information screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) ICT - contains all end item/component relationships. This table also contains all of the usage factors of the components and it lists the type of component, primary or secondary.

5.13.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.13.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

Primary Component Information

FSC: 1315 DODIC: C256 SSN: E18401

Nomenclature: CTG 81MM HE M374A3

Primary Components

FSC	DODIC	Nomenclature	USAGE FACTOR
1499	X142	BLACK POWDER	.000468
1499	X143	LEAD AZIDE	.000024
1499	X172	PROPELLANT SB M10	.2352
1499	X177	PROPELLANT DB M9	.01725
1499	X220	CTG IGN M299 F/81MM	1.05
1499	X223	LEAD CUP ASSY F/PROJ 15MM M692/M731	1.25
1499	X224	PRIMER PERC M54	1.5
1499	X227	DELAY ELEM M53	1.05
1499	X228	DETONATOR STAB M76	1.08
1499	X229	DETONATOR SQ M98	1.08
1499	X254	LEAD ASSY F/M567	1.05
1376	ZZ08	RDX TYPE II	.005575
1376	ZZ11	COMPOSITION B	2.4465
1376	ZZ13	COMPOSITION A-5	.001742

Figure 5.13.1-1. Example of Primary Component Information Input Screen Filled with Information

5.13.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.13.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.13	JSM 5.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.14 Program. Yearly Consumption Requirements Input Screen (JSM_6.INP)

5.14.1 Program Description. The Yearly Consumption Requirements screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 6.

a. Identification - The source code for the Yearly Comsumption Requirements screen is identified by a file labeled JSM_6.INP. After being compiled through ORACLE IAG, a run file is created named JSM_6.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Yearly Consumption Requirements, retrieves item information from two PJSM data base tables titled, ITEM and REQTS_ARMY. The purpose of this screen is for reviewing the training requirements for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed either through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4)

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From REQTS_ARMY table:

(1) FY --- INTEGER (2).

(2) Package Level (PKGL) --- INTEGER (2)

(3) Required Quantity (QUANTITY) --- INTEGER (10).

d. Processing -

(1) The Yearly Consumption Requirements screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second through sixth blocks, which relate to REQTS_ARMY table, also use default WHERE and ORDER BY clauses which are by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Yearly Consumption Requirements screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2, 3, 4, 5, or 6 with the tie between the blocks being the DODIC of the item. The information in blocks 2 through 6 is retrieved from the REQTS_ARMY table.

(6) The screen incorporates rules designed specifically for the user which only allows data that equal certain package levels to be displayed. For this screen, they are the packages considered training requirements. Another rule incorporates logic to search the REQTS_ARMY table for training requirements. If the screen fails to find any, it will retrieve the requirement from a preset item. This item simply has a zero as its requirement.

(7) Certain rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.14.1-1).

f. Security - The Yearly Consumption Requirements screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) REQTS_ARMY - contains all of the Army requirements for each item. This table is constructed with a column where new requirements are stored until they can be verified. This ensures that baseline data will not be deleted accidently.

5.14.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.14.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

FOR VIEWING ONLY*
****

Consumption Requirements

FSC: 1305 DODIC: A080 SSN: E01102 NSN: 1305-00-182-3217 ISN: 00102

Nomenclature: CTG 5.56MM SGL RD BLANK M200

Package		Program Year 89	90	91	92	93
2	Test	36000	26000	30000	23000	23000
5	Level 1 Training	150000000	150000000	150000000	150000000	150000000
6	Depot 1	82499984	82499984	82499984	82499984	82499984
10	Level 2 Training	162630976	162630976	162630976	162630976	162630976
11	Depot 2	89447024	89447024	89447024	89447024	89447024

Figure 5.14.1.1 Example of Yearly Consumption Requirements Input Screen Filled with Information

5.14.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.14.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.14	3. PROGRAM NAME JSM 6.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.15 Program. Production Project Specification Input Screen (JSM_7.INP)

5.15.1 Program Description. The Production Project Specification screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 7.

a. Identification - The source code for the Production Project Specification screen is identified by a file labeled JSM_7.INP. After being compiled through ORACLE IAG, a run file is created named JSM_7.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Production Project Specification, retrieves item information from two PJSM data base tables titled, PROJECT and PRODT_FY. The purpose of this screen is to review, input, or update specific project data that are related to an item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed either through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From PROJECT table:

(1) Project Code (PROJECT) --- CHARACTER (8).

(2) Planning Year (PLAN_YR) --- INTEGER (2).

(3) Cost of the Project (COST) --- NUMBER (F7.1); e.g., XXXXXXX.X.

(4) Title of Project (TITLE) --- CHARACTER (30).

(b) From PRODT_FY table:

(1) DODIC --- CHARACTER (4).

(2) RCN --- INTEGER (2).

(3) Plant (PLANT) --- CHARACTER (2).

(4) Line (LINE) --- INTEGER (3).

(5) Month Available (MO) --- INTEGER (2).

(6) FY --- INTEGER (2).

(7) 1-8-5 Production Rate (R185) --- INTEGER (9).

(8) 1-8-5 Staffing (S185) --- INTEGER (4).

(9) 2-8-5 Production Rate (R285) --- INTEGER (9).

(10) 2-8-5 Staffing (S285) --- INTEGER (4).

(11) Shifts Available (AVAIL) --- INTEGER (1).

(12) Minimum Procurement Quantity (MPQ) --- INTEGER (7).

(13) Maximum Production Allowed (MPA) --- INTEGER (10).

d. Processing -

(1) The Production Project Specifications screen incorporates a few unique rules:

(a) In the first block, which pertains to the PROJECT table, this screen uses a default WHERE and ORDER BY clause which is by project code. If a query is executed with no given information, projects will be returned to the screen in ascending order. These rules include the use of a predelete statement, in addition to answering no to the update field prompt.

(b) The second block, which relates to the PRODT_FY table, also uses a default WHERE and ORDER BY clause which is by DODIC. The data which relate to the project in block 1 will be displayed in DODIC order.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Production Project Specifications screen.

(4) From PJSM data base, PROJECT table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the 2-digit plant code of the plant. The information in block 2 is retrieved from the PRODT_FY table.

(6) Updating and inserting new information about a project can be accomplished in block 1 and 2.

(a) When a user updates or inserts information in block 1, the data are directly placed into the PROJECT table.

(b) When a user attempts to update or insert new information in block 2, the screen automatically checks to see if there is an active RCN. If not, a message will state that to the user. If there is an active RCN, the screen will take all of the information that can be updated/inserted for that PROJECT and place the new data into the maximum RCN that is active. A user is unable to update baseline data which are stored under RCN zero. The user cannot delete information that is being retrieved from the PRODT_FY table.

(c) When inserting a new record, the following fields are mandatory:

(1) Project Code (PROJECT).

(2) Planning Year (PLAN_YR).

(3) Cost of the Project (COST).

(4) Title of the Project (TITLE).

(5) DODIC.

(6) Plant (PLANT).

(7) Line (LINE).

(8) Month Available (AVAIL).

(9) FY.

(10) 1-8-5 Production Rate (R185).

(11) 1-8-5 Staffing (S185).

(12) 2-8-5 Production Rate (R285).

(13) 2-8-5 Staffing (S285).

(14) Shifts Available (AVAIL).

e. Output - Filled screen (see figure 5.15.1-1).

f. Security - The Production Project Specification screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) PROJECT - contains general information for each project. This table contains the identifiers from which data for all projects relate to.

(2) PRODT_FY - contains item data that are based on various projects. This table contains the specific data for an item based on a project.

5.15.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

Production Project Specifications

PROJECT: 5852386 PLANNING FY: 85 COST$(M): 10

TITLE: EXP COMBINED EFFECTS MUNITION

DODIC	RCN	Plt	Line	Date Mon-YR	1-8-5 Rate	1-8-5 Staff	2-8-5 Rate	2-8-5 Staff	Sft Avl	Min Proc Quantity	Max Prod Allowed
X135	1	KS	131	3 87	3330	375	8325	750	1		

Figure 5.15.1-1. Example of Production Project Specification Input Screen Filled with Information

5.15.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.15.4 Error Conditions. When a user receives a message that an ORACLE error has occured, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.15.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.15	JSM 7.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.16 Program. Army Requirements Stratification Input Screen (JSM_8.INP)

5.16.1 Program Description. The Army Requirements Stratification screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 8.

a. Identification - The source code for the Army Requirements Stratification screen is identified by a file labeled JSM_8.INP. After being compiled through ORACLE IAG, a run file is created named JSM_8.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Army Requirements Stratification, retrieves item information from two PJSM data base tables titled, ITEM and REQTS_ARMY. The purpose of this screen is for reviewing all Army requirements for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through either the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN -- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From REQTS_ARMY table:

(1) FY --- INTEGER (2).

(2) Package Level (PKGL) --- INTEGER (2).

(3) Required Quantity (QUANTITY) --- INTEGER (10).

d. Processing -

(1) The Army Requirements Stratification screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second through sixth blocks, which relate to REQTS_ARMY table, also use default WHERE and ORDER BY clauses which are by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Army Requirements Stratification screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on blocks 2, 3, 4, 5, or 6 with the tie between the blocks being the DODIC of the item. The information in blocks 2 through 6 is retrieved from the REQTS_ARMY table.

(6) The screen incorporates rules designed specifically for the user which checks if the item is a New Materiel Fielding Item. If so, the screen will add packages 2 through 7 and place the total in package 1. Another rule incorporates logic to search the REQTS_ARMY table for an Army requirement. If the screen fails to find any, it will retrieve the requirement from a preset item. This item simply has a zero as its requirement.

(7) Certain rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.16.1-1).

f. Security - The Army Requirements Stratification screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) REQTS_ARMY - contains all of the Army requirements for each item. This table is constructed with a column where new requirements are stored until they can be verified. This ensures that baseline data will not be deleted accidently.

5.16.2 Conventions

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.16.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.16.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

FOR VIEWING ONLY * Army Requirements Stratification
****

FSC: 1320 DODIC: D528 SSN: E67800 NSN: 1320-01-140-2611 ISN: 08453
Nomenclature: PROJ 155MM SMK M825 W/O FUZE

Package		89	90	91	92	93
1	MMF					
2	Test	1000	0	0	0	0
3	AIIQ	3000	4000	4000	3000	3000
4	OPS	0	0	0	0	0
5	Level 1 Training	0	0	0	0	0
6	Depot 1	0	0	0	0	0
7	War Reserve 1.0	159000	188000	191000	189000	189000
8	War Reserve 2.0	129000	145000	149000	148000	148000
9	War Reserve 3.0	139000	148000	151000	153000	153000
10	Level 2 Training	0	0	0	0	0
11	Depot 2	0	0	0	0	0
12	MOB A	0	0	0	0	0
13	WRSA 1.0	18000	18000	18000	18000	18000
14	War Reserve 4.0	161000	163000	170000	171000	171000
15	WRSA 2.0	2000	2000	2000	2000	2000
16	MOB B	0	0	0	0	0
17	War Reserve 5.0	103000	106000	117000	112000	112000

Figure 5.16.1-1. Example of Army Requirements Stratification Input Screen Filled with Information

5.16.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.16	3. PROGRAM NAME JSM 8.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.17 Program. Other Customer Requirements-Order Input Screen (JSM_9.INP)

5.17.1 Program Description. The Other Customer Requirements-Order screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 9.

a. Identification - The source code for the Other Customer Requirements-Order screen is identified by a file labeled JSM_9.INP. After being compiled through ORACLE IAG, a run file is created named JSM_9.FRM, both located in SAXPGM $INP.

b. Functions - The input screen, Other Customer Requirements-Order, retrieves item information from two PJSM data base tables titled, ITEM and REQTS_OTHER. The purpose of this screen is for reviewing all other service requirements for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through either the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From REQTS_OTHER table:

(1) FY --- INTEGER (2).

(2) Service (SERVICE) --- CHARACTER (2).

(3) Quantity (QUANTITY) --- INTEGER (10).

d. Processing -

(1) The Other Requirements-Order screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second through sixth blocks, which relate to the REQTS_OTHER table, also use default WHERE and ORDER BY clauses which are by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Other Customer Requirements-Order screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2, 3, 4, 5, or 6 with the tie between the blocks being the DODIC of the item. The information in blocks 2 through 6 is retrieved from the REQTS_OTHER table.

(6) The screen incorporates a rule designed specifically for the user which checks if the item has an other service requirement for each service, in each year. If the screen fails to find one, it will retrieve the requirement from a preset item. This item simply has a zero as its requirement.

(7) Certain rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.17.1-1).

f. Security - The Other Customer Requirements-Order screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) REQTS_OTHER - contains all of the other service requirements for each item. This table is constructed with a column where new requirements are stored until they can be verified. This ensures that baseline data will not be deleted accidently.

5.17.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.17.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.17.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Error Display function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

FOR VIEWING ONLY *
********** Other Customer Requirements-Orders

FSC: 1305 DODIC: A540 SSN: E06903 NSN: 1305- ISN: 00900

Nomenclature: CTG CAL .50 LKD 4API MB/1 TR M17

Customer		Program Year/Production Year 89	90	91	92	93
AF	Air Force	1499453	666000	666000	666000	666000
AM	Army Misc	0	0	0	0	0
CG	Coast Guard	149200	149200	149200	149200	149200
FM	FMS	0	0	0	0	0
MC	Marine Corps	3118689	1584486	1404464	1425186	1425186
NA	Nav Air	0	0	0	0	0
NS	Nav Sea	13322534	5525000	7000000	6859000	6859000
OT	Other	1500000	1500000	1500000	1500000	1500000

Figure 5.17.1-1. Example of Other Customer Requirements - Orders Input Screen Filled with Information

5.17.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.17	3. PROGRAM NAME JSM_9.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.18 Program. Secondary Component Information Input Screen (JSM_10.INP)

5.18.1 Program Description. The Secondary Component Information screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 10.

a. Identification - The source code for the Secondary Component Information screen is identified by a file labeled JSM_10.INP. After being compiled through ORACLE IAG, a run file is created named JSM_10.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Secondary Component Information, retrieves secondary component relationships between items as well as their usage factors. This information comes from two PJSM data base tables titled, ITEM and ICT. The purpose of this screen is to review, input, or update specific component information about each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (4).

(4) Nomenclature --- CHARACTER (48).

(b) From ICT table:

(1) Component DODIC (COMP) --- CHARACTER (4).

(2) Usage Factor (PF) --- NUMBER (F11.6).

(c) In block 2, the FSC and Nomenclature for the components are retrieved from the ITEM table.

d. Processing -

(1) The Secondary Component Information screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information.

(b) The second block, which relates to the ICT table, also uses a default WHERE and ORDER BY clause which is by type. Only the secondary components related to the DODIC in block 1 will be displayed.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Secondary Component Information screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the ICT table.

(6) Updating and inserting new information about an item can only be accomplished in block 2, or the ICT table. When executed, the screen will replace the old value in the ICT table

with the new value the user has input. The user cannot delete information that is being retrieved from the ICT table.

When inserting a new record, the following fields are mandatory:

(a) Component DODIC (COMP).

(b) Usage Factor (PF).

e. Output - Filled screen (see figure 5.18.1-1).

f. Security - The Secondary Component Information screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) ICT - contains all end item/component relationships. This table also contains all of the usage factors of the components and it lists the type of component, primary or secondary.

5.18.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.18.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

Secondary Component Information

FSC: 1320 DODIC: D864 SSN: E68500

Nomenclature: PROJ 155MM ER XM864

Secondary Components

FSC	DODIC	Nomenclature	USAGE FACTOR
1320	D532	CHG PROP 155MM RB ZONE M203 W/O PRIMER	1.05
1390	M285	FUZE MTSQ M577	1.05
1390	M523	PRIMER PERC M82	1.05
1390	T762	FUZE ET M762	1.05

Figure 5.18.1-1. Example of Secondary Component Information Input Screen Filled with Information

5.18.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.18.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION	1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.	30 Nov 87
2. PROGRAM ID 5.18	3. PROGRAM NAME JSM 10.INP
4. REV NO./DATE	5. DESCRIPTION OF REVISION

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.19 Program. Line Descriptors Input Screen (JSM_11.INP)

5.19.1 Program Description. The Line Descriptors screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 11.

a. Identification - The source code for the Line Descriptors screen is identified by a file labeled JSM_11.INP. After being compiled through ORACLE IAG, a run file is created named JSM_11.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Line Descriptors, retrieves general line information from the PJSM data base table titled LINE. The purpose of this screen is to review or input general line data.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base: From LINE table.

(a) Model Line Number (LINE) --- INTEGER (3).

(b) Plant Code (PLANT) --- CHARACTER (2).

(c) Line Description (LINE_DESC) --- CHARACTER (40).

d. Processing -

(1) The Line Descriptors screen incorporates a few unique rules:

(a) In the first block, which pertains to the LINE table, this screen uses a default WHERE and ORDER BY clause, which is by line number. If a query is executed with no given information, the order in which the lines will be returned to the screen is in ascending order.

(b) The user is unable to delete any line information through this screen.

(2) Read terminal type to set function keys appropriately on each users terminal.

(3) Read ORACLE identification and password to determine if user has access to the table being used in the Line Descriptors screen.

(4) From PJSM data base, LINE table, read information associated with value query was executed on. Display to input screen.

(5) When updating value, replace old value in LINE table with new value as inserted to the screen, based on key identifier (LINE NUMBER).

(6) When inserting a new line (record), enter screen values into LINE table.

(a) When inserting a new record, the following fields are mandatory:

(1) Model Line Number (LINE).

(2) Plant Code (PLANT).

e. Output - Filled screen (see figure 5.19.1-1).

f. Security - The Line Descriptors screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following table in the PJSM data base:

(1) LINE - contains general information for each line. This table's main function is to store a description of the line the model is using for simulation purposes.

5.19.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

Line - Plant Cross-reference

Line	Code	Plant Name	Description
1	IA	IOWA	LINE 4A
2	IA	IOWA	LINE 3
3	IA	IOWA	LINE 9
4	IA	IOWA	LINE 9
5	IA	IOWA	LINE 5A
6	IA	IOWA	LINE 2
7	IA	IOWA	LINE 2
8	IA	IOWA	LINE 2
9	IA	IOWA	LINE 3A
10	IA	IOWA	LINE 1
11	IA	IOWA	LINE 1
12	IA	IOWA	LINE 1
13	IA	IOWA	LINE 4B
14	IA	IOWA	LINE 4B
15	IA	IOWA	
16	IA	IOWA	
17	IA	IOWA	
18	IA	IOWA	

Figure 5.19.1-1. Example of Line Descriptors Input Screen Filled with Information

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.19.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.19.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.19.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.19	3. PROGRAM NAME JSM 11.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.20 Program. Update or Display Army Requirements Input Screen (JSM_12.INP)

5.20.1 Program Description. The Update or Display Army Requirements Input screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAF is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 12.

a. Identification - The source code for the Update or Display Army Requirements Input screen is identified by a file labeled JSM_12.INP. After being compiled through ORACLE IAG a run file is created named JSM_12.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Update or Display Army Requirements, retrieves item information from two PJSM data base tables titled, ITEM and REQTS_ARMY. The purpose of this screen is to review, insert, or update Army requirements for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From REQTS_ARMY table:

(1) Package Level (PKGL) --- INTEGER (2).

(2) FY --- INTEGER (2).

(3) Required Quantity (QUANTITY) --- INTEGER (10).

(4) Quantity (CHANGE) --- INTEGER (10).

d. Processing -

(1) The Update or Display Army Requirements screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information.

(b) The second block, which relates to REQTS_ARMY table, also uses a default WHERE and ORDER BY clause which is by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Update or Display Army Requirements.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the REQTS_ARMY table.

(6) Updating and inserting new information about an item can only be accomplished in block 2 of the Update or Display Army Requirements screen. When updating an item, the user will insert the new requirement into the new quantity field for the corresponding package level and year. The screen will compute the difference between the original quantity and the new quantity and insert that difference into the Change field. If there is no revised quantity, the new quantity field will remain blank. Users can insert new requirements by inputting the year, package level, and new value. The user cannot delete information that is being retrieved from the REQTS_ARMY table.

When inserting a new record, the following fields are mandatory:

(a) Package Level (PKGL).

(b) FY.

(c) Quantity (CHANGE).

e. Output - Filled screen (see figure 5.20.1-1).

f. Security - The Update or Display Army Requirements screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for item in the data base, it is used in the majority of the input screens.

(2) REQTS_ARMY - contains all of the Army requirements for each item. This table is constructed with a column where new requirements are stored until they can be verified. This ensures that baseline data will not be deleted accidently.

5.20.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

Army Requirements Input Screen

FSC: 1320 DODIC: D528 SSN: E67800 NSN: 1320-01-140-2611 ISN: 08453

Nomenclature: PROJ 155MMSMK M825 W/O FUZE

Package Level	FY	Quantity	New Quantity
2 Test	89	1000	0
3 AIIQ	88	3000	0
3 AIIQ	89	3000	0
3 AIIQ	90	4000	0
3 AIIQ	91	4000	0
3 AIIQ	92	3000	0
3 AIIQ	93	3000	0
7 War Reserve 1.0	88	158000	0
7 War Reserve 1.0	89	159000	0
7 War Reserve 1.0	90	188000	0
7 War Reserve 1.0	91	191000	0
7 War Reserve 1.0	92	189000	0

Figure 5.20.1-1. Example of Update or Display Army Requirements Input Screen Filled with Information

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.20.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.20.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.20.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.20	3. PROGRAM NAME JSM 12.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.21 Program. Update or Display Other Service Requirements Input Screen (JSM_13.INP)

5.21.1 Program Description. The Update or Display Other Service Requirements Input screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 13.

a. Identification - The source code for the Update or Display Other Service Requirements Input screen is identified by a file labeled JSM_13.INP. After being compiled through ORACLE IAG a run file is created named JSM_13.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, Update or Display Other Service Requirements, retrieves item information from two PJSM data base tables titled, ITEM and REQTS_OTHER. The purpose of this screen is to review, insert, or update other service requirements for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through either the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) ISN --- CHARACTER (5).

(6) Nomenclature --- CHARACTER (48).

(b) From REQTS_OTHER table:

(1) Service Code (SERVICE) --- CHARACTER (2).

(2) FY --- INTEGER (2).

(3) Unit Price (UNIT_PRICE) --- NUMBER (F12.4).

(4) Required Quantity (QUANTITY) --- INTEGER (10).

(5) Quantity (CHANGE) --- INTEGER (10).

d. Processing -

(1) The Update or Display Other Service Requirements screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

(b) The second block, which relates to REQTS_OTHER table, also uses a default WHERE and ORDER BY clause which is by FY. The data which relate to the DODIC in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Update or Display Other Service Requirements screen.

(4) From PJSM data base, ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block with the tie between the two blocks being the DODIC of the item. The information in block 2 is retrieved from the REQTS_OTHER table.

(6) Updating and inserting new information about an item can only be accomplished in block 2 of the Update or Display Other Service Requirements screen. When updating an item, the user will insert the new requirement into the new quantity field for the corresponding service and year. The screen will compute the difference between the original quantity and the new quantity and insert that difference into the Change field. If there is no revised quantity, the new quantity field will remain blank. Users can insert new requirements by inputting the year, service, and new value. The user cannot delete information that is being retrieved from the REQTS_OTHER table.

When inserting a new record, the following fields are mandatory:

(a) Service Code (SERVICE).

(b) FY.

(c) Quantity (CHANGE).

e. Output - Filled screen (see figure 5.21.1-1).

f. Security - The Update or Display Other Service Requirements screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will have access only through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) REQTS_OTHER - contains all of the service requirements for each item. This table is constructed with a column where new requirements are stored until they can be verified. This ensures that baseline data will not be deleted accidently.

Other Service Requirements Input Screen

FSC: 1305 DODIC: A540 SSN: E06903 NSN: 1305- ISN: 00900

Nomenclature: CTG CAL .50 LKD 4API M8/1 TR M17

Service		FY	Unit Price	Quantity	New Quantity
AF	Air Force	89	1.2331	1499453	0
AF	Air Force	90	1.3104	666000	0
AF	Air Force	91	1.344	666000	0
AF	Air Force	92	1.3763	666000	0
AF	Air Force	93	1.3763	666000	0
CG	Coast Guard	88	1.1908	149200	0
CG	Coast Guard	89	1.2331	149200	0
CG	Coast Guard	90	1.3104	149200	0
CG	Coast Guard	91	1.344	149200	0
CG	Coast Guard	92	1.3763	149200	0
CG	Coast Guard	93	1.3763	149200	0
MC	Marine Corps	88	1.1908	970814	0

Figure 5.21.1-1. Example of Update or Display Other Service Requirements Input Screen Filled with Information

5.21.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.21.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.21.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.21.5 Listings. Program listing is located in <SYSSA>SAXPGM>#INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.21	3. PROGRAM NAME JSM 13.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.22 Program. DODIC/Project Cross-Reference Input Screen (JSM_14.INP)

5.22.1 Program Description. The DODIC/Project Cross-Reference screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 14.

a. Identification - The source code for the DODIC/Project Cross-Reference screen is identified by a file labeled JSM_14.INP. After being compiled through ORACLE IAG a run file is created named JSM_14.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, DODIC/Project Cross-Reference, retrieves project and related information from the PJSM data base tables titled PRODT_FY, ITEM, and PLANT. The purpose of this screen is for reviewing project data. The information that appears on this screen cannot be deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From PRODT_FY table:

(1) DODIC --- CHARACTER (4).

(2) Project Code (PROJECT) --- CHARACTER (8).

(b) From ITEM table: Nomenclature --- CHARACTER (48).

(c) From PLANT table: Plant Name (PLANT) --- CHARACTER (15).

d. Processing -

(1) The DODIC/Project Cross-Reference screen incorporates a few unique rules:

(a) In the first block, which pertains to the PRODT_FY table, this screen uses a default WHERE and ORDER BY

clause, which is by DODIC. If a query is executed with no given information, the order in which items will be returned to the screen is by DODIC.

(b) The user is unable to delete any project information through this screen.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the DODIC/Project Cross-Reference screen.

(4) From PJSM data base, PRODT_FY table, read information associated with value query was executed on. Display to input screen.

(5) From PJSM data base, ITEM table, read the nomenclature that is related to the DODIC which was returned from the PRODT_FY table. Display to input screen. From PJSM data base, PLANT table, read the plant name that is related to the plant code which was returned from the PRODT_FY table. Display to input screen.

e. Output - Filled screen (see figure 5.22.1-1).

f. Security - The DODIC/Project Cross-Reference screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) PRODT_FY - contains item data that are based on various projects. This table contains the specific data for an item based on a project.

(3) PLANT - contains general information regarding the production facilities that are represented in the PJSM.

DODIC/Project Cross-Reference

DODIC	Nomenclature	Project #	Plant
A064	CTG 5.56MM LK 4BALL M855/1 TRCR M856 MLB M27		LAKE CITY
A064	CTG 5.56MM LK 4BALL M855/1 TRCR M856 MLG M27		LAKE CITY
B519	CTG 40MM PRAC M781		MILAN
B519	CTG 40MM PRAC M781		MILAN
C699	CTG 4.2 IN HE M329A W/O FUZE		LOUISIANA
C699	CTG 4.2 IN HE M329A W/O FUZE		LOUISIANA
C868	CTG 81MM HE IMPROVED UKM821 W/MULTI OPT FUZE M73		COMMERCIAL
C868	CTG 81MM HE IMPROVED UKM821 W/MULTI OPT FUZE M73		COMMERCIAL
C870	CTG 81MM SMK RP SCREENING XM819		PINE BLUFF
C995	LT WT MULTIPURPOSE SYSTEM (AT-4)		COMMERCIAL
D501	PROJ 155MM HE ADAM M692 W/O FUZE	5902471	LOUISIANA
D502	PROJ 155MM HE ADAM M731 W/O FUZE	5902471	LOUISIANA
D510	FINAL ASSY OF WHD COPPERHEAD M712		COMMERCIAL
D563	PROJ 155MM HE ICM M483A1 W/O FUZE	XYZ	MILAN
D563	PROJ 155MM HE ICM M483A1 W/O FUZE	XYZ1	KANSAS
D563	PROJ 155MM HE ICM M483A1 W/O FUZE		KANSAS

Figure 5.22.1-1. Example of DODIC/Project Cross Reference Input Screen Filled with Information

5.22.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.22.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.22.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist which simply means that the user does not have access to that table.

5.22.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.22	3. PROGRAM NAME JSM 14.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4782-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.23 Program. RAMP Item Input Screen (JSM_15.INP)

5.23.1 Program Description. The RAMP Item screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAF is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 15.

a. Identification - The source code for the RAMP Item screen is identified by a file labeled JSM_15.INP. After being compiled through ORACLE IAG, a run file is created named JSM_15.FRM, both located in SAXPGM:$INP.

b. Functions - The input screen, RAMP Item, retrieves item information from two PJSM data base tables titled ITEM and RAMP_ITEM. The purpose of this screen is to review, input, or update general production data for the RAMP years for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) Nomenclature --- CHARACTER (48).

(b) From RAMP_ITEM table:

(1) FY --- INTEGER (2).

(2) Beginning Calendar Month (BEG_MO) --- INTEGER (2).

(3) Beginning Calendar Year (BEG_CY) --- INTEGER (2).

(4) Ending Calendar Month (END_MO) --- INTEGER (2).

(5) Ending Calendar Year (END_CY) --- INTEGER (2).

(6) Number of Months in FDP (NO_MOS) --- INTEGER (2).

(7) Beginning Assets (BEG_ASSETS) --- INTEGER (10).

(8) Army Production (PROD_ARMY) --- INTEGER (10).

(9) Total Army Production (PROD_TOT) --- INTEGER (10).

(10) Test, Training, and Other Losses (LOSSES) --- INTEGER (10).

(11) Other Customer Deliveries (OC_DEL) --- INTEGER (10).

(12) Ending Assets (END_ASSETS) --- INTEGER (10).

(13) Army Buy Quantity (ARMY_BUY_QTY) --- INTEGER (10).

(14) Army Unit Price (UNIT_PRICE) --- NUMBER (F12.4).

(15) Dollar Value of Funded Army Quantity (DV_ARMY) --- INTEGER (10).

d. Processing -

(1) The RAMP Item screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information.

(b) The second through fourth blocks, which relate to RAMP_ITEM table, also use default WHERE and ORDER BY clauses which are by FY. The data which relate to the DODIC in block 1 will be displayed in order by FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the RAMP Item screen.

(4) From PJSM data base, ITEM table, read information associated with value the query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 through 4, with the tie between the blocks being the DODIC of the item. The information in block 2 thorugh 4 is retrieved from the RAMP_ITEM table.

(6) Updating and inserting new information about an item can only be accomplished in block 2 through 4. When a user updates or inserts information in these blocks, the data are directly placed into the RAMP_ITEM table. The user cannot delete information that is being retrieved from the RAMP_ITEM table.

When inserting a new record, the following fields are mandatory:

(a) Beginning Calendar Month (BEG_MO).

(b) Beginning Calendar Year (BEG_CY).

(c) Ending Calendar Month (END_MO).

(d) Ending Calendar Year (END_CY).

(e) Number of Months in FDP (NO_MOS).

(f) Beginning Assets (BEG_ASSETS) - (Only in the first year).

(g) Army Production (PROD_ARMY).

(h) Total Production (PROD_TOT).

(i) Test, Training, and Other Losses (LOSSES).

(j) Other Customer Deliveries (OC_DEL).

e. Output - Filled screen (see figure 5.23.1-1).

f. Security - The RAMP Item screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) RAMP_ITEM - contains general summary data by item for the RAMP years. This table and the RAMP_PROD table contain all of the item information for the RAMP years.

5.23.2 Conventions.

a. The data stored in ORACLE is case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used tc group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.23.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.23.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.23.5 Listings. Program listing is located in <SYSSA>SAXPGM>#INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

Ramp Year Program Status - Item Data

FSC: 1305 DODIC: A059 SSN: E04601 NSN: 1305-01-155-5462 NOMEN: CTG 5.56MM BALL

FDP:	FY86 & PRIOR	FY87	FY 88
Beginning Period (MO/CY)	10/86	10/87	10/88
End (MO/CY/# of Months)	7/87/10	7/88/12	7/89/12
Beginning Assets	77381000	130462000	119175000
Army Production	85343000	36113000	36490000
Total Production	109739000	129723000	139665000
Test/Trng/Other Losses	32262000	47400000	84802000
Other Customer Deliveries	24396000	93610000	103175000
End of Period Assets	130462000	119175000	70863000

ARMY BUY SUMMARY:	FY86	FY87	FY 88
Army Quantity		18845000	
Army Unit Price		.1751	
Total Dollars		3300000	

Figure 5.23.1-1. Example of RAMP Item Input Screen Filled with Information

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.23	3. PROGRAM NAME JSM 15.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.24 Program. RAMP Production Input Screen (JSM_16.INP)

5.24.1 Program Description. The RAMP Production screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 3, Review or Update Input Data, then option 1, Screens for Review or Update, and finally as option 16.

a. Identification - The source code for the RAMP Production screen is identified by a file labeled JSM_16.INP. After being compiled through ORACLE IAG, a run file is created named JSM_16.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, RAMP Production, retrieves item information from two PJSM data base tables titled ITEM and RAMP_PROD. The purpose of this screen is to review, input, or update general production data for the RAMP years for each item. The general item information that appears at the top of this screen cannot be updated, inserted, or deleted through this screen. These functions have to be performed either through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) NSN --- CHARACTER (16).

(5) Nomenclature --- CHARACTER (48).

(b) From RAMP_PROD table:

(1) FY --- INTEGER (2).

(2) Production Plant (PLANT) --- CHARACTER (2).

(3) Total Production (PROD_TOT) --- INTEGER (10).

(4) Dollar Value of Production (DV_TOTAL) --- INTEGER (10).

(5) Work-years for Total Production (WRKYRS_TOT) --- NUMBER (F7.2); e.g., XXXXXXX.XX.

(6) Total Undelivered RAMP Quantity (UNDEL_TOT) --- INTEGER (10).

(7) Army Undelivered RAMP Quantity (UNDEL_AR) --- INTEGER (10).

(8) Production Year Quantity (PROD_YR_QTY) --- INTEGER (10).

d. Processing -

(1) The RAMP Production screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

(b) The second through fourth blocks, which relate to RAMP_ITEM table, also use default WHERE and ORDER BY clauses which are by plant. The data which relate to the DODIC in block 1 will be displayed in order by plant.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the RAMP Production screen.

(4) From PJSM data base, ITEM table, read information associated with value the query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 through 4, with the tie between the blocks being the DODIC of the item. The information in block 2 thorugh 4 is retrieved from the RAMP_PROD table.

(6) Updating and inserting new information about an item can only be accomplished in block 2 through 4. When a user updates or inserts information in these blocks, the data are directly placed into the RAMP_PROD table. The user cannot delete information that is being retrieved from the RAMP_PROD table.

When inserting a new record, the following fields are mandatory:

(a) Production Plant (PLANT).

(b) Total Undelivered RAMP Quantity (UNDEL_TOT).

(c) Army Undelivered RAMP Quantity (UNDEL_AR).

e. Output - Filled screen (see figure 5.24.1-1).

f. Security - The RAMP Production screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM system will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the input screens.

(2) RAMP_PROD - contains general summary data by item and plant for the RAMP years. This table and the RAMP_ITEM table contain all of the item information for the RAMP years.

5.24.2 Conventions.

a. The data stored in ORACLE is case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates

Ramp Year Program Status - Production Data

FSC: 1305 DODIC: A059 SSN: E04601 NSN: 1305-01-155-5462 NOMEN: CTG 5.56MM BALL

PLANT: FDP:	FY86	LC FY87	LC FY88
Total Production		100472400	144391052
Dollar Value of Prod.		3339705	8081997
WRKYRS to produce Qty.		29.75	42.75
Total RAMP Balance		0	0
Army RAMP Balance		0	0

PRODUCTION YEAR SUMMARY:	FY87	FY88	FY89
Quantity			

Figure 5.24.1-1. Example of RAMP Production Input Screen Filled with Information

to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.24.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.24.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.24.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.24	3. PROGRAM NAME JSM 16.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.25 Program. JSM Simulation Guidelines Input Screen (GUID_1.INP)

5.25.1 Program Description. The JSM Simulation Guidelines screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 1, Set up or Modify Study Parameters, and then option 1.

a. Identification - The source code for the JSM Simulation Guidelines screen is identified by a file labeled GUID_1.INP. After being compiled through ORACLE IAG, a run file is created named GUID_1.FRM, both located in SAXPGM>$INP.

b. Functions - The input screen, JSM Simulation Guidelines, retrieves model guidance information from two PJSM data base tables titled, REVTAB and GOALS. The purpose of this screen is to review, input, or update the guidance to be used in PJSM mode runs.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Status (STATUS) --- CHARACTER (1).

(3) Date (CREATED) --- DATE (14).

(4) Analyst (AUSER) --- CHARACTER (12).

(5) Title (REMARKS) --- CHARACTER (40).

(6) Beginning FY (BFY) --- INTEGER (2).

(7) Number of FYs (NFY) --- INTEGER (2).

(8) Level of Training (LVTRNG) --- INTEGER (1).

(9) Depot Level (DEPOTR) --- INTEGER (1).

(10) Use Initial Assets (ASSETS) --- CHARACTER (1).

(11) Other Service Use Assets (SHARING) --- CHARACTER (1).

(b) From GOALS table:

(1) FY --- INTEGER (2).

(2) Buildup (BUILDUP) --- INTEGER (3).

(3) Draw Down (DRAWDN) --- INTEGER (3).

(4) Training Level Percentage (PCT_TRNG) --- INTEGER (3).

(5) Activity I Dollars (ACTIVITY_I) --- NUMBER (F7.1).

(6) Activity II Dollars (ACTIVITY_II) --- NUMBER (F7.1).

d. Processing -

(1) The JSM Simulation Guidelines screen incorporates a few unique rules:

(a) In the first block, which pertains to the REVTAB table, this screen uses a default WHERE and ORDER BY clause which is by RCN. If a query is executed with no given information, RCN's or studies, will be returned to the screen in ascending order.

(b) The second block, which relates to the GOALS table, also uses a default WHERE and ORDER BY clause which is by FY. The data which relate to the RCN in block 1 will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the table being used in the JSM Simulation Guidelines screen.

(4) From PJSM data base, REVTAB table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 with the tie between the two blocks being the RCN of the study. The information in block 2 is retrieved from the GOALS table.

(6) Updating and inserting new information about an RCN can be accomplished in block 1 and 2.

(a) When a user updates or inserts information in block 1, the data are directly placed into REVTAB table.

(b) When a user updates or inserts new information in block 2, the data are directly placed into the GOALS table.

(c) When inserting a new record, the following fields are mandatory:

(1) RCN.

(2) Status (STATUS).

(3) Date (CREATED).

(4) Analyst (AUSER).

(5) Title (REMARKS).

(6) Beginning FY (BFY).

(7) Number of FYs (NFY).

(8) Level of Training (LVTRNG).

(9) Depot Level (DEPOTR).

(10) Use Initial Assets (ASSETS).

(11) Other Service Use Assets (SHARING).

(12) FY.

(13) Buildup (BUILDUP).

(14) Draw Down (DRAWDN).

(15) Training Level Percentage (PCT_TRNG).

(16) Activity I Dollars (ACTIVITY_I).

(17) Activity II Dollars (ACTIVITY_II).

e. Output - Filled screen (see figure 5.25.1-1).

f. Security - The JSM Simulation Guidelines screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following table in the PJSM data base:

(1) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(2) GOALS - contains rules that the PJSM accesses for a model run. One of the important rules contained here is the dollar amount the model will use in figuring how much production to allow.

5.25.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.25.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.25.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.25.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

JSM Simulation Guidelines

RCN: 0 Active (Y/N): Y Date: 14-AUG-86
Analyst: WELLS Title: Baseline Data

Beginning Fiscal Yr	Number of Fiscal Yrs	Training Level	Depot Level	Use Initial Assets (Y/N)	Other Services Use Army Inventory (Y/N)
89	5	1	1	Y	N

Goals

Fiscal Year	Readiness Buildup	Inventory Protect	Training % Goal	TOA Activity_I (in millions)	TOA Activity_II (in millions)
89	180	90	100	2195	280.7
90	180	90	100	2115	345.5
91	180	90	100	2575	395.4
92	180	90	100	3007	530.2
93	180	90	100	3007	530.2

Figure 5.25.1-1. Example of JSM Simulation Guidelines Input Screen Filled with Information

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.25	3. PROGRAM NAME GUID 1.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.26 Program. RDAISA (Army Requirements) Input Program (PKG.FOR)

5.26.1 Program Description. Program to read an AAO tape file, compute Army requirements by package level, and load or update the REQTS_ARMY table (PKG). The PKG program is written in structured FORTRAN 77 with embedded SQL statements. The source code is in a file named PKG.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file -- PKG.F77 -- which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable run file -- PKG.RUN.

a. Function - The program reads an RDAISA AAO tape file, builds Army requirements by package level, and builds the PJSM ORACLE REQTS_ARMY table.

b. Input -

(1) From the command line:

(a) ORACLE user name.

(b) ORACLE password.

(2) From terminal or command stream:

(a) Beginning FY.

(b) Number of FYs.

(c) New AAO tape file (yes/no) -

(1) If Yes:

(a) Pathname of new AAO tape file.

(b) Name of an exception file.

(2) If No:

(a) Pathname of existing exception file.

(b) Pathname of new exception file.

(3) From the ICT table: A list of secondary items.

(4) From the ITEM table: DODIC.

(5) From the AAO tape file:

(a) SSN.

(b) FY.

(c) Nomenclature.

(d) Unit of measure.

(e) Test losses for each FY.

(f) Level 1 and Level 2 ammunition losses for each FY.

(g) AAIQ for NATO, Korea, RDF, other, and Pacific Reserve.

(h) Ammunition resupply quantities by theater (see 5.26.1b(5)(g)).

(i) Plant operations projects.

(j) Depot Level 1 requirement.

(k) Mobilization training A requirements.

(l) WRSA 0 - 30 and 31 - balance.

(m) Mobilization training B requirements.

c. Processing -

(1) The program checks to see if a valid ORACLE user name and password were supplied on the command line.

(2) The program reads in the values for the beginning FY and the number of FYs (usually five).

(3) The names of an input and exception file are read in and opened.

(4) If a new AAO file is being read, the REQTS_ARMY table is updated by setting the change column equal to minus quantity and quantity to zero.

(5) A list of secondary item DODICs is retrieved from the ICT. This list is sorted in ascending sequence so that an efficient binary search routine can be subsequently used to screen out secondary items from the AA0 file.

(6) The data are read in five records at a time. See Annex D for file layouts. For each group of records the following checks are performed:

(a) End of file.

(b) Read errors (data/format mismatch).

(c) FY out of range.

(d) SSN is not in the ITEM table.

(e) The item is a secondary item.

(7) The group of records is shipped for any condition 5.26.1c(6)(b) through 5.26.1c(6)(e).

(8) For each group of records that pass the above tests, the following steps are performed:

(a) Package levels 2 through 17 are computed as follows:

```
      AIIQ(19) = 0
      RESUPPLY30(6) = 0
      RESUPPLY45(5) = 0
      RESUPPLY60(5) = 0
      RESUPPLY90(5) = 0
      MOBTNGA(4) = 0
      MOBTNGB(3) = 0

      DO 6 I=1,18
    6 AIIQ(19) = AIIQ(19) + AIIQ(I)

      DO 7 I=1,5
    7 RESUPPLY30(6) = RESUPPLY30(6) + RESUPPLY30(I)

      DO 8 I=1,4
      RESUPPLY45(5) = RESUPPLY45(5) + RESUPPLY45(I)
      RESUPPLY60(5) = RESUPPLY60(5) + RESUPPLY60(I)
    8 RESUPPLY90(5) = RESUPPLY90(5) + RESUPPLY90(I)
```

```
      DO 9 I=1,3
    9 MOBTNGA(4) = MOBTNGA(4) + MOBTNGA(I)

      DO 10 I=1,2
   10 MOBTNGB(3) = MOBTNGB(3) + MOBTNGB(I)

      YEAR = FY$ = BFY$ + 1

      PKG 2 = TEST(YEAR)
      PKG 3 = AIIQ(19)
      PKG 4 = OPPROJ
      PKG 5 = L1AL(YEAR)

      If DEP1 .GT. 0
         PKG 6 = 0.55 * PKG 4
      Else
         PKG 6 = 0
      End if

      PKG  7 = RESUPPLY30(6)
      PKG  8 = RESUPPLY45(5)
      PKG  9 = RESUPPLY60(5)
      PKG 10 = L2AL(YEAR)

      If DEP2 .GT. 0
         PKG 11 = 0.55 * PKG 9
      Else
         PKG 11 = 0
      End if

      PKG 12 = MOBTNGA(4)
      PKG 13 = WRSA30
      PKG 14 = RESUPPLY90(5)
      PKG 15 = WRSABAL
      PKG 16 = MOBTNGB(3)
      PKG 17 = RESUPPLYBAL
```

(b) If there is a record in REQTS_ARMY that matches the DODIC, FY, and package, that record is updated as follows:

If quantity > 0, change = requirement - quantity.

Otherwise, change = change + requirement.

(c) If the record is not found in the REQTS_ARMY table, it is inserted with change = quantity.

5.26.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the ORACLE tables usually consist of the corresponding table column name with a 's' character added.

5.26.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.26.4 Error Conditions. Error messages will be printed in the PKG.COMO file.

5.26.5 Listings. The program listing contains comments to assist in making program changes. This program listing (PKG.LIS) is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.26	3. PROGRAM NAME PKG.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FORM 5752, NOV 78, WHICH IS OBSOLETE.

5.27 Program. ICAPP Unit Prices (Including Army's) and Other Customer Requirement Input Program (ICAPP.FOR)

5.27.1 Program Description. The ICAPP program is written in structured FORTRAN 77 with embedded SQL statements. The source code is in a file named ICAPP.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file -- ICAPP.F77 -- which is compiled with the PRIME F77 compiler. The PRIME BIND Linker utility is used to produce an executable run file -- ICAPP.RUN.

a. Functions - The program extracts requirements and unit price data from an ICAPP tape file and inserts these data into the PJSM ORACLE data base.

b. Input -

(1) From the command line:

(a) ORACLE username.

(b) ORACLE password.

(2) From terminal or command stream:

(a) FY of POM; e.g., 89.

(b) Beginning FY location in ICAPP file (1-8).

(c) Number of FYs (usually 5).

(d) New ICAPP tape file (yes or no):

(1) If yes -

(a) Pathname of new ICAPP file.

(b) Name of an exception file.

(2) If no -

(a) Name of an existing exeption file.

(b) Name of a new exception file.

(e) Update Army unit prices (yes or no).

(3) From the ICAPP tape file:

(a) DODIC.

(b) SSN.

(c) Service Code.

(d) Required quantities by FY.

(e) Unit prices for Army by FY.

c. Processing -

(1) The program checks to see if a valid ORACLE user name and password were supplied on the command line.

(2) A list of DODICs and a SSN/DODIC cross reference array are built from the data in the ITEM table. These arrays are sorted in ascending sequence to facilitate the subsequent use of a fast binary search routine.

(3) The terminal or command stream data are read in and checked.

(4) If a new ICAPP tape file is being processed, the change and quantity columns in the REQTS_OTHER table are zeroed out.

(5) The ICAPP file is opened and read one line at a time. For each line the following checks are performed:

(a) End of file.

(b) Data/format mismatch.

(c) The DODIC and SSN are checked against the arrays built in step 5.27.1c(2). If both SSN and DODIC are not found, the line is rejected.

(6) For each good line in the ICAPP file the following is performed:

(a) If the service code is 'AR' for Army, an attempt is made to insert a new row containing DODIC, FY and unit price into the ITEM_DATA table. If the insert fails because the record is already in the table, the unit price column is updated if this option is in effect.

(b) For service codes for other customers, an attempt is made to insert a new row into the REQTS_OTHER table. If the insert fails because of duplicate value in index, the quantity and unit price columns are set to the new values.

5.27.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a # character added.

5.27.3 Verification Procedures. The program was verified by inspection of the unit prices inserted into the ITEM_DATA and REQTS_OTHER tables.

5.27.4 Error Conditions. Error messages will be printed in the ICAPP.COMO file produced at each execution of the program.

5.27.5 Listings. The program listing contains comments to assist in making program changes. This listing (ICAPP.LIS) is located in <SYSSA>SAXPGM>#PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.27	3. PROGRAM NAME ICAPP.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.28 Program. Alternate Priority Scheme Generator (APS.FOR)

5.28.1 Program Description. The Alternate Priority Scheme Generator program is written in structured FORTRAN 77. The source code is in a file named APS.FOR which is input to the ORACLE precompiler that creates the APS.F77 file. This file in turn is compiled with the F77 compiler and loaded with the BIND utility to produce the APS.RUN run file. The following paragraphs describe the APS program:

a. Identification - Source code for this program is found in the APS.FOR file. Its run file equivalent is APS.RUN. APO users have access to this program only through the JSM Master Menu.

b. Functions - The APS program determines an alternate priority for each item in the ITEM.DATA table based on work-years calculated from the 1-8-5 production rate, 1-8-5 staffing rate, and the unit price. Alternate priorities will start at 1 for the highest non-zero work-year item and increment by 1. Zero work-year items are ordered by unit price and assigned large alternate priorities based on this ordering. The ITEM.DATA table is updated with the calculated alternate priority.

c. Input -

(1) Interactive. ORACLE identification and password.

(2) From data base accessed by program.

(a) From REVTAB: Beginning FY, number of years.

(b) From ITEM.DATA: Read list of DODICs and their unit prices, order this select by FY and unit price.

(c) From PRODT: For each item, read its 1-8-5 production and staffing rates.

(d) From PRODT.FY: If no data exists in PRODT for an item, read 1-8-5 production and staffing rates.

d. Processing Logic.

(1) User Input. Interactively read ORACLE user identification and password.

(2) Program Input.

(a) From ITEM.DATA table read list of DODICs and their unit prices ordered by FY and unit price.

(b) From REVTAB table, read beginning FY and number of FYs.

(c) From PRODT table or PRODT.FY table, read the associated 1-8-5 production and staffing rates.

(d) Calculate work-years based on the following equation:

Work-years = 1-8-5 Staffing Rate ÷ (1-8-5 Prod Rate * 12 * Unit Price)

(e) Determine the position of this item's calculated work-years with respect to other items; this procedure develops a pointer array with the value of the alternate priority for the item.

(f) Update the ITEM.DATA table with the determined alternate priorities. Lower alternate priorities imply larger computed work-years.

e. Output - Only update ALT.PRI column in ITEM.DATA table.

f. Security - No unique considerations.

g. Interfaces - This program requires no interface with other programs. It can be executed at will but may result in a differing order of alternate priorities due to data read from PRODT or PRODT.FY.

h. Tables - See Annex B.

5.28.2 Conventions. The program has been written with the convention that all local variables end with a '$' character to distinguish them from similarly named ORACLE columns.

5.28.3 Verification. Verification will necessarily be done manually using ORACLE's User Friendly Interface to directly access the data base.

5.28.4 Error Conditions. Errors encountered will be documented in the COMO file produced at each execution.

5.28.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.28	3. PROGRAM NAME APS.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.29 Program. Individual Item Data Report (IIDR.FOR).

5.29.1 Program Description. The Individual Item Data Report program is written in structured FORTRAN 77. The source code can be found in a file called IIDR.FOR. The following paragraphs describe the IIDR.FOR program:

a. Identification - Source code for the program is located in the IIDR.FOR file. This file is input to the ORACLE precompiler which builds an IIDR.F77 file. The IIDR.F77 file is compiled with the F77 compiler and its binary counterpart loaded using the BIND utility. The executable program will be named IIDR.RUN.

b. Functions - The IIDR.FOR program produces the Item Input Data Report which is a detailed report of input information for each item. Including production data, RAMP years data, and requirements by year and package level.

c. Input -

(1) Interactive. User inputs ORACLE identification and password and enters RCN.

(2) Input by Program.

(a) From REVTAB table: Read beginning FY and number of FYs.

(b) From PACKAGE table: Read package levels (PKGL) and package names.

(c) From ITEM table: Read FSC, DODIC, SSN, SEQ, NSN, ISN, Family, Use, NMF, Nomenclature, and UOM columns for use in report.

(d) From RAMP.ITEM table: For each DODIC read from Item, read in the assets for beginning FY.

(e) From PRODT table: Read line number, 1-8-5 and 2-8-5 production and staffing rates, and MPA columns.

(f) From PLANT table: Read plant name.

(g) From RAMP.PROD table: Read Plant, Line UNDEL_AR, UNDEL_TOT, PROD_ARMY, DV_ARMY, WRKYRS.AR, PROD_TOT, DV_TOTAL, and WRKYRS_TOT columns.

(h) From ITEM.DATA table: Read FY, unit price, PRI, and MIA columns.

(i) From REQTS.OTHER table: Read FY, service, unit price, and quantity.

(j) From REQTS.ARMY table: Read FY, package level, and quantity.

d. Processing -

(1) User enters ORACLE identification and password. User also enters desired RCN.

(2) Read in all data from ITEM.DATA table. Read RAMP.ITEM table for beginning assets for beginning FY by item.

(3) For each item read from ITEM.DATA table, do the following:

(a) Read from PRODT table 1-8-5 and 2-8-5 production and staffing rates and the maximum production allowed and write to output for each plant where the item is produced.

(b) Write item information (SSN, Family code, NSN, Beginning asset, DODAC, Use code, Nomenclature, Unit of Measure, New Materiel Fielding) to output.

(c) Read RAMP.PROD table for production, dollar value and undelivered amounts for Army and total for the 3 years prior to the beginning FY. Write these RAMP year data to the output report. If there are no data for a certain year, then write such on output.

(d) From ITEM.DATA table read the unit price, priority and maximum inventory allowed, by year starting with the beginning FY. Write these data to the output report by year.

(e) From REQTS.OTHER table read the service, unit price and requirement, by year. Write the service name and unit price by year to the output. The requirements are written next in a separate block with service and requirement by year.

(f) From REQTS.ARMY table read the requirement by FY and package level. Write the package number, package name, and Army requirements by year to output report.

(g) Loop through procedure for next item until end.

(h) Close report, end program.

(4) It should be noted that the output is formatted but the ^001^001 character at the top signals the printer for format control. No option on the SPOOL command is necessary.

e. Output - Item Input Data Report.

f. Interfaces - Users will only have access to the executable program or the output report through the JSM Master Menu. This program requires no other interface with any program.

g. Tables - See Annex B.

5.29.2 Conventions. The program has been written with the convention that all local variables end with a '$' character. This will help distinguish them from similarly named table columns.

5.29.3 Verification Procedures. Verification must done as using the ORACLE UFI utility to access the data base.

5.29.4 Error Conditions. Errors will be documented in the COMO file produced at each execution of the program.

5.29.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.29	3. PROGRAM NAME IIDR.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.30 Program. Plant Item Data Report (PIDR.FOR)

5.30.1 Program Description. The Plant Item Data Report is written in structured FORTRAN 77. The source code is located in a file called PIDR.FOR which is input to the ORACLE precompiler that creates the PIDR.F77. The PIDR.F77 is input to the F77 compiler and then loaded using the BIND utility producing the executable version PIDR.RUN. The following paragraphs describe the PIDR program:

a. Identification - Source code for this program is located in a file called PIDR.FOR. Its executable equivalent is PIDR.RUN. Functional users will have access to this program only through the JSM Master Menu.

b. Functions - The PIDR.RUN program produces the Plant Item Data Report which is a detailed list of production data for all items separated by plant. This information includes components by item.

c. Input -

(1) Interactive: User inputs ORACLE identification and password. Also inputs preferred revision control number.

(2) By program:

(a) From REVTAB table: Read beginning FY and number of FYs.

(b) From PLANT table: Read plant name, production overhead, and non-production overhead by plant.

(c) From ITEM table: Read in list of DODICs and nomenclature.

(d) From STAFF table: Read in direct labor and OMA by plant.

(e) From PRODT table: Read the line, 1-8-5 and 2-8-5 production and staffing rates, line availability, and maximum produced quantity.

(f) From ICT: Read procurement factor.

(g) From PROJECT: Read project title.

(h) From PRODT.FY table: Read the line, FY, month, and 1-8-5 and 2-8-5 production and staffing rates.

d. Processing -

(1) User enters ORACLE identification and password.

(2) User enters desired revision control number. Program reads beginning FY and number of fiscal years from REVTAB.

(3) Program reads in list of plants, plant names, and their production and non-production overhead factors from PLANT table.

(4) Program reads in list of items from ITEM table.

(5) Program reads from PRODT table the plants at which each item is produced.

(6) For each plant read from PLANT table, do the following:

(a) Read STAFF table for the direct labor and OMA by FY for the beginning FY through the ending FY, where the ending FY is equal to the beginning FY plus the number of FYs minus one.

(b) For each FY calculate the total production overhead, non-production overhead, and total labor according to the following equations:

Production Overhead = Direct labor * Production Overhead Factor + 5

Non-production Overhead = (Direct labor + OMA + Production Overhead) * Non-production Overhead Factor + 5

Total labor = Direct labor + OMA + Production Overhead + Non-production Overhead

(c) Write the plant name and the direct labor, OMA, production overhead, non-production overhead, and total labor by FY.

(d) Open a cursor to retrieve all items and item information from PRODT table for a plant.

(1) For each item read from PRODT read the nomenclature and beginning asset posture.

(2) Write SSN, FSC, DODIC, 1-8-5 and 2-8-5 production and staffing rates, line number, alternate items, maximum produced quantity, maximum production allowed, beginning assets, and nomenclature to output file.

(3) From the ICT table, read the list of components for this DODIC.

(4) For each component read, read its procurement factor and nomenclature from ITEM table.

(5) Write component information to output.

(6) Loop to new end item.

(e) Open a cursor to retrieve all project information from the PRODT.FY and PROJECT tables. Read the project title, line number, effective date (month and year), 1-8-5 and 2-8-5 production and staffing rates. Write project information to output and loop to new project.

(f) After all project data read, loop through to next plant.

(7) After all plants completed, close output report and end program.

e. Output - Plant Item Data Report.

f. Interfaces - This program requires no interfaces with any other programs. Users will have access to this program through the JSM Master Menu.

g. Tables - See Annex B.

5.30.2 Conventions. This program has been written with the convention that all local variables end with a '$' character to distinguish them from similarly named ORACLE columns.

5.30.3 Verification Procedures. Verification of the output will necessarily be done using ORACLE's UFI utility to directly access the data base.

5.30.4 Errors. Errors encountered will be documented in the COMO file produced at each execution.

5.30.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.30	3. PROGRAM NAME PIDR.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.31 Program. Data Extract for Ammunition Executive Management System (AEMS.FOR)

5.31.1 Program Description. The AEMS program is written in structured FORTRAN 77. Source code for the program is in the AEMS.FOR file which is input to the ORACLE precompiler that builds the AEMS.F77 file. This file is compiled with the F77 compiler and loaded using the BIND utility to produce the executable program AEMS.RUN. The following paragraphs describe the AEMS program:

a. Identification - Source code for the program is located in the AEMS.FOR file. Its executable equivalent is AEMS.RUN. The PMO and DCSRDA users have access to this program only through the JSM Master Menu.

b. Functions - The AEMS program reads production data and asset posture from the JSM data base for a given list of items and produces a formatted output report which is used for interfacing with DCSRDA's AEMS.

c. Input -

(1) Interactive:

(a) User inputs ORACLE identification and password.

(b) User inputs beginning FY and number of FYs.

(c) User enters filename of DODAC file containing the list of pertinent items.

(d) User enters output filename.

(e) Pathnames are supported for reading and writing files across the network.

(2) By program:

(a) From REQTS_ARMY table: Read FY, package level, and quantity.

(b) From REQTS_OTHER table: Read FY and quantity.

(c) From ITEM table: Read SSN, nomenclature, and unit of measure.

(d) From RAMP_ITEM table: Read ending assets.

(e) From ITEM_DATA table: Read the priority.

(f) From PRODT table: Read 1-8-5 production rate.

(g) From PRODT_FY table: Read 1-8-5 production rate.

d. Processing -

(1) User enters ORACLE identification and password.

(2) Program connects to ORACLE.

(3) Assume RCN equals to zero for any information selected from the data base.

(4) User enters the beginning FY and the number of FYs (only the last two digits of the FY are pertinent).

(5) User enters file name for input file. Program opens specific file if found.

(6) User enters output file name. Program opens specific file.

(7) Declare a cursor to retrieve all quantity information by FY and package level for a specific DODIC from REQTS_ARMY table.

(8) Declare a cursor to retrieve quantity by fiscal year from the REQTS_OTHER table for same DODIC as above.

(9) For each item read from the input file, do the following:

(a) Search ITEM table for SSN, nomenclature, and unit of measure for the item read; if the item is not found, write out a message to that effect and go to next item.

(b) Retrieve the END_ASSETS asset for previous year from the RAMP_ITEM table (this will be used as the beginning asset posture for the current year); if the DODIC is not found, write out a message and set beginning assets to zero.

(c) Divide beginning assets by 1,000 to get in thousands.

(d) Open the previously declared REQTS_ARMY cursor and retrieve the Army requirements for the DODIC across all packages and all years studied.

(e) Close the cursor.

(f) Open the previously declared REQTS_OTHER cursor and retrieve the other service requirements for the DODIC across all study years.

(g) Close the cursor.

(h) For each year studied, read the item's priority from the ITEM_DATA table and the 1-8-5 production rate from the PRODT table. The latter value may have been updated so the PRODT_FY table must be read, this value will be used if any exists in the PRODT_FY table.

(i) Write the gathered data to the output file.

(j) Loop to next item read.

(10) End report when end of file is reached.

(11) End the program.

e. Output - The output file is formatted. Please note the enclosed example for the required format.

f. Interfaces - This program only requires that the specified input file be accessible. It requires no interfaces with other programs.

g. Tables - See Annex B.

5.31.2 Conventions. This program was written with the convention that all local variables end with a '$' character to distinguish them from similarly named ORACLE columns.

5.31.3 Verification. Verification will necessarily be done using the ORACLE UFI utility to manually access the data base.

5.31.4 Error Conditions. Any errors will be documented in the COMO file produced at each execution. Note: They will not be documented in the output file.

5.31.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.31	AEMS.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.32 Program. Increased Workload Impact Report (IWIR.FOR)

5.32.1 Program Description. The Increased Workload Impact Report program is written in structured FORTRAN 77. Source code is found in the IWIR.FOR file which is input to the ORACLE precompiler that builds the IWIR.F77 file. This file is then compiled with the F77 compiler and loaded using the BIND utility to build the executable program IWIR.RUN. The following paragraphs describe the IWIR.FOR program:

a. Identification - Source code for this program is found in the IWIR.FOR file. Its executable equivalent is IWIR.RUN. Users will have access to this program only through the JSM Master Menu.

b. Functions - The IWIR program produces the Increased Workload Impact Report which details, by end item, the impact in work-years that would result by increasing production of each end item by 1,000 units.

c. Input -

(1) Interactive: User inputs ORACLE identification and password.

(2) By program:

(a) From PLANT table: Read in list of plant codes.

(b) From ICT table: Read in list of end items, also components and procurement factors.

(c) From ITEM table: Read in nomenclature for end items.

(d) From PRODT table: Read plant, line, 1-8-5 and 2-8-5 production and staffing rates, and line availability.

d. Processing -

(1) User enters ORACLE identification and password.

(2) Read in list of valid plant codes from PLANT table.

(3) Read in list of end items from ICT table.

(4) For each end-item do the following:

(a) Declare and open a cursor that will return the plant, line, 1-8-5 and 2-8-5 production and staffing rates, and line availability for each plant which produces the end item from PRODT. If no plants are retrieved, write a warning message to the screen and loop to the next end item.

(b) Check the plant just retrieved against the list of valid plants. If the plant is valid then continue; otherwise, write an error message and pass the current end item.

(c) Calculate work-years for this item according to the following formula:

Work-years = ((1-8-5 staffing ÷ (1-8-5 production * 12)) * 1000

(d) If the current plant is the first retrieved for the current end item, then write new page headings to output.

(e) Keep track of work-years by plant for the end item for subsequent processing.

(f) Write plant, line, staffing, and production data for this plant to the output.

(g) Loop, retrieve next plant for current end item. If no more are found, close the cursor and continue; otherwise, continue as in 5.32.1d(4)(b).

(h) The following pertain to component information. The sequence will be passed through twice; first time for primary components and second for secondary components.

(1) Declare and open a cursor that will retrieve component DODICs and their procurement factors pertinent to the current end item.

(2) Loop through this cursor keeping track of the component DODICs and the procurement factors.

(3) Close the cursor.

(4) For each component retrieved do the following:

(a) Select the nomenclature from the ITEM table.

(b) Select the plant, line, 1-8-5 and 2-8-5 production and staffing rates, and the line availability from the PRODT table.

(c) Check the plant just retrieved for validity. If the plant is not valid then write a warning message to the screen and pass this component.

(d) Calculate the work-years for this item by the following:

Work-years = (1-8-5 staffing ÷ (1-8-5 production * 12)) * (1000 * procurement factor)

(e) Keep track of the work-years by plant and item status; i.e., end item, primary component, secondary component, for subsequent processing.

(f) Write the component type, DODIC, nomenclature, and production data to the output.

(g) Declare and open a cursor that will retrieve all primary subcomponents and their procurement factors from the ICT table for the current component.

(h) For each primary subcomponent selected do the following:

(1) Select the nomenclature from the ITEM table.

(2) Select production data from the PRODT table.

(3) Check the plant selected for validity. If the plant is not valid, then generate a warning message to the screen and pass this plant.

(4) Compute work-years for the primary subcomponent according to the following:

Work-years = (1-8-5 staffing rate for primary component ÷ (1-8-5 staffing rate for primary component * 12)) * (1000 * procurement factor of current primary or secondary component * procurement factor of primary subcomponent)

(5) Keep track of work-years by plant and component type for subsequent processing.

(6) Write the production information to the output file.

(7) Loop for next primary subcomponent.

(1) Loop to next component (primary or secondary) and continue.

(5) Loop to secondary components and continue.

(1) Loop through the valid plants for the current end item. Keep track of the minimum and maximum work-years for this item. Write the plant name and the work-years associated with the primary component, secondary component, and primary subcomponents to the output. Keep track of total work-years for primary and secondary components for subsequent processing.

(j) Write the minimum and maximum plant work-years to the output.

(k) Write the primary component work-years and the secondary component work-years to the output.

(l) Loop to next end item and continue.

(5 This program is obviously recursive in nature but ORACLE does not allow recursion at this time; therefore, duplication of coding was required.

e. Output - Increased Workload Impact Report.

f. Interfaces - This program requires no special interfacing with any other program.

g. Tables - See Annex B.

5.32.2 Conventions. This program was written with the convention that local variables all end with a '$' character. This will help distinguish them from similarly named table columns. Also, no format control is required when the output is spooled. The output contains information that will signal the printer to look for format controls.

5.32.3 Verification Procedures. Verification must be done using the ORACLE UFI utility to access the data base.

5.32.4 Error Conditions. All errors will be documented in the COMO file produced at each execution of the program.

5.32.5 Listings. Program listing is located in <SYSSA>SAXPGM>*PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.32	3. PROGRAM NAME IWIR.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.33. Program. Ammunition Scheduling Model (PS.FOR)

5.33.1 Program Description. Program for generating an ammunition production schedule for a specified Program Objective Memorandum (POM) period of years. Production Scheduling (PS) is written in structured FORTRAN 77 with embedded SQL statements. The source code contains about 4,100 lines including comments and is in a file named PS.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file -- PS.F77 -- which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable run module -- PS.RUN.

a. Functions - The program reads information from a file produced by the BUILD_PTR program called PTR_ARRAYS, extracts item, production and requirements information from the DJSM ORACLE data base, produces an ammunition production schedule for a specified POM period, and writes detailed and summary results to the data base. The results in the data base are then available for all of the output report generators needed for an analysis of the ammunition schedule.

b. Input - The program requires a valid ORACLE user name password and RCN, and several guidance parameters related to the restart option. All other data are obtained from a file called PTR_ARRAYS and the ORACLE data base:

(1) From PTR_ARRAYS file: A list of DODICs, the relationship between end items and their primary components, procurement factors (PFs) of the primary components, relationship between end items and their secondary components, procurement factors of the secondary components, plant codes, production location (plant and line) of each item, and relationship between the primary components and the end items they are used on.

(2) From REVTAB table: Beginning FY, number of FYs, level of training, depot pipeline, asset indicator, and sharing indicator, where RCN is that specified by the user.

(3) From LINE table: Line number and plant code, ordered by line.

(4) From RAMP_ITEM table: DODIC and END-ASSETS, where FY = Beginning FY - 1, ordered by DODIC.

(5) From PRODT table: 1-8-5 production rate, 2-8-5 production rate, 1-8-5 direct labor staffing, 2-8-5 direct labor staffing, line availability, maximum production allowed, where DODIC = current DODIC being processed and line = current line number which the DODIC can be produced.

(6) From ITEM table: DODIC, new materiel fielding code (NMF), family code (FAMILY), and sub-family code (SUB_FAMILY), ordered by DODIC.

(7) From SERVICE table: Service codes.

(8) From ITEM_DATA table:

(a) UNIT_PRICE (for Army) and Maximum Inventory Allowed (MIA) where RCN is that specified by user or equal to zero, and DODIC and FY = current ones being processed.

(b) DODIC and POM_COST, where FY is the current one being processed, POM_COST is greater than zero, and RCN is either zero or the one specified by the user, order by DODIC and RCN.

(c) DODIC and PRI, where FY is the current one being processed, and RCN = 0, ordered by PRI.

(d) DODIC and ALT_PRI, where FY is the current one being processed and RCN = 0, ordered by ALT_PRI.

(9) From STAFF table: Number of direct labor employees and the percent of the direct labor employees which can be work-loaded, where RCN is either the one specified by the user or zero, and PLANT and FY equal to the current one being processed.

(10) From PRODT_FY table: DODIC, plant code, line number, production rates for a 1-8-5 and a 2-8-5 operation along with their respective direct labor staffing, month, availability of the line, and maximum production allowed, where FY is less than or equal to the current FY being processed, RCN = 0 or the one specified by the user, order by FY, DODIC, LINE, and RCN.

(11) From TREQ table: DODIC, draw down quantity, buildup quantity, other service requirement, Army test and training requirement, and total Army requirement, where RCN is the one specified by the user, and FY = the current one being processed, order by DODIC.

(12) From RAMP_PROD table: DODIC, plant code, line number and total undelivered quantity, where UNDEL_TOT > 0 and FY = the beginning FY - 1.

(13) From REQTS_OTHER table: DODIC, service code, unit price and required quantity, where FY is the current one being processed, order by DODIC and SERVICE.

(14) From REQTS_ARMY table: DODIC and required quantity, where FY and PKGL are the current ones being processed.

(15) From GOALS table: Activity_I dollar amount and percent of training which is applied to all items, where FY = the current one being processed and RCN is either 0 or the one specified by the user.

(16) From PFSTAB table: PKGL and the PDIP funding sequence, where RCN = 0 or the one specified by the user, order by PKGL and RCN.

(17) From RESULT4 table: Beginning assets for the restart option, DODIC and E_ASSETS (which are used as the beginning assets for the restart option), where RCN = the one specified by the user and FY = beginning FY - 1, order by DODIC.

(18) From RESULT1 table: DODIC and SFALL_QTY (which are used as the undelivered 'RAMP year' quantities for the restart option), where RCN = the one specified by the user, FY = beginning FY - 1, PKGL = 0, and SFALL_QTY > 0.

c. Processing - Since this program is fairly lengthy, this processing section is sub-divided into an overview section, a narrative of the decision flow process, a subroutine cross-reference section, and a detailed walk through the model.

(1) Overview - The following is a brief overview of the production scheduling model:

(a) The ORACLE user name and password, the RCN, and the restart flag are obtained from the command screen. If the restart flag is set to 'Y', then the beginning year for the study run, the number of years for the run, and the last FY with available requirements are also obtained from the command screen.

(b) From REVTAB study parameters are obtained, such as the beginning FY, number of FYs, level of training, depot pipeline, asset indicator, and sharing indicator.

(c) Any data related to the RCN are deleted from the RESULT tables.

(d) Initial data are read from the PTR_ARRYS and obtained from the ORACLE data base. These data include the items to be studied, the relationship between end items and their primary and secondary components, the beginning assets for each item, and the production data.

(e) For each year to be processed, the following steps are performed. After all steps are performed for that year, the next year's data are processed in the same manner. This continues until the number of FYs for this RCN are processed.

(1) Yearly data are updated and some arrays are initialized. From the ORACLE data base, each item's unit price is retrieved along with any maximum inventory constraints. The plant staffs, any changes to the production capabilities, and each item's draw down and buildup targets are also retrieved.

(2) The undelivered RAMP year quantities are scheduled for production at designated facilities along with the item's primary components. At each facility the amount of staff used and the amount of production capacity used are subtracted from the amount available. The detailed results for scheduling the undelivered RAMP year quantities are written to the RESULT1 table.

(3) The other customer requirements and the item's primary components are scheduled for production at the appropriate production facilities. As each item is scheduled, the staffing level and the amount of production capacity used are subtracted from the amounts available. The detailed results for scheduling the other customers quantities are written to the RESULT2 table.

(4) The Army requirements are now processed. Before processing each item's requirement, the Activity I dollars are retrieved, along with the percent of training which is applied to all items. The POM costs are obtained and subtracted from the Activity I dollars, yielding an amount which can be used to purchase additional hardware. The PDIP funding sequence and each item's relative priority are read.

(5) Each PDIP is processed in the specified PDIP funding sequence. Within each PDIP, the items are processed in the specified priority order. For each item, production is scheduled to satisfy its requirements and its related primary components. The requirements are computed and accumulated for its related secondary components. After processing all of the required items in the PDIP,

and before proceeding to the next PDIP, the accumulated requirements for the secondary items are scheduled for production. The detailed results for this PDIP are written to the RESULT1 table and then processing continues with the next PDIP.

(6) Once all of the PDIPs are processed for a FY, summary results are written to the RESULT3, RESULT4, RESULT5 and RESULT6 tables. Processing then resumes for the next FY.

(2) Narrative Description - A narrative description of the decision flow process embedded in the production scheduling model is presented in Annex C.

(3) Subroutine Cross-reference - Following is an alphabetical listing of the subroutines used in the production scheduling model program. Figure 5.33.1-1 depicts the interrelationship among these subroutines.

(a) AREQ. Processes the Army requirementss in a specified priority order within each PDIP, writes to RESULT1 table (via routines RFILE and RSLT1), accumulated data for other output tables (RESULT3 through RESULT6), and computes and accumulates the requirements for secondary items (prop charges, primers, and fuzes).

(b) BLDPRI. Builds the priority arrays used to establish the sequence in which items are processed within each PDIP.

(c) CRESLT. Clears the previous results for this RCN from the tables (RESULT1 through RESULT6).

(d) INIT. Reads data from the PTR_ARRAYS file and builds the pointer arrays used in this program. It also retrieves the beginning assets, production data, and family codes from the ORACLE tables.

(e) INVEN. Applies any available inventory to the Army items requirement.

(f) MATCH. Uses a binary search to locate and match the requested DODIC to one in the ITEM array. When a match is found, the ITEM_NO is returned.

(g) OCR. Processes the other customer requirements in DODIC order, accumulates the results via routine RFILE, writes the results to RESULT2 table, and accumulates data for other output products (RESULT3 through RESULT6).

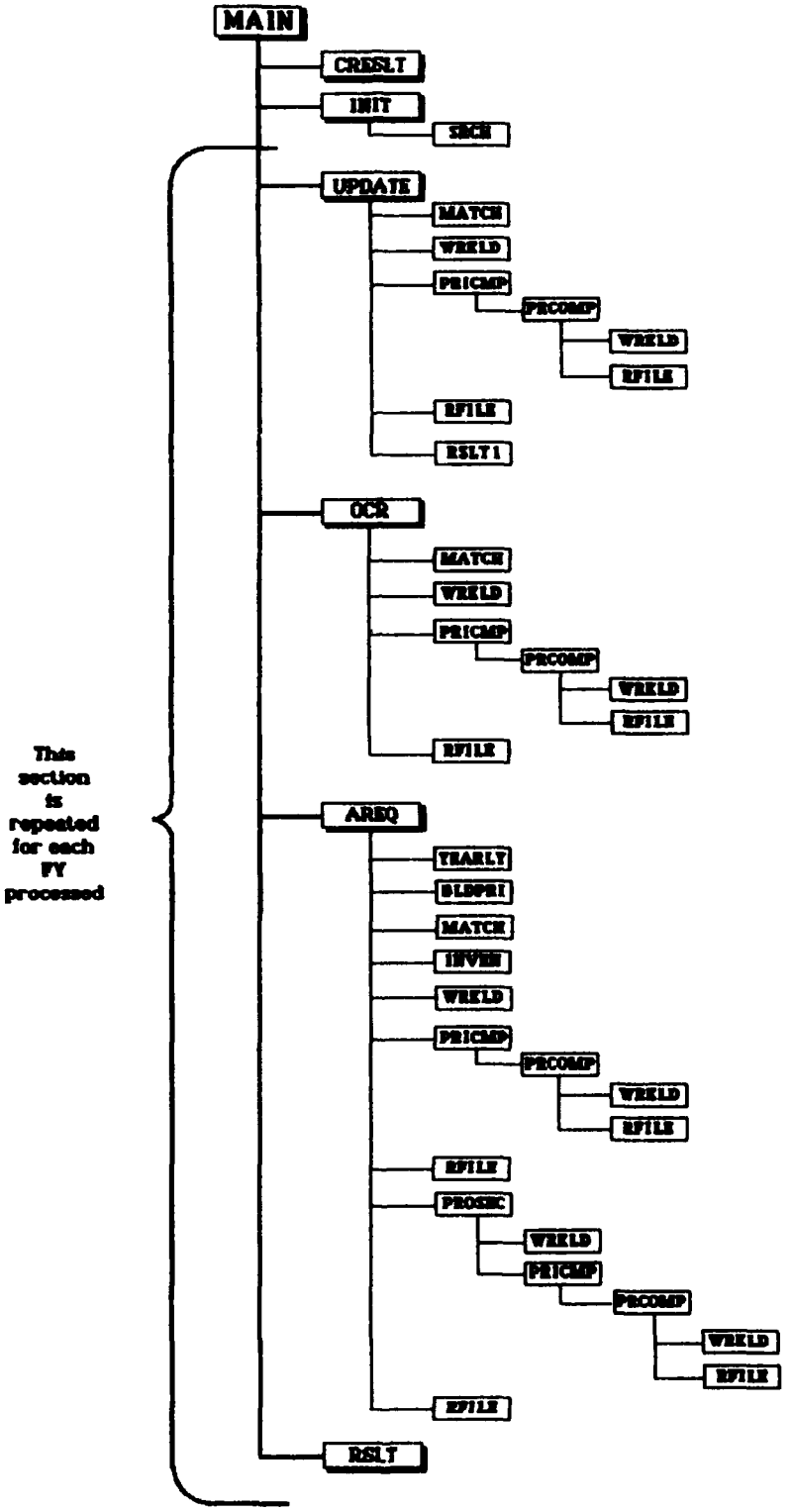


Figure 5.33.1-1. Subroutine Calls for the Production Scheduling Model

(h) PRCOMP. Processes each primary component which is passed to it by routine PRICMP, writes results to either RESULT1 or RESULT2 table via routines RFILE and RSLT1, and accumulates data for other output tables (RESULT3 through RESULT6).

(i) PRICMP. Selects the primary components for the required item, computes the component's requirements, and passes those requirements to the routine PRCOMP. This is done for the item's primary components as well as for the primary component's primary components.

(j) PROSEC. Processes the secondary component requirements, writes results to the RESULT1 table via routines RFILE and RSLT1, and accumulates data for other output tables (RESULT3 through RESULT6).

(k) RFILE. Records the transactions for RESULT1 and RESULT2 tables in temporary files. These transactions will be written to the tables via Routine RSLT1 after further processing in the program.

(l) RSLT1. Writes the accumulated results to RESULT1 table.

(m) RSLT. Prepares and writes the accumulated data to RESULT3 through RESULT6 tables.

(n) SRCH. Searches the item array in DODIC order, matches the requested DODIC, and then returns the ITEM_NO.

(o) WRKLD. Determines if there are available resources to produce a required item, and if so, schedules it for production and decrements the remaining resources appropriately.

(p) UPDATE. Updates the yearly data for unit prices, plants staffs, production capability, and each item's draw down and buildup targets. It also controls the processing and scheduling of the undelivered RAMP year quantities.

(q) YEARLY. Obtains TOA Activity I dollars, the POM costs for specified items, the PDIP funding sequence, and the percent of level I training to be funded for each item.

(4) Detailed Review - This section will provide a more detailed account of the process with reference to data arrays and subroutine calls.

(a) The ORACLE user name and password, the RCN, and the restart flag are obtained from the command stream. If the restart flag is set to 'Y', then the beginning year for the study run, the number of years for the run, and the last FY with available requirements are also obtained from the command stream.

(b) From REVTAB, the following values are extracted: Beginning FY, number of FYs, level of training requirements, depot pipeline indicator, asset indicator, and the sharing indicator.

(c) Routine CRESLT is called, which deletes from the result tables all data related to the RCN, which is greater than or equal to the beginning year for this study run. Upon completion, the operation returns to MAIN.

(d) Routine INIT is called, which reads the initial input data.

(e) From the PTR_ARRAYS file:

(1) Each item's DODIC is read and loaded into the ITEMS array.

(2) Each item's primary component and procurement factor are read and loaded into the PCOMP_PTR array (a pointer array), the PCOMP array (a data array which points to the item's DODIC in the ITEMS array and also contains a pointer to the next primary component), and the PFAC_PRIME array (a data array containing procurement factors.

(3) Each item's secondary component and procurement factor are read and loaded into the SCOMP_PTR array (a pointer array), the SCOMP array (a data array which points to the item's DODIC in the ITEMS array and also contains a pointer to the next secondary component), and the PFAC_SEC array (a data array containing procurement factors).

(4) The plant codes are read and loaded into the PLANT_CODE array.

(5) Each item's production location(s) is read and loaded into the IPL_PTR array (a pointer array), the PR_LINE array (an array which points to the next production location), and the IPL array (a data array containing the plant and line information).

(6) For each primary component, the list of end items which use it are read and loaded into END_ITEM_PTR array (a pointer array), and the END_ITEM array (a data array which points to the item's DODIC in the ITEMS array and also contains a pointer to the next end item).

(7) From LINE table, the line/plant combinations are extracted from the LINE table (in order of the line number) and loaded into the LPLANT array.

(8) The E_ASSETS array is now loaded with beginning assets from either the RAMP_ITEM table or the RESULT4 table. If the beginning FY of this run equals the beginning FY specified in the REVTAB table, then the DODIC and END_ASSETS are obtained from RAMP_ITEM where FY = beginning FY - 1, order by DODIC. Otherwise, the DODIC and E_ASSETS are obtained from RESULT4, where RCN = the one specified by the user and FY = beginning FY - 1, order by DODIC.

(9) From the PRODT table, production data (R185, R285, S185, S285, AVAIL, and maximum production allowed (MPA)) are extracted and loaded into the PROD_CAP array (a data array which contains the yearly production capability for both the first and the second shifts of operation), the STAFF_REQTS array (a data array which contains the staff required to support the production rates), and the MAXPA array (a data array which contains any MPA constraint for an item). For the first shift, the monthly production rate (R185) is multiplied by 12 and loaded into the PROD_CAP array, and its related staffing requirement is loaded into the STAFF_REQTS array. For the second shift, the monthly production rate (R285) is multiplied by 12, the yearly production for the first shift is subtracted, and the resultant is loaded into the PROD_CAP array. The people needed to support the second shift (S285 - S185) is also loaded into the STAFF_REQTS array.

(10) From the ITEM table, each item's NMF and family and subfamily codes are obtained and loaded into the NMF_CODE array and the FAM_CODE array. In order to load the FAM_CODE array, the FAMILY code is multiplied by 10 and added to the SUB_FAMILY; that result is loaded into the FAM_CODE array.

(11) From the SERVICE table, the service codes are read and loaded into the SERV array.

(12) Upon completion of the above, the operation returns to the MAIN.

(f) In the MAIN, a loop is set up whereby each FY is processed, 1 year at a time. Processing begins with the starting FY for the run, and is incremented by one after all the following steps have been performed.

(g) Routine UPDATE is called to read the data that can fluctuate each year and also process the undelivered RAMP year quantities.

(1) All arrays which contain yearly data are initialized to their appropriate values. For example, the RPCT_LINE array (the remaining percent of line array) and the RPCT_STAFF array (the remaining percent of staff at each plant) are initialized to 100 percent.

(2) From the ITEM_DATA table, for the FY being processed, each item's unit price and maximum inventory allowed (MIA) constraint are extracted and loaded into the UP array and the MAXIA array. This is one for RCN = the one specified by the user. If no data exists for that item, the data is extracted from the baseline (RCN = 0).

(3) From the STAFF table, the values for goal (percent) and direct labor are extracted for each plant. This is done for RCN = the one specified by the user. If no data exists for that RCN, then data is extracted from the baseline (RCN = 0). The value for GOAL is divided by 100 to obtain a percent. This percent is multiplied by DIRECT, with that product loaded into the PSTAFF array (the plant staff array).

(4) From the PRODT_FY table, values for production data and MPA which vary by year are obtained and used to update the data in the PROD_CAP array (production capability array), the STAFF_REQTS array, and MAXPA array (maximum production allowed array). For the DODIC and line that is being processed, if AVAIL = 0, then the DODIC is not allowed to be produced on that line and the production rates and staffing requirements are set to zero in the PROD_CAP and STAFF_REQTS arrays. Otherwise a variable RATIO is computed, setting it equal to (12. - M0 + 1.)/12. This ratio is a weighting factor, specifying a percent (portion of a year) to be applied to the new data read in from PRODT_FY. For the first shift, the PROD_CAP array is recomputed as (1-RATIO) times its current value plus the product of RATIO times 12. times R185. In like manner, STAFF_REQTS = its current value times (1-RATIO) plus (S285-S185) times RATIO, and STAFF_REQTS = its current value times (1.-RATIO) plus (S285-S185) times RATIO. Once the PRODT_FY table is

read, the IPL_PTR array is checked; if IPL_PTR for an item = 0 then RCODE (reason code) = 'H' for that item, meaning that it cannot be produced at any production facility.

(5) From the TREQ table, each item's summarized requirements are extracted. Values for the buildup quantity, draw down quantity, total other customer requirement, test and training requirement, and total Army requirement are loaded into the BUILDUP, DRAWDN, TOT_OC, TOT_TATR, and TOT_ARMY arrays, respectively.

(h) The undelivered RAMP year quantities are now processed. For the first year processed, if the FY being processed equals the beginning FY in REVTAB fro the specified RCN, the undelivered RAMP year quantities are obtained from the RAMP_PROD table, where UNDEL_TOT > 0 and FY = beginning FY. If the first year processed is for a restart where the first year is not equal to the beginning FY in REVTAB for the specified RCN, then DODIC and SFALL_QTY are obtained from RESULT4 where RCN = the one specified by the user, FY = beginning FY + 1, PKGL = 0, and SFALL_QTY > 0. For the remaining years of the study, the undelivered quantities which were not scheduled the first year are obtained from several internally stored arrays -- UDODIC (an array containing the DODIC which has undelivered quantities which need to be scheduled), UITM (an array containing an internally assigned item number), UPTR (a pointer array which points to the location of the production facility for that item), and UQTY (an array containing the quantity which needs to be scheduled for production).

(1) Routine WRKLD is called in an attempt to satisfy a requirement. Since routine WRKLD is called from eight different places throughout the model, it is addressed separately.

(2) If the required item was scheduled for production (via routine WRKLD), then routine PRICMP is called where production of the primary components is scheduled. Since routine PRICMP is called from several places throughout the model, it is also discussed separately.

(3) As each transaction is processed, routine RFILE is called in order to store the data. Again, this routine is called many times and is addressed separately.

(4) Once all undelivered quantities are processed for the FY, routine RSLT1 is called to write the data to the RESULT1 table.

(5) Upon completion of the above, the operation returns to the MAIN.

(i) In the MAIN, the next step is to process the other customer requirements. To do this, subroutine OCR is called.

(1) From the REQTS_OTHER table, each service's requirements for each item are obtained in alphabetical order and processed.

(2) If the other customers are allowed to use Army inventory in excess of the Army's draw down quantity, then that quantity is used to satisfy as much of the other customer's requirements as possible.

(3) Routine WRKLD is called in an attempt to satisfy the unfilled portion of the requirement. Since routine WRKLD is called from eight different places throughout the model, it is addressed separately.

(4) If the required item was scheduled for production (via routine WRKLD), then routine PRICMP is called where production of primary components is scheduled. Since routine PRICMP is called from several places throughout the model, it is also discussed separately.

(5) As each transaction is processed, routine RFILE is called in order to store the data. This routine is called many times and is addressed separately.

(6) Once all the other customer requirements are processed for the FY, the transactions are written to the RESULT2 table for the FY and RCN being processed. Before writing each transaction, the dollar value of the inventory is set equal to the quantity of inventory used times the unit price, and the dollar value of production is set equal to the quantity produced times the unit price. Upon completion of the other customer requirements, the operation returns to the MAIN.

(j) In the MAIN, the next step is to process the Army requirements. To do this, subroutine AREQ is called.

(1) When processing the Army requirement, first some data related to the FY must be processed. Those data are extracted via subroutine YEARLY.

(2) In YEARLY, the amount of money available (in millions) for Activity I and the percent of level I training to be used for all items is obtained from the GOALS table, where FY is the current one being processed and RCN is either the one specified by the user or, if not found, RCN = 0. The POM costs (in millions) are then obtained from the ITEM_DATA table, where FY = the current year being processed, POM_COST > 0, and the RCN in either the baseline (RCN = 0) or in the value specified by the user, ordered by DODIC and RCN. The POM_COST item is first checked to determine if it is for production base or for hardware; i.e., the FAM_CODE for that item is checked. If FAM_CODE = 73 (Family, subfamily 3), then it is for production base and those POM costs are not included. Otherwise, the POM costs are subtracted from the Activity I dollars, yielding the amount which can be allocated to buy the required ammunition.

(3) From the PFSTAB, the sequence in which the PDIPs are to be processed is obtained.

(4) From the ITEM_DATA table, the percent training values are obtained for each item which varies from the percent training value obtained from the GOALS table. The percent training values in the GOALS and ITEM_DATA table are in whole numbers and need to be divided by 100 to become actual percentages. The percent training values for each item are stored in an array called TRNG. Processing is then returned to the AREQ subroutine.

(5) Routine BLDPRI (abbreviation for Build Priorities) is then called in order to obtain the priority order by which to process the items.

(6) In BLDPRI, the priorities for processing the items are obtained from the ITEM_DATA table. The DCSOPS priorities are stored in an array called PRIORITY, while an alternate priority scheme is stored in an array called ALT_PRIORITY. Upon completion, the operation is returned to subroutine AREQ.

(7) In AREQ, the Army requirements are processed in the order that the PDIPs are to be funded. For each PDIP, the Army requirements are extracted for each item from the REQTS_ARMY table and loaded into an array called AREQTS.

(8) Within each PDIP, the items are processed in the priority sequence which was stored in the array called PRIORITY. The item's requirement is initially set equal to a variable called SFALL_QTY$, and as the requirement is filled, the SFALL_QTY (shortfall requirement) is reduced.

(9) If the PDIP being processed is number 10 or 11 and Level 2 training is not to be funded, then the item's requirement cannot be filled and a reason code = 'F' is assigned.

(10) If the pipeline (PKGL = 6 or 11) is to be filled over a 3-year period and the current year being processed is either the first or second year, then the portion of the pipeline which is not allowed to be filled is computed and stored in a local variable called EXTRA and a reason code = 'G' is assigned. When the first year is being processed, EXTRA = the required quantity times (1. - (90./200.)); otherwise when the second year is being processed, EXTRA = the required quantity times (1. - 145./200.)).

(11) If the item has assets in inventory, then subroutine INVEN is called. In INVEN, if the PDIP being processed is a test or training requirement (package number 2, 5, or 10), then any assets in excess of the item's draw down quantity are applied to the item's requirements. For any other PDIP, the available assets are applied to the requirement. If assets are applied, then the ending asset position and the draw down and buildup levels are adjusted. The operation is then transferred back to routine AREQ.

(12) The reason code array (RCODE) for the item is now checked to determine if processing should continue. No further processing for the item is required if RCODE = 'A' (lack of sufficient funds), 'B' (item has a production capacity constraint), 'C' (there is a staffing constraint at the plant(s) where the item is produced), 'D' (one of the item's primary components can not be produced), or 'H' (the item has no production facilities available).

(13) If the item's requirement was not filled with assets from inventory and the PDIP being processed is for a Days of Supply (DOS) requirement (PDIP number 3, 4, 6, 7, 8, 9, 11 - 17), the requirement is checked to see if it exceeds the buildup target. If it does, the requirement in excess of the buildup target is stored in a local variable called EXTRA and a reason code = 'E' is assigned.

(14) For the remaining requirement, a check is made to see if any funds are available for production. This check is made by comparing the variable TOA$ (which is the remaining production dollars available for the FY being processed) with the production cost of the item (which is the required quantity times the item's unit price). If funds are not available, then the requirement in excess of the available funds is stored in a local variable called EXTRA and a reason code = 'A' is assigned.

(15) For the remaining requirement, a check is made to see if the inventory growth of the item is restricted through the use of the maximum inventory allowed parameter (MAXIA). If it is restricted, then the requirement in excess of the MAXIA is stored in a local variable called EXTRA and a reason code = 'K' is assigned.

(16) Routine WRKLD is called in an attempt to satisfy the unfilled portion of the requirement. Since routine WRKLD is called from eight different places throughout the model, it is addressed separately.

(17) If the required item was scheduled for production (via routine WRKLD), then routine PRICMP is called in order to schedule production of primary components. Since routine PRICMP is called from several places throughout the model, it is also discussed separately.

(18) The item's unfilled requirement, which was stored in the local variable EXTRA, is now added back to the variable SFALL_QTY#. The dollar value of the inventory applied is computed as the quantity of inventory applied times the item's unit price. The dollar value of the production is computed as the quantity produced times the item's unit price. The remaining production dollars (TOA#) are adjusted by subtracting off the item's production cost.

(19) If the required item was either scheduled for production or had inventory applied, then requirements for each of the item's secondary components are computed and accumulated in an array called SREQTS. For each of the item's secondary components, the secondary component's requirements are increased by the quantity of the end item produced or applied out of inventory, times a procurement factor.

(20) As each transaction is processed, routine RFILE is called in order to store the data. Again, this routine is called many times and is addressed separately.

(21) Once all of the Army requirements for this PDIP are processed, then the routine PROSEC is called in order to process the secondary item requirements.

(22) In PROSEC, the Army's secondary item requirements are processed in the order that they are stored in the SREQTS array.

(23) The secondary item's requirement is first filled with assets from inventory if any are available.

(24) If the secondary item's requirement was not filled with assets from inventory, a check is made to see if any funds are available for production. This check is made by comparing the variable TOA# (which is the remaining production dollars available for the FY being processed) with the production cost of the secondary item (which is the required quantity times the secondary item's unit price). If funds are not available, then the requirement in excess of the available funds is stored in a local variable called EXTRA and a reason code = 'A' is assigned.

(25) Routine WRKLD is called in an attempt to satisfy the unfilled portion of the secondary item's requirement. Since routine WRKLD is called from eight different places throughout the model, it is addressed separately.

(26) If the secondary item was scheduled for production (via routine WRKLD), then routine PRICMP is called in order to schedule production of primary components. Since routine PRICMP is called from several places throughout the model, it is also discussed separately.

(27) The secondary item's unfilled requirement, which was stored in the local variable EXTRA, is now added back to the variable SFALL_QTY#. The dollar value of the inventory applied is computed as the quantity applied times its unit price. The dollar value of the production is computed as the quantity produced times the unit price. The remaining production dollars (TOA#) are adjusted by subtracting off the item's production cost.

(28) As each transaction is processed, routine RFILE is called in order to store the data. Again, this routine is called many times and is addressed separately. After processing all of the secondary item requirements for this PDIP, the operation returns to routine AREQ.

(29) In AREQ, once the processing is completed for this PDIP, subroutine RSLT1 is called where the transactions are written to the RESULT1 table.

(30) In routine RSLT1, for each transaction that was stored (via routine RFILE) the data are retrieved and put in local variables. The ending asset position, stored in an array called E_ASSETS, is adjusted accordingly and inserted into the RESULT1 table. Before writing the transaction the following

computations are made: the percent of the line utilized (PCT_LINE$) = the percent used times 100 (in order to show percent as a whole number), the dollar value of inventory applied (INV_DV$) = inventory applied times its unit price, and dollar value of production (PROD_DV$) = quantity produced times its unit price. Operation is then returned to routine AREQ.

(31) Once all the Army requirements are processed in each of the PDIPs, the operation is returned to MAIN.

(k) In the MAIN, the next step is to process and write summary data to the RESULT2 through RESULT6 tables. To do this, subroutine RSLT is called.

(1) In routine RSLT, data are first prepared for and written to the RESULT3 table, which is a summary of the production of each item at each production facility.

(2) For each item, the amount produced is retrieved from the PRODL array (data was stored via routine WRKLD). The percent of the line utilized, the work-years used, and the dollar value of the production are computed. The PCT_LINE_AR$ is set equal to the total quantity produced for Army (PROD_TATR$ + PROD_ADOS$) divided by the production capacity based on a 1-8-5 shift operation. The PCT_LINE_OT$ is set equal to the quantity produced for other customers (PROD_OTHER$) divided by the same 1-8-5 production capacity. After the work-years are computed, these percent line values are multiplied by 100 to make them whole numbers. The WRKYRS_AR$ and WRKYRS_OT$ are computed by multiplying their respective percent line utilized times the staffing requirements for a 1-8-5 shift operation. The dollar value of production for Army (DV_ARMY$) and for other customers (DV_OTHER$) are obtained from DVITMLN where that data was stored. These data are accumulated in the SUMRY6 array which will be used to write the RESULT6 table. Data are then inserted into the RESULT3 table.

(3) Data are now prepared for and written to the RESULT4 table, which is a summary of each item, showing mainly the item's ending asset position and the unfilled requirements.

(4) For each item, data are retrieved from the E_ASSETS, INV_USED, TOT_OC, TOT_ARMY, and SHORT arrays. The percent of the item's requirement that was satisfied is computed as 100 times (1. - shortfall quantity/required quantity); this was done for each of the three categories: other customers, Army test and training, and Army days-of-supply The total produced for the Army is set equal to the sum of what was produced for Army test and training and Army days-of-supply. These data are then inserted into the RESULT4 table.

(5) Data are now prepared for and written to the RESULT5 table, which is a summary, by each plant, of the work-years and dollars scheduled for each service and for each of the Army's PDIPs.

(6) For each plant and for each of the Army's PDIPs, data are retrieved from the SMRY5A array. The data for the prior year undelivered production are assigned package level zero. The data are then inserted into the RESULT5 table.

(7) Then for each plant and for each service, data are returned from the SMRY5B array and inserted into the RESULT5 table.

(8) Data are now prepared for and written to the RESULT6 table, which is a summary of the production activity on each line at each plant.

(9) For each line at each plant, data are retrieved from the array called SUMRY6 and inserted into the RESULT6 table. Operation is then returned to the MAIN.

(l) The program ends when all processing has been completed for each of the FYs.

(m) Subroutine WRKLD is called from many places throughout the model and is documented here. Subroutine WRKLD is given a requirement for an item, and its intended purpose is to schedule production of that item at some production location.

(1) For each production location, the remaining percent of the line and the remaining available staff is checked. If none are available, the next production location is selected.

(2) If an Army requirement is being processed, the maximum production allowed (MAXPA) constraint is checked to see if production is restricted. If so, then the requirement in excess of the MAXPA is stored in a local variable called EXTRA and a reason code = 'J' is assigned.

(3) The unscheduled plant staff which could be available to produce the item as well as the staff required to support the unscheduled portion of shift 1 and 2 of the line is computed. The unscheduled plant staff is equal to the plant staff times the remaining percent of that staff which has not been workloaded. The staff required to support the unscheduled portions of shift 1 and 2 for that item are computed as the required staff to support the production line times the remaining percent of the line which has not been scheduled. This is done separately for each shift, and will indicate if there is a possible staff constraint. Based on these data, a ratio is determined for each shift (R1 and R2), indicating the portion of each shift which is available for use. Then the unconstrained production for the first and second shifts (UP1 and UP2) are computed by multiplying the production capacity for that item by the remaining percent of the line which has not been scheduled. Finally, the constrained production for each shift (CP1 and CP2) are computed by multiplying the unconstrained production (UP1 and UP2) by the appropriate ratios (R1 and R2). The unconstrained production for the first and second shifts are computed, after which the constrained production is computed.

(4) If the required amount is less than the constrained production, the item is scheduled on that line. However, if only a portion is scheduled then it is determined if the shortfall was due to a staffing constraint or a production capacity constraint, and a reason code of 'C' or 'B' is assigned, respectively.

(5) If the item was scheduled for production, then the percentages of the line and the plant staff still available are recomputed. If the amount scheduled for production (AMT_PROD) is less than CP1, adjustments are made to the remaining percent of line (RPCT_LINE) by subtracting the percent of the first shift used to produce AMT_PROD, and the remaining percent of staff available by subtracting the percent of the staff used to support the production quantity AMT_PROD. If the amount produced (AMT_PROD) is greater than CP1, then similar adjustments must be made for both shifts.

(6) Data are accumulated for the RESULT5 table by storing information in arrays called SMRY5A and SMRY5B. Note that the dollar values accumulated are calculated by multiplying the amount produced by the item's unit price.

(7) Data are also accumulated for the RESULT3 and 4 tables by storing it in the arrays called PRODL, DVITMLN, and PROD.

(8) If the item being processed is a prior year undelivered item, the operation is now returned to subroutine UPDATE.

(9) If the item's requirement is still unfilled, the next production location for this item is considered. After all production locations have been tried or if the requirement has been filled, the operation is returned to the location in the model which called this subroutine.

(n) Subroutine PRICMP is called from many places throughout the model and is documented here. Subroutine PRICMP is given the amount produced for an item and its intended purpose is to select the primary components for the required item, compute the component's requirement, and pass those requirements to subroutine PRCOMP. This is done for the item's primary components as well as for the primary subcomponents.

(1) Each time a primary component is selected and its requirement is computed, that information is passed to subroutine PRCOMP.

(2) In PRCOMP, the primary component's requirement is first filled with assets from inventory if any are available.

(3) If the primary component's requirement was not filled with assets from inventory, then routine WRKLD is called in an attempt to satisfy the unfilled portion of the requirement. Since routine WRKLD is called from many places throughout the model, it is addressed separately.

(4) If the primary component still has unfilled requirements, it is designated a 'pacer' which will restrict the production of all items which use it as a primary component. All items associated with a pacer are assigned a reason code = 'D' so that the next time those items are processed with requirements, a shortfall record would automatically be written.

(5) As each transaction is processed, routine RFILE is called in order to store the data. Again, this routine is called many times throughout the model and is addressed separately. After processing the primary component, the operation returns to the routine PRICMP.

(6) If any of the primary components for the required item were pacing components, then the already scheduled production of the required item must be decreased, as well as all the internal arrays in the model which are impacted by this decrease. If there were no pacers, operation returns to the location in the model which called this routine.

(7) If there was a pacing component, then the variable SFALL_END contains the quantity which must be subtracted from the end items production quantity. The following adjustments are made to the end item: production (PROD_APP#) is decreased by SFALL_END, while the item's shortfall quantity (SFALL_QTY#) is increased by the same amount. The work-years are adjusted by multiplying work-years by the ratio of the adjusted production quantity to the original production quantity. The percent line is adjusted in the same manner. Arrays containing production data are adjusted by subtracting the value of SFALL_END. Since many arrays keep values by item/line combinations, a variable called ADJUSTMENT is computed. ADJUSTMENT equals SFALL_END if the amount produced on that line is large enough to cover SFALL_QTY, else ADJUSTMENT is equal to the quantity produced on that line.

(8) Based on the variable ADJUSTMENT, SMRY5A and SMRY5B arrays are modified; the dollar value of production is decreased by the product of ADJUSTMENT and unit price, while the work-years are decreased by the product of staff requirements and the ratio of ADJUSTMENT to the production capacity of the 1-8-5 shift operation. The remaining percent of staff is increased by the product of the staff requirements for that line and the ratio of ADJUSTMENT to the production capacity of the 1-8-5 shift operation. After completing the adjustments, operation returns to the location in the model which called this routine.

(o) Subroutine RFILE is called from many places throughout the model and is documented here. Subroutine RFILE is given the data for each transaction for the purpose of storing it in internal arrays. Later on these data will be retrieved and written to the RESULT1 and RESULT2 tables.

(1) A printer array called ARY_PTR is used as an entry point into the storage arrays for each item. When a transaction is given to subroutine RFILE, the ARY_PTR array is checked to see if that item had a previous transaction. If that transaction was for the same item and for the same service, then the previous transaction is updated. Otherwise, the new transaction is added to the storage arrays.

(2) After the transaction is stored, the operation returns to the point in the model which called routine RFILE.

5.33.2 Conditions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '$' character added.

5.33.3 Verification Procedures. The program can be verified by analyzing some of the output (use the report generators to obtain the reports via access through the PJSM Main Menu). In addition, one could spot check some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.33.4 Error Conditions. Any error messages will be printed to the file called MODEL.COMO, found in SAXJSM>$OUT.

5.33.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PS. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION	1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.	30 Nov 87

2. PROGRAM ID	3. PROGRAM NAME
5.33	PS.FOR

4. REV NO./DATE	5. DESCRIPTION OF REVISION

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.34 Program. Plant Job Scheduling Model Post Processor (JSMPP)

5.34.1 Program Description. JSMPP is an acronym for the Job Scheduling Model Post Processor. The JSMPP program is written in structured FORTRAN 77 with embedded SQL statements. It is used to modify the output of the PJSM.

a. Identification - The source code for JSMPP is in five source files -- JSMPP.FOR, CUTBAK.F77, CUTCMP.F77, PLUSUP.F77, and PLSCMP.F77. JSMPP.FOR must be precompiled using the ORACLE FORTRAN Host Language Precompiler (PCC) which replaces the embedded SQL statements with calls to library routines. The output of the precompiler is an intermediate file -- JSMPP.F77. The other F77 source files do not contain embedded SQL statements, thus, these files do not need to be precompiled. The F77 source files are compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable run module -- JSMPP.RUN. A Command Procedure Language (CPL) procedure -- BIND3IX.CPL -- is available which precompiles, compiles, links, and runs the program. This procedure automatically recompiles any of the source files that have been modified and relinks the program. If none of the source files were modified since the program was last linked, BIND3IX.CPL just runs the program. The five source files contain a total of 35 subroutines as shown in the following table:

SOURCE FILE/SUBROUTINE CROSS REFERENCE TABLE

JSMPP.FOR	CUTBAK.F77	CUTCMP.F77	PLUSUP.F77	PLSCMP.F77
MAIN	CUTBK1	CUTCP1	PLSUP1	PLSCMP
INPUT	CUTINV	CUTCP3	PLSINV	CHECKP
RESULT1	CUTBK3	CUTCP4	PLSUP3	CHGCMP
RESULT3	CUTBK4	CUTCP5	PLSUP4	
RESULT4	CUTBK5	CUTCP6	PLSUP5	
RESULT5	CUTBK6		PLSUP6	
RESULT6	BLIST			
R1UPDT	SLIST			
R3UPDT	SLIST			
R4UPDT				
R5UPDT				
R6UPDT				

b. Functions - The JSMPP is used to allow changes to be made to the Army production schedules produced by the PJSM. The JSMPP essentially reads tentative end item production or dollar value changes that the user has entered into the PJSM RESULT4 table, determines which changes are valid, and makes the appropriate end

item and corresponding component changes in the PJSM result tables -- RESULT1, RESULT3, RESULT4, RESULT5, and RESULT6. The program runs the same way regardless of how the tentative changes were entered into the RESULT4 table. They can be entered with the Change Screen (see paragraph 5.57), with a FORTRAN program, or by the ORACLE SQLPLUS commands.

c. Input - The program requires a valid ORACLE user name, password, and a RCN. These are entered on a command line like this: R JSMPP USER PSWD RCN; e.g., R JSMPP TRIER xxx 30. All other data elements are obtained from the PJSM data base. The RCN is used to restrict the rows that are accessed. When reading STAFF, GOALS, PFSTAB, PRODT_FY, and ITEM_DATA tables, the default values are read in first. These are the rows where RCN = 0. Then, any rows where the RCN matches the RCN on the command line are retrieved, and the data elements from these rows override or augment the default values. In REVTAB, TREQ, and all RESULT tables, only the row(s) where the RCN matches the one on the command line are ever accessed. The columns that are accessed are as follows:

(1) From REVTAB: Beginning FY (BFY), number of FYs, level of training, and depot training requirement. BFY is used to make sure that whenever FY is selected from any table, only those rows where FY is in the 5 year range (BFY to BFY+4) are accessed.

(2) From STAFF: Plant, FY, and the number of direct labor employees.

(3) From GOALS: The TOA for hardware for each year of the POM period.

(4) From PFSTAB: Package levels and the sequence in which they are funded.

(5) From ICT: A list of end items, their primary components, and the procurement factors. Only rows where type = 'P' are retrieved.

(6) From PRODT: DODIC, plant, line, the 1-8-5 and 2-8-5 production rates, 1-8-5 and 2-8-5 direct labor staffing, and line availability.

(7) From PRODT_FY: The same information as the PRODT table plus FY. Any data in PRODT_FY for a particular RCN overrides the same data from the PRODT table.

(8) From ITEM_DATA: DODIC, FY, RCN, unit price, priority, draw down and build up targets in DOS, and the maximum inventory allowed.

(9) From TREQ: DODIC, FY, draw down quantity, and the build up quantity.

(10) From RESULT1: DODIC, FY, package level, type, pacer, inventory applied, inventory dollar value, production applied, production dollar value, work-years used, percent of line capacity used, shortfall quantity, and reason code. Rows where package level is less than 1 are not selected.

(11) From RESULT3: DODIC, plant, line, FY, type, other service production, Army production for test and training, Army production for DOS, dollar value of Army production, work-years used for Army production, percent of line capacity used for Army production, dollar value of other service production, work-years used for other service production, and percent of line capacity used for other service production.

(12) From RESULT4: DODIC, FY, type, inventory used for test and training, inventory used for DOS, ending assets, percent of the test and training requirement filled, percent of the DOS requirement filled, test and training shortfall, DOS shortfall, total Army production, dollar value of Army prpduction, production change, dollar value change, and cumulative change. Only rows where type is 'E', 'S', or 'P' are selected.

(13) From RESULT5: FY, plant, package level, service code, dollar value of production, and work-years used. Only rows where PKGL > 0 and service = 'AR' are retrieved.

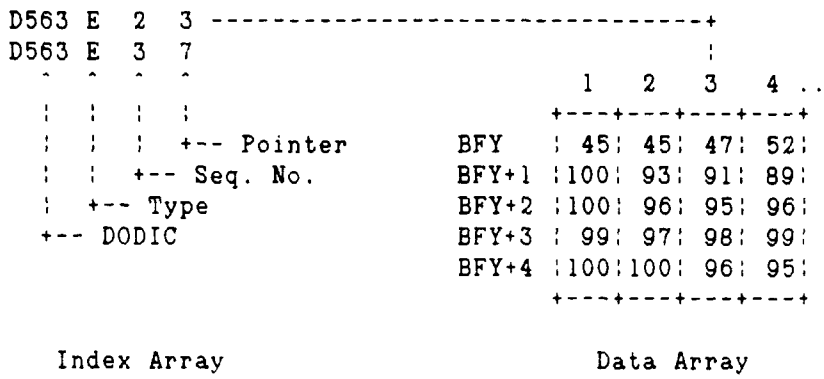
(14) From RESULT6: FY, plant, line, dollar value of production, work-years used, and percent of line capacity used.

d. Processing - This section is sub-divided into an overview section and a detailed walk through the Post Processor.

(1) Overview - The following is a brief overview of the Production Scheduling Post Processor:

(a) All required data elements listed in 5.34.1c through 5.34.1c(14) are read from the PJSM input and result tables. These data elements are loaded into several arrays and dynamic data structures. Each data structure consists of an index array and one or more data arrays. Each index entry consists of one or more data elements that form a unique concatenated key and a pointer to the

columns in the data arrays where other data elements are stored. Each data array has five rows -- one for each fiscal year in the POM. A typical data structure is depicted by the following illustration:



(b) The items that are being decreased are processed first to free up funds and production capacity which may be needed for the items that will be increased. For each item that is being decreased, the following steps are performed:

(1) The RESULT1 data structure is searched in reverse package funding sequence order for the lowest priority package level that has production applied. If more than one package level is needed, the program will reduce production applied to zero for the current package level before going to the next one.

(2) The plant/line combination is determined by making a three way match in the RESULT1, RESULT3, and RESULT5 data structures. The DODIC must match in RESULT1 and RESULT3 data; the plant must match in RESULT3 and RESULT5 data; and the package level must match in RESULT1 and RESULT5 data. Failure to obtain this match indicates a discrepancy in the result tables, and the Post Processor does not attempt to resolve it. If the entire change cannot be accommodated by the first plant, the program will look for additional plants.

(3) All RESULT data structures are checked for consistent production applied and associated dollar values. If there are any discrepancies, the package level change is reduced to prevent production applied or dollar value from going negative in any of the data structures.

(4) The appropriate changes are made in the result data structures for both end items and their primary components. Each of these data structures contains a flags array for

keeping track of the data items that have changed. The appropriate flags are set whenever one or more of the corresponding data elements is changed.

(c) Production increases for individual items is more involved than decreases because there are more constraints. A primary component could be a pacer. Additionally, the Post Processor considers the following to be constraints that cannot be exceeded:

(1) TOA dollar cap.

(2) Build up target.

(3) Maximum direct labor at a plant.

(4) Line capacity.

(d) For each item that is being increased the following steps are performed:

(1) The RESULT1 data structure is searched in package funding sequence order for the highest priority package level that has a shortfall. If more than one package level is needed, the program will reduce the shortfall to zero for the current package level before going to the next one.

(2) All available plant/line combinations that are not already in the RESULT3 data structure are added.

(3) If there is no entry in the RESULT5 data structure for the plant and package level, a new entry is added.

(4) All constraints on the end item production increase are tested, and the increase is cut back if necessary.

(5) The program loops through all primary components to determine associated component changes, using the appropriate procurement factors and checks the constraints. If any of the components is determined to be a pacer, the end item increase is restricted accordingly.

(6) The result data structures are updated the same way as for decreases.

(e) After all changes have been internally processed, the program searches through the all of the result flags arrays to determine which rows in the PJSM result tables need to be updated or inserted. Existing rows in the PJSM tables are updated

by setting selected columns equal to the the latest value from the corresponding data elements in the data structures. If new rows have to be inserted, zero is inserted into all columns related to other services.

(2) Detailed Walk Through - This section provides a more detailed description of Post Processor structure, functions, and control logic:

(a) The Post Processor gets all the information it needs before any processing is done. Each data element is read only once from the PJSM data base regardless of how many times it is used in the program. Data input is accomplished in the following steps:

(1) The ORACLE user name and password, and RCN are obtained from the command line.

(2) In the main program, the beginning FY, number of FYs, level of training requirements, and the depot training requirement are retrieved from the REVTAB table for the appropriate RCN.

(3) In subroutine INPUT, the columns listed in 5.34.1c(2) through 5.34.1c(9) are retrieved from PJSM STAFF, GOALS, PFSTAB, ICT, PRODT, PRODT_FY, ITEM_DATA, and TREQ tables. The data elements are loaded into arrays and data structures.

(4) Subroutine RESULT4 reads all rows and the columns listed in 5.34.1c(12) from the PJSM RESULT4 table for the appropriate RCN. Only rows where type = 'E' or 'S' are retrieved. The dollars value of Army production are summed to provide a total for comparison with the TOA constraint. The PROD_CHG and DV_CHG columns are checked for non-zero entries. If the PROD_CHG column is non-zero but the DV_CHG is zero, the correct DV_CHG is computed using the appropriate unit price. Likewise, a DV_CHG could be entered, and the corresponding PROD_CHG computed. Data elements for items subject to change, in one or more years, are loaded into the RESULT4 data structure. A second pass is made through the RESULT4 table to pick up the primary components. If a primary component is a component for one or more of the end items already in the RESULT4 data structure, the data elements for the component are also added to the RESULT4 data structure. The RESULT4 data structure consists of several data arrays -- R4PROD_TATR, R4PROD_ADOS, R4AINV_TATR, R4AINV_ADOS, R4SFALL_TATR, R4SFALL_ADOS, R4PROD_ARMY, R4DV_ARMY, R4E_ASSETS, R4PROD_CHG, R4DV_CHG, R4CUM_CHG, R4PFILL_TATR, R4PFILL_ADOS, and an index array. Each index entry consists of a

DODIC and a pointer to the columns in the data arrays where other data elements are stored. Each data array has five rows, one for each FY in the POM cycle.

(5) Subroutine RESULT1 reads all rows from the PJSM RESULT1 table for the appropriate RCN and checks for end item and component DODICs which have been entered into the RESULT4 index array. Data elements listed in 5.34.1c(10) for the items subject to change are loaded into the RESULT1 data structure. Each element in the RESULT1 index array consists of a DODIC, type, and a SCN where SCN = SEQ(PKGL#) to access the data in package funding sequence order or SCN = 18 - SEQ(PKGL#) to reverse the package funding sequence order. An array -- PKGL -- is used to convert the SCN back to the package level; e.g., PKGL# = PKGL(SCN). The index array is initially built in reverse package funding sequence order because production decreases are processed first and package levels must be backed out in reverse order.

(6) Subroutine RESULT3 reads all rows from the PJSM RESULT3 table for the appropriate RCN and checks for end item and component DODICs which have been entered into the RESULT4 index array. The data elements listed in 5.34.1c(11) into the RESULT3 data structure for the items that are subject to change. Also, the total production for test and training (R4PROD_TATR) and days of supply (R4PROD_ADOS) are updated in the RESULT4 data structure. These totals are accumulated here because there are no PROD_TATR or PROD_ADOS columns in the RESULT4 table.

(7) Subroutine RESULT5 reads all rows and columns listed in 5.34.1c(13) from PJSM RESULT5 table for the appropriate RCN and loads the data elements into the RESULT5 data structure.

(8) Subroutine RESULT6 reads all rows and columns listed in 5.34.1c(14) from PJSM RESULT6 table for the appropriate RCN and loads the data elements into the RESULT6 data structure.

(b) After all input and result tables have been read, control returns to subroutine MAIN and the program disconnects from the ORACLE system. The arrays and data structures contain all the required data elements, and they are several orders of magnitude more accessible than the corresponding ORACLE tables; thus, the program processes all the changes internally without interfacing with the ORACLE system. Production decreases are processed first to free up funds and production capacity that may be needed later for increases. Items are processed in the priority order assigned by the DCSOPS. Because these priorities can change

with FY, each year is processed separately. This is accomplished by building a temporary end item index array that contains just those items that are subject to change in a given FY. Each element in this index array consists of the priority, DODIC, type, and a pointer into the RESULT4 data structure. Each DODIC in the RESULT4 data structure where PROD_CHG is negative in a given FY is loaded into the end item index array. The priority of each item is obtained from the ITEM_DATA data structure. The type and the pointer are copied from the RESULT4 index array. The end item index array is sorted by priority, DODIC, and type so the items will be processed in this order. The production decreases for each item in the end item index array are then processed as follows:

(1) Subroutine CUTBK1 determines which package level(s) will be reduced or eliminated completely. This is done by looping through the RESULT1 data structure in reverse package funding sequence order to find the lowest priority package level that has production applied. If there is not enough production in a package, it is reduced to zero before going to the next lowest priority package level. The package level loops until either the reduction passed from MAIN has been fully accommodated or the package levels are exhausted. For each package level, the following steps are performed:

(a) The production change in the package (P_CHG_PKL) is determined. This is the smaller of the production applied in the RESULT1 data structure or the amount still needed to complete the tentative change passed from MAIN.

(b) The package level change is passed to subroutine CUTBK3 to check for compatible production and associated dollar values in the other RESULT data structures and to process the changes in the RESULT3 data structure.

(c) The production applied, shortfall, and the dollar value of the production are updated in the RESULT1 data structure.

(d) If the package level is for DOS, CUTINV is called to reduce the inventory that can be applied in the following years.

(e) Subroutine CUTCP1 is called to process the corresponding primary component production decreases.

(2) In subroutine CUTBK3 one or more plant/line combinations are determined by looping through the combinations in the RESULT3 index array. For each plant from the RESULT3 index,

the RESULT5 index array is checked to determine if there is a matching plant and package level. Depending on the package level, a check is made on either test and training or DOS production applied in the RESULT3 data structure to determine if it is compatable with the value passed from CUTBK1. CUTBK3 then calls CUTBK4, CUTBK5, and CUTBK6 to make compatability checks in the RESULT4, RESULT5, and RESULT6 data structures respectively. Finally, the changes in the production applied for test and training or days of supply, the dollar value of Army production, work-years for Army production, and percent line utilization for Army production are made in the RESULT3 data structure. The plant/line loop continues until either the package level change is fully accommodated or the plant/line combinations in the RESULT3 index array are exhausted.

(3) Subroutine CUTBK4 determines if the appropriate test and training or DOS production applied is at least as large as the change passed from CUTBK3. After additional data checking in subroutines CUTBK5 and CUTBK6, CUTBK4 is called a second time to make the appropriate changes to the production applied and shortfall for test and training or days of supply, ending assets, percent of requirement filled, total Army production, production change, cumulative change, and dollar value of Army production in the RESULT4 data structure.

(4) Subroutine CUTBK5 checks the RESULT5 data structure to determine if there is sufficient dollar value of production for the plant and package level. After additional compatability checking in subroutine CUTBK6, CUTBK5 is called a second time to update the dollar value of production and the work-years used in the RESULT5 data structure.

(5) Subroutine CUTBK6 checks the dollar value of production in the RESULT6 data structure for the plant and line passed from CUTBK3. CUTBK6 is called a second time to update the dollar value of production, work-years used, and the percent line used in the RESULT6 data structure.

(6) Subroutine CUTINV is called from CUTBK1 if the package level is for DOS. CUTINV reduces, if possible, the inventory applied in the years following the current year being processed. For each of the following years, CUTINV searches through the RESULT1 data structure for package levels that have inventory applied. The inventory applied is then reduced and the shortfall is increased in both the RESULT1 and RESULT4 data structures. The package levels where the inventory applied is reduced will not necessarily be the same as the package levels where production was decreased in a previous year. Also, the ending assets, and the percent of the test and training and days of supply requirements

filled are updated in the RESULT4 data structure. Inventory reductions in test and training packages are not carried over to the next year.

(7) Subroutine CUTCP1 determines which components must be processed by searching the item/component array. The component change quantities are determined by the end item change and the procurement factor for each component. If there is not enough component production to reduce, the inventory applied is reduced. The package level is determined by the associated end item. The same data elements are affected as when reducing end item production, except no dollar values are attached to component production. The production applied, inventory applied, and the shortfall quantity are updated in the RESULT1 data structure. Then CUTCP1 calls subroutine CUTCP3.

(8) Subroutine CUTCP3 performs the following functions for each component and Package level determined by the end item. A plant/line combination is obtained from the RESULT3 index array. The RESULT5 index array is then checked to determine if there is an entry for the plant and package level. Each production decrease which can be accomplished within a given package level and plant/line combination is processed separately. Depending upon the package level, the appropriate test and training or DOS production is reduced in the RESULT3 data structure, and the work-years and percent line are updated. CUTCP3 then calls subroutines CUTCP4, CUTCP5, and CUTCP6 to process corresponding RESULT4, RESULT5, and RESULT6 data structures respectively. The plant/line loop continues until either the plant/line combinations are exhausted or the reduction is fully achieved.

(9) Subroutine CUTCP4 updates the RESULT4 data structure for the component being processed. Depending on the package level the appropriate test and training or DOS production and/or inventory applied are reduced in the RESULT4 data structure. If the production applied and the inventory applied do not fully accommodate the decrease, the difference is added to the ending assets (R4E_ASSETS).

(10) Subroutine CUTCP5 updates the work-years in the RESULT5 data structure for a given plant and package level.

(11) Subroutine CUTCP6 updates the work-years and percent line used in the RESULT6 data structure for the appropriate plant and line.

(c) After all decreases have been processed for all FYs, the increases are processed the same way in the main program except the RESULT4 data structure is searched for positive production changes. First, however, the RESULT1 index array is modified by replacing each of the SCNs with 18 - SCN and resorting the array so the data will be indexed in package funding sequence order. (The order was reversed before.) The PKGL and the FSEQ arrays are also modified to provide for the correct conversions of SCN to package level and vice versa. For each FY, production increases for each DODIC in the end item index array are performing in the following steps:

(1) The total dollars value is compared with the TOA. If the total dollar value reaches the TOA, all remaining items will not be processed for the current FY.

(2) Subroutine PLSUP1 determines which package level(s) will be increased. This is done by looping through the RESULT1 data structure in package funding sequence order to find the highest priority package level that has a shortfall. If there is not enough shortfall in a package, it is reduced to zero before going to the next highest priority package level. For each package level the following steps are performed:

(a) The production change in the package (P_CHG_PKL) is determined. This is the smaller of the shortfall in the RESULT1 data structure or the amount still needed to fully accommodate the increase passed from MAIN.

(b) If the package level is for DOS, the build up constraint is checked.

(c) The package level increase is passed to subroutine PLSUP3 to check constraints and data compatibility, and to update the other data structures.

(d) Production applied, production dollar value, and shortfall values are updated in the in the RESULT1 data structure.

(e) If the package level is for DOS, PLSINV is called to process the additional inventory that can be applied in subsequent years.

(3) In PLSUP1, the package level loop continues until one of the following happens:

(a) One or more package levels have fully accommodated the increase passed from MAIN.

(b) A primary component has been determined to be a pacer.

(c) A constraint or an incompatibility between two or more of the RESULT tables has prevented the shortfall in a package to be completely processed.

(d) The package levels in the RESULT1 data structure have been exhausted.

(4) Subroutine PLSUP3 is called from PLSUP1 for each package level. Any available plant/line combinations that are not already in the RESULT3 data structure are added. The program then loops through plant/line combinations in the RESULT3 index array. If there is no entry in the RESULT5 data structure for the plant and package level, a new entry is added. The total work-years at the plant is compared with the direct labor at the plant. If there are not enough unused work-years, the production increase is reduced accordingly. PLSUP3 then calls subroutines PLSUP4 and PLSUP6 where additional constraints on the end item increase are checked. Subroutine PLSCMP is then called to check for primary component constraints and to process the changes for the components. Depending on the package level, the appropriate test and training or DOS production is increased in the RESULT3 data structure. Also, the dollar value of production is increased in the RESULT3 data structure, and the work-years and percent line used are incremented in both the RESULT1 and RESULT3 data structures. Finally, PLSUP3 calls subroutines PLSUP4, PLSUP5, and PLSUP6 to update the other data structures. The plant/line loop continues until one of the following happens:

(a) One or more plant/line combinations have been identified that can fully accommodate the package level production increase.

(b) A primary component has been found to be a pacer. If this happens, there is no need to consider any other plant/line combinations or any other package levels.

(c) Available plant/line combinations are exhausted.

(5) In subroutine PLSUP4, depending upon the package level the appropriate test and training or DOS shortfall is checked in the RESULT4 data structure. If this is less than the

production increase passed from PLSUP3, the increase is restricted accordingly. After additional constraint checking on the end item change in subroutine PLSUP6 and on the primary component changes in subroutine PLSCMP, PLSUP4 is called again to update the appropriate test and training or DOS production, shortfall, and the percent of the requirement filled in the RESULT4 data structure. Also, the ending assets are updated if the package level is for DOS, and the total Army production, total dollar value for Army production, the production change, and the cumulative change data items are updated in the RESULT4 data structure.

(6) Subroutine PLSUP5 updates the dollar value of production and, if possible, the work-years in the RESULT5 data structure for a given plant and package level.

(7) In subroutine PLSUP6, if there is no entry in the RESULT6 data structure for a given plant and line, a new entry is added. Checks are made to determine is there is sufficient line capacity remaining for the line and enough direct labor available at the plant to accommodate a given production increase. If either of these is a constraint, the production increase is reduced accordingly. After checking for primary component pacers, PLSUP6 is called a second time to update the dollar value of production, work-years, and the percent of line used in the RESULT6 data structure.

(8) PLSINV is called from PLSUP1 if the Package level is for DOS. PLSINV increases, if possible, the inventory applied in the years following the current year being processed. For each of the following years, PLSINV searches through the RESULT1 data structure for package levels that have a shortfall. The inventory applied is then increased and the shortfall is reduced in both the RESULT1 and RESULT4 data structures. The package levels where the inventory applied is increased will not necessarily be the same as the package levels where production was increased in a previous year.

(9) Subroutine PLSCMP is called from subroutine PLSUP3 for each package level and plant/line combination under consideration. PLSCMP checks the item/component array to determine if any primary components exist for the item. If primary components exist, each corresponding component change is determined by the end item change and the appropriate procurement factor. The component change will be made by taking from excess ending assets (R4E_ASSETS) in the RESULT4 data structure if possible. Otherwise, subroutine CHECKP is called to determine if the change can be accomplished by increased component production. If not, the component is

a pacer and the end item production increase is restricted accordingly. For each component subroutine CHGCMP is called to actually make the necessary changes in the RESULT data structures.

(10) In subroutine CHECKP all available plant/line combinations that are not already in the RESULT3 data structure are added. CHECKP then selects the first plant/line combination from the RESULT3 data structure and checks the total work-years used at the plant and the percent of the line already utilized. These quantities are compared with the direct labor available at the plant and the allowable 250 percent line utilization, respectively. Additional production possible for the plant/line is determined by either the unused work-years at the plant or the unused line capacity, whichever is most restrictive. Additional plant/line combinations are considered if necessary. If there is not enough unused capacity at all available plant/line combinations, the component is a pacer and the end item production increase is reduced accordingly.

(11) Subroutine CHGCMP goes through exactly the same plant/line combinations as CHECKP. However, instead of checking constraints, it updates all of the RESULT data structures.

(d) After all of the changes have been internally processed, the Post Processor reconnects to ORACLE to put the accumulated changes back into the PJSM result tables. The RESULT1, RESULT3, and RESULT4 tables are updated first because these contain the DODIC. The program calls the following subroutines for each DODIC in the RESULT4 index array.

(1) Subroutine R1UPDT updates the appropriate rows in the PJSM RESULT1 table. For each element in the RESULT1 data structure flags array that has been set to 'Y', the corresponding row in the RESULT1 table is updated. If the row is not found, a new entry is inserted. Each column is set to the corresponding value in the RESULT1 data structure.

(2) Subroutine R3UPDT updates the appropriate rows in the PJSM RESULT3 table. It loops through all plant/line combinations and FYs for each DODIC and updates or inserts new rows as required.

(3) Subroutine R4UPDT searches the RESULT4 flags array for all FYs for a given DODIC to determine which elements have been set to 'Y'. The corresponding rows in the PJSM RESULT4 table are updated. Whenever a Select for Update does not find an existing row, a new row is inserted.

(e) The remaining tables -- RESULT5 and RESULT6 -- which do not contain DODIC are updated by the following subroutines:

(1) Subroutine R5UPDT is called only one time. It searches the RESULT5 flags array for all plant/package level combinations and all FYs for entries that have been modified. The corresponding rows in the PJSM RESULT5 table are updated or inserted as appropriate.

(2) Subroutine R6UPDT, is also called only one time. It updates or inserts appropriate rows in the PJSM RESULT6 table, for all plant/line combinations and all FYs where a change has been made or a new combination was added.

(f) After all Army production changes are processed and the output tables have been appropriately changed, the modified Army Ammunition Program can then be viewed via screens, reports, and graphics.

e. Output - In addition to the updated PJSM result tables, the Post Processor produces three COMO files -- JSMPP.COMO, CUTCMP.COMO, and PLSCMP.COMO. JSMPP.COMO contains step-by-step details of the end item processing, and it will contain any error messages and warnings. CUTCMP.COMO and PLSCMP.COMO contain details of the processing of the primary component decreases and increases, respectively.

f. Security - The Post Processor processed information that is unclassified. A valid ORACLE username and password are required to run the program.

g. Interfaces - The Post Processor interfaces with the ORACLE Relational Data Base System through the FORTRAN Host Language Interface. The user must have access to ORACLE system and the PJSM data base to precompile and/or run the program.

5.34.2 Conventions. The program is written in structured FORTRAN 77. Variables which accept values directly from an ORACLE table usually consist of the corresponding table column name with a '$' sign appended; e.g., DODIC$, FY$... Data arrays that are used to store data retrieved from the PJSM RESULT tables consist of an 'R' followed by the table number then the column name; e.g., R4PROD_CHG is used to store the data items from the PROD_CHG column of the RESULT4 table.

5.34.3 Verification Procedures. The Post Processor received a thorough shakedown during the 4 month long Ammunition Production Base Management Study. In this study, the Post Processor was used to process hundreds of changes to several PJSM runs so the results would agree with various predetermined Army ammunition programs.

5.34.4 Error Conditions. Error messages are displayed in the JSMPP.COMO file.

5.34.5 Listings. Program listings are located in <SYSSA>SAXPGM>*PS.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.34	3. PROGRAM NAME JSMPP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FORM 5752, NOV 78, WHICH IS OBSOLETE.

5.35 Program. Build Pointer Arrays Program (For PS and PPJSM) (BUILD_PTR.FOR)

5.35.1 Program Description. BUILD_PTR.FOR is written in structured FORTRAN 77. The source code is in a file named BUILD_PTR.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file BUILD_PTR.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program BUILD_PTR.RUN.

a. Identification - Source code is located in the BUILD_PTR.FOR file which is under SAXPGM>#PS. Its executable equivalent is BUILD_PTR.RUN.

b. Functions - The program extracts information from the PJSM ORACLE data base and produces the PTR_ARRAYS file. This file displays all the data necessary for the pointer array used in the production scheduling program (PS.FOR) and the post processor program (PPJSM). The file displays the DODICs of all the items in the data base, each item's primary components and their related procurement factor, each item's secondary components (primers, prop charges, and fuzes), the plants in the data base, all plant/line combinations where each item can be produced, and a list of all end items associated with a primary component.

c. Input - The program requires a valid ORACLE user name and password. All other data are obtained from the ORACLE data base:

(1) From ITEM table: DODICs.

(2) From ICT table: For each DODIC, obtain the DODICs of all primary and secondary components and their related procurement factors.

(3) From PLANT table: Plant codes.

(4) From PRODT table: For each DODIC, obtain the plant code and the line number where the item can be produced.

(5) From PRODT_FY table: For each DODIC obtain the plant code and the line number. Order by DODIC and Line.

d. Processing -

(1) DODICs are extracted in alphabetical order from the ITEM table and stored in the ITEMS array.

(2) The number of items retrieved and a list of all the DODICs are written to the output file PTR_ARRAYS.

(3) Primary component data are obtained from the ICT and stored in a dynamic data structure consisting of two data arrays (PCOMP and PFAC_PRIME) and a pointer array (PCOMP_PTR). The PCOMP_PTR array is used to point to the location in the PCOMP array of the DODIC's first primary component. If the value in the PCOMP_PTR array equals zero, then the item does not have any primary components. The PCOMP array contains two rows; the first row indicates the location in the ITEM array of primary components DODIC (its procurement factor is in the PFAC_PRIME array), while the second row points to the location in the PCOMP array which contains the item's next primary component if the second row equals zero, then there are no more primary components.

(4) The PCOMP_PTR array; the maximum pointer in the PCOMP array; the PCOMP array, and the PFAC_PRIME array are written to the PTR_ARRAYS file.

(5) Secondary component data are extracted from the ICT and stored in a dynamic data structure consisting of two data arrays (SCOMP and PFAC_SEC) and a pointer array (SCOMP_PTR). The SCOMP_PTR array is used to point to the location in the SCOMP array of the DODIC's first secondary component. If the value in the SCOMP_PTR array equals zero, then the item does not have any secondary components. The PCOMP array contains two rows; the first row indicates the location in the ITEM array of the secondary component's DODIC (its procurement factor is in the PFAC_SEC array), while the second row points to the location in the SCOMP array which contains the item's next secondary component. If the second row equals zero, then there are no more secondary components.

(6) The SCOMP_PTR array, the maximum pointer in the SCOMP array, the SCOMP array, and the PFAC_SEC array are written to the PTR_ARRAYS file.

(7) The plant codes are retrieved from the PLANT table and stored in the PLANT_CODE array.

(8) The number of plants and the list of plant codes are written to the PTR_ARRAYS file.

(9) The available production locations for each item are obtained from the PRODT table and stored in a dynamic data structure consisting of a data array (IPL) and two pointer arrays (IPL_PTR and PR_LINE). The IPL_PTR array is used to point to the location in the IPL array of the DODIC's first production location, showing the plant and the line. If the value in the IPL_PTR array equals zero, then the item does not have any designated production location. Using the same pointer from the IPL_PTR array, one can enter the PR_LINE array to determine the location in the IPL array which contains the item's next production location. When the value in the PR_LINE array equals zero, there are no more production locations for that item.

(10) From PRODT_FY, any additional production locations are extracted and added to the arrays specificed in the above paragraph.

(11) The IPL_PTR array, the maximum pointer in the IPL array, the IPL array, and the PR_LINE array are written to the PTR_ARRAYS file.

(12) For each primary component, a list of end items which use it are retrieved from the ITC and stored in a dynamic data structure consisting of a data array (END_ITEM) and a pointer array (END_ITEM_PTR). The END_ITEM_PTR array is used to point to the location in the END_ITEM array of the first end item which uses the primary component. If the value in the END_ITEM_PTR array equals zero, then the item is not used as a primary component on any end item. The END_ITEM array has two rows; the first row indicates the location in the ITEM array of the end item's DODIC, while the second row points to the location in the END_ITEM array which contains the next end item which uses that primary component. If the second row equals zero, there are no more end items which use that primary component.

(13) The END_ITEM_PTR, the maximum pointer in the END_ITEM array, and the END_ITEM array are written to the PTR_ARRAYS file.

5.35.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '#' character added. The name of the output file is PTR_ARRAYS.

5.35.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.35.4 Error Conditions. Error messages will be printed to a file called MODEL.COMO.

5.35.5 Listings. The program listing contains comments to assist in making program changes. The program listing (BUILD_PTR.LIS) is located in <SYSSA>SAXPGM>#PS. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.35	BUILD PTR.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.36 Program. Requirement Summarization Program (TREQ.FOR)

5.36.1 Program Description. The TREQ program is written in structured FORTRAN 77 with embedded SQL statements. The source code is in a file named TREQ.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file TREQ.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable run file TREQ.RUN.

a. Function - The program reads data from the ORACLE JSM data base and builds the total requirement (TREQ) table. Army requirements from the REQTS_ARMY table are accumulated in accordance with the guidance parameters obtained from the REVTAB and GOALS tables. Other customer requirement quantities are read from the REQTS_OTHER table, summed over service, and added to the TREQ table.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, level of training, and the depot requirements parameter.

(2) From GOALS table: FY, draw down, and buildup goals in days of supply.

(3) From REQTS_ARMY table: DODIC, FY, package level, and required quantity.

(4) From ITEM_DATA table: Draw down and buildup target levels in days of supply.

(5) From ITEM table: New materiel fielding code.

(6) From REQTS_OTHER table: DODIC, FY, service code, and required quantity.

c. Processing -

(1) Guidance parameters are obtained from REVTAB and GOALS tables and stored in FORTRAN variables and arrays.

(2) All old rows with the same RCN as the one being run are deleted from the TREQ table.

(3) Requirements data are extracted from the REQTS_ARMY table by PKGL for each DODIC/FY combination.

(4) For each DODIC/FY combination the following is performed; computational details are given in the program listing.

(a) Package levels 6 and 11 are recomputed as follows:

```
If PKG 6 > 0
   If Depot requirement = 3
       If YEAR = Beginning FY
           PKG 6 = PKG 6 * (90./200.)
       Else if YEAR = Beginning FY + 1
           PKG 6 = PKG 6 * (145./200.)
       Else
           PKG 6 = PKG 6
       End if
   End if
End if

If PKG 11 > 0
   If Depot requirement = 3
       If YEAR = Beginning FY
           PKG 11 = PKG 11 * (90./200.)
       Else if YEAR = Beginning FY + 1
           PKG 11 = PKG 11 * (145./200.)
       Else
           PKG 11 = PKG 11
       End if
   End if
End if
```

(b) Army test and training requirement is computed as follows:

```
If Level of Training = 1
   Army test and training = PKG 2 + PKG 5
Else
   Army test and training = PKG 2 + PKG 5 + PKG 10
End if
```

(c) The total Army requirement is computed. This is the sum of PKG 2 through PKG 17.

(d) The draw down and buildup targets in Days of Supply (DOS) are obtained from the ITEM_DATA table if this information is there; otherwise, default values from the GOALS table are used.

(e) Target quantities are computed using the DOS targets obtained in 5.36.1c(4)(d). Details of this conversion follow:

```
If Level of Training = 1
   PROTECT = PKG 3 + PKG 4 + PKG 6
Else
   PROTECT = PKG 3 + PKG 4 + PKG 6 + PKG 11
End if

TEMP1 = PKG 7 + PKG 8
TEMP2 = TEMP1 + PKG 9
TEMP3 = TEMP2 + PKG 12 + PKG 13
TEMP4 = TEMP3 + PKG 14
TEMP5 = TEMP4 + PKG 15 + PKG 16
TEMP6 = TEMP5 + PKG 17

If DOS < 30
   QTY = PROTECT + (DOS / 30.0) * PKG 7
Else if DOS = 30
   QTY = PROTECT + PKG 7
Else if DOS < 45
   QTY = PROTECT + PKG 7 + ((DOS - 30.0) / 15.0) * PKG 8

Else if DOS = 45
   QTY = PROTECT + TEMP1
Else if DOS < 60
   QTY = PROTECT + TEMP1 + ((DOS - 45.0) / 15.0) * PKG 9

Else if DOS = 60
   QTY = PROTECT + TEMP2
Else if DOS < 90
   QTY = PROTECT + TEMP3 + ((DOS - 60.0) / 30.0) * PKG 14

Else if DOS = 90
   QTY = PROTECT + TEMP4
Else if DOS < 180
   QTY = PROTECT +TEMP5 + ((DOS - 90.0) / 90.0) * PKG 17

Else if DOS = 180
  QTY = PROTECT + TEMP6
End if
```

(f) The new materiel fielding code for the DODIC is retrieved from the ITEM table. If the item is a new materiel fielding item, the new materiel fielding quantity is computed; otherwise, it is set to zero. The new materiel fielding quantity is the sum of PKG 3 through PKG 7.

(g) A new record containing all the quantities computed in 5.36.1c(4)(b) to 5.36.1c(4)(f) is inserted into the TREQ table. The OTHER_REQ column is set to zero.

(5) Other customer requirements quantities are obtained from the REQTS_OTHER table and summed over service.

(6) For each DODIC/FY combination the following is performed:

(a) A check is made to determine if the DODIC/FY combination was inserted into the TREQ table when the Army requirements were loaded.

(b) If the row exists, the OTHER_REQ column is updated.

(c) If the row does not exist, a new row is added with all Army requirements columns set to zero.

5.36.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the ORACLE tables usually consist of the corresponding table name with a '$' character added.

5.36.3 Verification Procedures. The program can be verified by comparing the TREQ with the source tables REQTS_ARMY and REQTS_OTHER.

5.36.4 Error Conditions. Any error messages will be printed in the TREQ.COMO file.

5.36.5 Listings. The program listing comments assist in making program changes. The program listing TREQ.LIS is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.36	3. PROGRAM NAME TREQ.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.37 Program. Army Acquisition Report (RPT1.FOR)

5.37.1 Program Description. Program PRT1.FOR report is written in structured FORTRAN 77. The source code is in a file named RPT1.FOR. This file must be precompiled using the ORACLE FORTRAN host language embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT1.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program RPT1.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces the Army Acquisition Report. This report displays quantities and dollar values by family and item for the 5-year POM cycle plus two RAMP years. Subtotals are printed for each family, non-AAO hardware, non-AAO production base projects, total non-AAO, total hardware, and grand total.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot resourves (DEPOTR).

(2) From FAMILY table: Family names and family.

(3) From ITEM table: FSC; DODIC; SSN; HQ, AMC SEQ; family; sub-family; and item nomenclature.

(4) From RESULT1 table: DODIC, FY, Army production applied, and dollar value of Army production.

(5) From RAMP_ITEM table: DODIC, FY, Army buy quantity, and dollar value.

(6) From ITEM_DATA table: DODIC, FY, and POM cost.

c. Processing -

(1) Family names are retrieved from the FAMILY table and stored in the FAMILY_NAMES array.

(2) Selected columns are extracted from the ITEM table and stored ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(3) Data are retrieved from the RESULT1 table and stored in a dynamic data structure consisting of two data arrays (PROD_ARRAY and COST_ARRAY) and an index array. Each index entry consists of the family, sub-family, DODIC, and a pointer to the appropriate columns in the data arrays. Each data array has seven rows to provide space for five POM years plus two RAMP years.

(4) Data for two RAMP FYs are extracted from the RAMP_ITEM table and added to the data structure described in 5.37.1c(3).

(5) Quantities and costs accumulated in 5.37.1c(3) and 5.37.1c(4) are converted to thousands and millions, respectively.

(6) Items that have a POM cost are obtained from the ITEM_DATA table and added to the data structure described in 5.37.1c(3).

(7) Each column of accumulated data obtained in 5.37.1c(3), 5.37.1c(4), and 5.37.1c(6) is printed out along with related information from the ITEM_ARRAY. Form feeds and headers are printed so each family starts on a new page and to prevent printing over page breaks.

(8) In each index entry, the HQ, AMC SEQ is inserted into the first two bytes replacing family/sub-family. The index is resorted and 5.37.1c(7) is repeated except the items are now printed in HQ, AMC Sequence Number sequence.

5.37.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '$' character added. The name of the output file is RPT1.RCNx where 'x' is a particular revision control number (one or two digits).

5.37.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.37.4 Error Conditions. Error messages will be printed in the output file.

5.37.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT1.LIS) is located in <SYSSA>SAXPGM>$RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.37	RPT1.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.38 Program. Workload Summary Report by Item (by Plant and Line) (RPT2.FOR)

5.38.1 Program Description. The RPT2.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT2.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT2.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program RPT.RUN.

a. Functions - The program extracts information from the PJSM ORACLE data base and produces a detailed workload report for each plant/line combination. The report lists information for each item produced on each line. This information includes the items' SSN, DODIC, nomenclature, production rate, direct labor staffing, alternate producers of the item (if any). Army and other customer production, work-years, and percent line utilization are printed out for each FY.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR).

(2) From PLANT table: Plant code, plant name.

(3) From ITEM table: FSC, DODIC, SSN, and item nomenclature.

(4) From PRODT table: Plant code, line number, DODIC, 1-8-5 production rate, 1-8-5 direct labor staffing, and line availability. Only rows where avail >0 are retrieved.

(5) From RESULT3 table: DODIC, plant code, line number, FY, other customer production, total Army production, work-years for Army production, work-years for other customer production, total percent line utilization, and amount produced to support Army days of supply.

c. Processing -

(1) The data are extracted from the PLANT table and stored in arrays. A cross reference array is built which is subsequently used to convert plant numbers. Plant numbers are assigned in a manner that assures that the plant names will be in alphabetical order in the output report.

(2) Selected columns are obtained from the ITEM table and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(3) Data are retrieved from the PRODT table and stored in arrays. At the same time, a cross reference array is built which is subsequently used to check for alternative plants that produce an item.

(4) Data are obtained from the RESULT3 table and stored in a dynamic data structure consisting of five data arrays (ARMY_PROD, OTHR_PROD, ARMY_WRKY, OTHR_WRKY, and LINE_UTIL) and an index array. Each index entry contains plant number, line number, DODIC, and a pointer to the appropriate columns in the data arrays. Each data array has five rows -- one for each of the five FYs in the POM cycle.

(5) Print out each column of data in the data arrays along with related information obtained in steps 5.38.1c(1) to 5.38.1c(3).

5.38.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '$' character added. The name of the output file is RPT2.RCNx where 'x' is a particular RCN (one or two digits).

5.38.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.38.4 Error Conditions. Error messages will be printed in the output file.

5.38.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT2.LIS) is located in <SYSSA>SAXPGM>$RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TS 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.38	3. PROGRAM NAME RPT2.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.39 Program.. Plant Workload Utilization Detailed Report (RPT3.FOR)

5.39.1 Program Description. The RPT3.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT3.FOR. This file must be precompiled with the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT3.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program RPT3.RUN.

a. Functions - The program extracts information from the PJSM ORACLE data base and produces a detailed plant workload report for commercial and active Army Ammunition Plants. The report displays the quantity produced and work-years by plant, item, and FY.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base.

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR).

(2) From PLANT table: Plant code, plant name, production overhead factor and non-production overhead factor.

(3) From STAFF table: Plant, FY, and OMA overhead.

(4) From ITEM table: FSC, DODIC, SSN, and nomenclature.

(5) From RESULT3 table: DODIC, plant code, FY, other customer production + Army test and training production + Army additional days of supply production, and work-years Army + work-years other.

c. Processing -

(1) The data are extracted from the PLANT table and stored in arrays. A cross reference array is built which is subsequently used to convert plant codes into plant numbers. Plant numbers are assigned in a manner that assures that the plant names will be in alphabetical order in the output report.

(2) The plant code, FY and OMA overhead values are obtained from the STAFF table, and the OMA overhead data are stored in the array OMA_OH.

(3) Selected columns are retrieved from the ITEM table and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(4) The total production and total work-years data are extracted from the RESULT3 table and stored in a dynamic data structure consisting of two data arrays (TOTL_PROD and TOTL_WKYR) and an index array. Each index entry consists of the plant number, DODIC, and a pointer to the appropriate columns in the data arrays. There are five rows in the data arrays to provide space for five FYs.

(5) Each column of accumulated data in the production and work-years arrays is printed out in plant/DODIC sequence along with related information from the ITEM_ARRAY. Headers are added so each plant starts on a new page and to prevent printing over page breaks. Footer information for each plant computed using the overhead factors obtained in step 5.39.1c(2) and the OMA overhead obtained in step 5.39.1c(3).

5.39.2 Conventions. The program is written in structured FORTRAN 77. Program variables that directly receive values from the data base usually consist of the corresponding table column name with a '#' character added. The name of the output file is RPT3.RCNx where 'x' is a particular revision control number (one or two digits).

5.39.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.39.4 Error Conditions. Error message will be printed in the output file.

5.39.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT3.LIST) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.39	RPT3.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.40 Program. Plant Workload Summary Report (RPT4.FOR)

5.40.1 Program Desription. The RPT4.FOR program is written in structured FORTRAN 77 with embedded SQL statements. The source code is in a file named RPT4.FOR. This file is an input to the ORACLE FORTRAN host language precompiler that generates an intermediate file -- RPT4.F77 -- which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable program -- RPT4.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces a one page summary of plant workload in work-years for commercial and active Army Ammunition Plants for the 5-year POM cycle.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY.

(2) From PLANT table: Plant code, plant name, production overhead factor, and non-production overhead factor.

(3) From STAFF table: Plant code, FY, and OMA overhead.

(4) From RESULT3 table: Plant code, FY, and total work-years.

c. Processing -

(1) The plant code, plant name, the production overhead factor, and the non-production overhead factor are extracted from the PLANT table and stored in arrays. A cross reference array is built which is subsequently used to convert plant codes to plant numbers. Plant numbers are assigned to the plants in a manner that assures that the plants names will be in alphabetical order on the output report.

(2) The plant code, FY, and OMA overhead are obtained from the STAFF table, and the OMA overhead data are stored in an array -- OMA_OH.

(3) Plant code, FY and total work-years are retrieved from the RESULT3 table. The work-years data are accumulated in a dynamic data structure consisting of a data array and an index array. Each index entry consists of a plant number and a pointer to the appropriate column in the data array.

(4) Each column of accumulated data in the data array is printed out along with the total labor which is computed using the production overhead, non-production overhead, and OMA overhead data which were retrieved in steps 5.40.1c(1) and 5.40.1c(2). The following formulas are used:

(a) Production overhead = Production overhead factor * Total work-years.

(b) Non-production overhead = non-production overhead factor * (Total work-years + Production overhead + OMA overhead).

(c) Total labor = Total work-years + Production overhead + non-production overhead + OMA overhead.

5.40.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values from the data base usually consist of the table column name with a '$' character added. The name of the output file is RPT4.RCNx where 'x' is a particular RCN (one or two digits).

5.40.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.40.4 Error Conditions. Error messages will be printed in the output file.

5.40.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT4.LIS) is in <SYSSA>SAXPGM>$RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.		1. DATE 30 Nov 87
2. PROGRAM ID 5.40	3. PROGRAM NAME RPT4.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.41 Program. Analysis Worksheet Report (RPT5.FOR)

5.41.1 Program Description. The RPT5.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT5.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT5.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable program RPT5.RUN.

a. Function - The RPT5.FOR program extracts information from the PJSM ORACLE data base and produces a detailed ITEM ANALYSIS report. This is a long report (about 575 pages) because information is displayed for each plant/line/item combination. The report is ordered by family, sub-family, item (DODIC), plant, and line.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR).

(2) From PACKAGE table: Package numbers and names.

(3) From SERVICE table: Other customer codes and names.

(4) From PLANT table: Plant codes and names.

(5) From ITEM table: FSC, DODIC, SSN, family, sub-family, new materiel fielding code, item nomenclature, and unit of measure.

(6) From ITEM_DATA table: DODIC, FY, and unit price for Army.

(7) From RAMP_ITEM table: DODIC, FY, Army losses, unit price, and ending assets for the year prior to the 5-year POM cycle.

(8) From RAMP_PROD table: DODIC, FY, Army production and dollar value, other customer production (PROD_TOT - PROD_ARMY) and work-years (WRKYRS_TOT - WRKYRS_AR), and for year prior to the 5-year POM cycle.

(9) From PRODT table: DODIC, plant, line, 1-8-5 and 2-8-5 production rates and directed labor staffing, line availability, and minimum procurement quantity.

(10) From PRODT_FY table: Project code, DODIC, line, FY, month, 1-8-5 and 2-8-5 production rates and direct labor staffing. Only rows where RCN = specified RCN are retrieved.

(11) From RESULT1 table: DODIC, FY, package level, inventory applied, production applied, production dollar value, work-years, and shortfall quantity. Only rows where RCN = specified RCN, FY is between BFY - 3 and BFY + 4, and TYPE = 'E' are retrieved.

(12) From RESULT2 table: DODIC, FY, customer code, inventory applied, production applied, production dollar value, work-years, and shortfall quantity. Only rows where RCN = specified RCN, and FY is between BFY - 3 and BFY + 4 are retrieved.

(13) From RESULT3 table: DODIC, line, FY, and other customer production quantity. Only rows where RCN = specified RCN and FY is between BFY and BFY + 4 are retrieved.

(14) From RESULT4 table: DODIC, FY, and ending assets. Only rows where RCN = specified RCN and FY is between BFY and BFY + 3 are retrieved.

c. Processing -

(1) Package names are retrieved from the PACKAGE table and stored in the PKG array.

(2) Other customer codes and names are extracted from the SERVICE table and stored in arrays.

(3) Plant codes and names are obtained from the PLANT table and stored in arrays. A cross reference array is built which is subsequently used to convert plant codes into plant numbers. Plant numbers are assigned in a manner that assures that plant names will be in alphabetical order on the output report.

(4) Selected columns are extracted from the ITEM table and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(5) The ITEM_DATA table is retrieved to get unit price data by DODIC and FY. This information is stored in a dynamic data structure consisting of a data array and an index array.

(6) Projected beginning assets for the beginning FY are obtained from the RAMP_ITEM table. These data are stored in a dynamic data structure consisting of a data array and an index array. Army losses and unit prices for RAMP years are stored in other data structures.

(7) Army Production quantity, dollar value, other customer production quantity, and work-years data by DODIC and FY is retrieved from the RAMP_PROD table and stored in data structures.

(8) Selected columns are extracted from the PRODT table and stored in an array along with related data obtained in steps 5.41.1c(2) and 5.41.1c(3). At the same time two other arrays are built which subsequently are used to look up alternate plants for a given DODIC and alternate DODICs for a given line.

(9) Data are retrieved from the PRODT_FY table and stored in an array. These data are sorted by DODIC, line, FY, and month.

(10) Selected data are obtained from the RESULT1 table and stored in a dynamic in-memory data base system. This system consists of five data arrays (R1INV_APP, R1PROD_APP, R1PROD_DV, R1WRKYRS, and R1SFALL_QTR) and an index. The index uses DODIC and package level to form a unique concatenated key.

(11) Data are extracted from the RESULT2 table and stored in data arrays (R1INV_APP, R2PROD_APP, R2PROD_DV, R2WRKYRS, and R2SFALL_QTY). These arrays are indexed by a DODIC/service code concatenated index structure.

(12) Other customer production data are retrieved from the RESULT3 table and stored in a in-memory data base system. This consists of a data array (PROD_OTHER) and a DODIC/LINE composite index structure.

(13) Ending assets for each DODIC and for beginning FY through beginning FY + 3 are obtained from the RESULT4 table and added to the data obtained in step 5.41.1c(6).

(14) For each DODIC/plant/line combination obtained from the PRODT table in step 5.41.1c(8), do the following:

(a) Get the plant code and name from the array built in step 5.41.1c(3).

(b) Look up item information in the ITEM_ARRAY built in step 5.41.1c(4).

(c) Look up beginning assets in the data structure built in step 5.41.1c(6).

(d) Look up alternate plants and items in arrays built in step 5.41.1c(8).

(e) Look up any production base projects in the array built in step 5.41.1c(9). Write out data if any projects are found.

(f) Look up other customer requirements in the data structure built in step 5.41.1c(11) and print out.

(g) Get Army test and training losses from the in-memory data base system built in step 5.41.1c(6) and 5.41.1c(10).

(h) Compute and print out total quantities and dollar values for all package levels using the data from the RESULT1 and RAMP_PROD tables.

(i) Look up and print out unit prices for each FY using the data structure built in step 5.41.1c(5).

(j) Look up beginning assets for each FY in the PROJ_ASSETS array.

(k) Get other customer production from the RESULT3 data obtained in step 5.41.1c(12).

(l) Look up and print out Army production data by package level using the RESULT1 data obtained in step 5.41.1c(10).

(m) Look up and print out other customer production quantities and dollar values using the RESULT2 data obtained in step 5.41.1c(11).

5.41.2 Conventions. The program is written in highly structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '#' character added. The name of the output file is RPT5.RCNx where 'x' is a particular RCN (one or two digits).

5.41.3 Verification Procedures. The program can be verified by spot checking some of the output against other reports or data manually retrieved from the data base using the ORACLE UFI.

5.41.4 Error Conditions. Error messages will be printed in the output file.

5.41.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT4.LIS) is located in <SYSSA>SAXPGM>$RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.41	3. PROGRAM NAME RPT5.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.42 Program. Program Development Incremental Package (PDIP) Summary Report (RPT6.FOR).

5.42.1 Program Description. The RPT6.FOR program is written in FORTRAN 77. The source code is identified by a file labeled RPT6.FOR. After being precompiled with the ORACLE precompiler, compiled with the FORTRAN 77 compiler and loaded with BIND, the run file is identified as RPT6.RUN.

a. Function - The RPT6.FOR program extracts cost information by PDIP from the PJSM data base, summarizes the data by year, NMF and PDIP, and writes a one page summary report. The user, by means of the JSM Master Menu, can run the program and print the report. Users have access to this program only through the JSM Master Menu.

b. Input -

(1) Interactive: ORACLE identification and password along with the RCN.

(2) From data base and accessed by main program:

(a) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR).

(b) From PACKAGE table: Name of each of the package levels (PKGL).

(c) From ITEM table: List of items (DODICs), their family, subfamily, and New Materiel Fielding (NMF) code identifiers.

(d) From RESULT1 table: Dollar value of scheduled production by item, by package level, and by FY for each record for the specified RCN where the FYs are equal to the BFY through the BFY + 4 and the PKGL is equal to package 2 through 17.

(e) From ITEM_DATA table: Dollar value for classified items or for items which have no Authorized Acquisition Objective (non-AAO) for each record for RCN = 0 (Baseline) where the FYs are equal to the BFY through the BFY + 4 and POM_COST > 0.

c. Processing -

(1) The user's ORACLE identification and password, along with the specified RCN, are passed to the program on the command line.

(2) Input obtained from the PJSM ORACLE data base:

(a) From the REVTAB table, the BFY is obtained along with several other variables which are not used in this program.

(b) From the PACKAGE table, the package name for each package level (PKGL) is obtained.

(c) From the ITEM table, the list of items (DODICs), their NMF, and family and sub-family codes are obtained and loaded into ITEM_ARRAY.

(3) Subroutine HSORT is called where the items in the ITEM_ARRAY are sorted alphabetically by DODIC to facilitate the subsequent use of a fast binary search routine.

(4) Build output array.

(a) Cost Array - Initialize the COST_ARRAY (I,J) to zeros for each element, where I = 19 (# of rows) and J = 12 (# of columns). The odd numbered columns contain data for NMF items for each FY, with a summary in column 11. The even numbered columns contain data for all other items for each FY with a summary given in column 12. Rows 2 thru 17 of the COST_ARRAY contain data for packages 2 thru 17, respectively. Row 1 contains a total of packages 2 thru 17. Rows 18 and 19 contain the total costs of non-AAO hardware and projects, respectively. The program reads and accumulates all cost data, storing the results and relevant data in the COST_ARRAY. At the end of the program, the data will be written out in FY and package order to generate the report.

(b) From PJSM data base, RESULT1 table, read the item (DODIC), FY, package level (PKGL) and dollar value of scheduled production. With these data perform the following steps and repeat for each record retrieved from the RESULT1 table:

(1) Match the item (DODIC) with the list of items obtained from step 4.41.1c(2)(c), in order to obtain the item's NMF code.

(2) Add the dollar value of the scheduled production to the appropriate elements of the COST_ARRAY. If the item was a NMF item and the package level was 3, 4, 5, 6, or 7, then the dollar value would be added to the NMF columns; if not, then add the dollar value to the columns for other items. Also add the dollar value to the yearly totals and the package totals.

(c) Convert all values in the COST_ARRAY to millions of dollars (divide each element by 1,000,000).

(d) From PJSM data base, ITEM_DATA table, read the DODIC, FY, and the POM_COST where POM_COST>0. Perform the following steps for each record retrived from the ITEM_DATA table:

(1) Look up the item (DODIC) in the item array obtained in step 5.42.1c(2)(c) to determine the item's family and sub-family codes.

(2) Check the family code. All items that have a POM_COST should be in family 7. Write an error message if the family code is not 7 and skip 5.42.1c(4)(d)(3) and 5.42.1c(4)(d)(4) below.

(3) If the family code equals seven, then check the subfamily code to determine if the POM_COST is for hardware procurement (subfamily codes 1 or 2) or for production base improvement projects (sub-family 3).

(4) Add the POM_COST to the appropriate row and column (determined by FY) in the COST_ARRAY.

(5) Write output report.

(a) Write general information and column headings to the output file.

(b) Write the accumulated data in the COST_ARRAY to the output file.

(c) After all of the dollar values in the COST_ARRAY are accumulated, write the grand totals to the output file.

5.42.2 Conventions. Program is written in structured FORTRAN 77.

5.42.3 Verification Procedures. Both the developer and the proponent will test and sign off on any changes.

5.42.4 Error Conditions. Error conditions are documented on the hard copy report produced during each execution.

5.42.5 Listings. Program listings contain detailed comments to assist in making program changes and are located in the <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.		1. DATE 30 Nov 87
2. PROGRAM ID 5.42	3. PROGRAM NAME RPT6.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.43 Program. Compare Program (RPT7.FOR)

5.43.1 Program Description. The Compare Program is written in structured FORTRAN 77. Source code is in the COMP.FOR file. This file is input into the ORACLE precompiler that builds the COMP.F77 file which, in turn, is compiled with the F77 compiler and loaded with the BIND utility producing the executable version COMP.RUN. The following paragraphs describe the RPT7.FOR program:

a. Identification - Source code for this program is located in the COMP.FOR file. Its executable equivalent is COMP.RUN. Users will have access to this program only through the JSM Master Menu.

b. Functions - The RPT7.FOR program produces a report which details by year the differences in the Army production and the dollar value of the Army production for two RCNs. (All numbers appear in thousands.)

c. Input -

(1) Interactive: The user will enter the ORACLE identification and password, baseline RCN, and desired processing option as part of the command line. No prompts will be initiated by the program.

(2) By program:

(a) From REVTAB: Read beginning FY and number of FYs for the two user-input RCNs.

(b) From ITEM: Read the SSN, FSC, DODIC, new materiel fielding code, and the nomenclature for each item.

(c) From RESULT4: Read the Army production and dollar value of Army production for each item by FY for both RCNs.

(d) From RESULT1: Read the amount of undelivered production used; i.e., package level = 0, for each RCN and item by FY.

d. Processing -

(1) User enters ORACLE identification and password.

(2) User enters desired RCN for baseline.

(3) User enters desired RCN for comparison.

(4) Program reads beginning FY and number of FYs for both RCNs entered. The program then determines which years, if any, are common to both RCNs and can be used for comparison.

(5) Program reads ITEM table for list of DODICs and their associated SSNs, FSCs, NMF codes, and nomenclature.

(6) Declare and open a cursor that will return Army production and dollar value of Army production for all study years.

(7) After each successful return, select the production applied for the current RCN, DODIC, and FY for package level = 0 from RESULT1. This value must be subtracted from the Army production return from RESULT4 to account for the undelivered amounts embedded in the RESULT4 number.

(8) Complete steps 5.43.1d(6) and 5.43.1d(7) for both the baseline RCN and the alternative RCN.

(9) For each item do the following:

(a) For each year studied, determine if there is any difference between the Army production for each RCN.

(b) If there is no difference in any year and only the items with differences are being printed, then continue to next item.

(c) Otherwise, write out the item information as well as the baseline production and the difference between the two RCNs' production. Keep running tally of dollar values and differences by FY.

(d) Complete steps 5.43.1d(9)(a) to 5.43.1d(9)(c) for the dollar value of the Army production. Keep running tally of dollar values and differences by FY.

(10) Write out the totals from 5.43.1d(9)(d).

(11) End program.

e. Output - COMP.RCN**.VS.RCN**

f. Interfaces - This program only requires that the two RCNs to be compared both be resident in the RESULT4 table. It does not require interfaces with other programs.

g. Tables - See Annex B.

5.43.2 Conventions. This program was written with the convention that non-array local variables end with a '#' character. This will help distinguish them from similarly named table columns.

5.43.3 Verification Procedures. Verification must be done using the ORACLE UFI utility to access the data base.

5.43.4 Error Conditions. Errors encountered will be documented in the program output.

5.43.5 Listings. Program listing is located in <SYSSA>SAXPGM>#PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.		1. DATE 30 Nov 87
2. PROGRAM ID 5.43	3. PROGRAM NAME RPT7.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.44 Program. Summary of Training and Test Items not Achieving 100 Percent (RPT8.FOR)

5.44.1 Program Description. The RPT8.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT8.FOR. This file is an input to the ORACLE FORTRAN host language precompiler which generates an intermediate file RPT8.F77. This file is compiled with the PRIME F77 compiler, and the PRIME BIND utility is used to produce an executable program RPT8.RUN.

a. Function - The PJSM program extracts information from the ORACLE data base, and produces a report showing items which have a test and/or training shortfall in at least one of the five POM FYs.

b. Input - The program requires a valid ORACLE user name and password and a RCN.

All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY, training level (LVTRNG), and depot reserves (DEPOTR).

(2) From ITEM table: FSC, DODIC, SSN, and item nomenclature.

(3) From RESULT1 table: DODIC, FY, PKGL, INV_APP, PROD_APP, SFALL_QTY. Only rows having the specified RCN, FYs between BFY and BFY + 4, PKGL of 2, 5, or 10 and SFALL_QTY > 0 are retrieved.

c. Processing -

(1) Selected columns from the ITEM table are retrieved and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(2) The data from RESULT1 are retrieved and stored in a dynamic data structure consisting of a data array (PCT_FILL) and an index array. Each index entry consists of a unique package level/ DODIC combination followed by a pointer to the column in the PCT_FILL array which contains the data for that combination. There are five rows in the PCT_FILL array to provide space for five FYs. Percent filled quantities are computed as 100 * (INV_APP + PROD_APP)/ (INV_APP + PROD_APP + SFALL_QTY).

(3) The accumulated data in the PCT_FILL array are printed out in package level/DODIC sequence along with related information from the ITEM_ARRAY.

5.44.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values from the data base usually consist of the table column name with a '$' character added. The name of the output file is RPT8.RCNx. Where 'x' is a particular RCN (one or two digits).

5.44.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.44.4 Error Conditions. Error messages will be printed in the output file.

5.44.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT8.LIS) is located in <SYSSA>SAXPGM>$RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.44	3. PROGRAM NAME RPT8.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FORM 5752, NOV 73, WHICH IS OBSOLETE.

5.45 Program. Readiness Report (RPT9.FOR)

5.45.1 Program Description. The RPT9.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT9.FOR. This file is an input to the ORACLE FORTRAN host language precompiler which generates an intermediate file RPT9.F77 that is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program RPT9.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces a detailed report that displays for each item the SSN, DODAC, the use code, nomenclature, and the last package level and percent filled for a 5-year POM cycle.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY where RCN = specified RCN.

(2) From ITEM table: FSC, DODIC, SSN, use code, and item nomenclature.

(3) From PACKAGE: Package level, package name.

(4) From RESULT1: DODIC, FY, PKGL, INV_APP, PROD_APP, and SFALL_QTY. Only rows having the specified RCN, FY between BFY and BFY + 4, PKGL > 1, type of 'E' or 'U' and INV_APP > 0 are retrieved.

c. Processing -

(1) Package levels and corresponding package names are extracted from the PACKAGE table and displayed on the first page of the report.

(2) Selected columns from the ITEM table are retrieved and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(3) The data from the RESULT1 table are retrieved and stored in a dynamic data structure consisting of two data arrays and an index array. The first data array (LPF) contains the last package level for which inventory and/or production was applied for each item and FY. The second data array (PCF) contains the percent filled for the last package level for each item and FY. Each entry in the index array contains a sequence number, DODIC, and a pointer to the appropriate columns in the data arrays.

(4) Each column of accumulated data in the data arrays is printed out in DODIC sequence along with related information from the ITEM_ARRAY.

(5) The HQ, AMC Sequence Number is inserted into the first two bytes of each index key. The index array is then resorted, and 5.45.1(3)(d) is repeated except now the items are listed in HQ, AMC Sequence Number sequence.

5.45.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values from the data base usually consist of the table column name with a '$' character added. The name of the output file is RPT11.RCNx where 'x' is a particular RCN (one or two digits).

5.45.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.45.4 Error Conditions. Error messages will be printed in the output file.

5.45.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT9.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.45	3. PROGRAM NAME RPT9.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.46 Program. Other Service Shortfall Report (RPT10.FOR)

5.46.1 Program Description. The RPT10.FOR program is written in structured FORTRAN 77 with embedded SQL statements. The source code is in a file named RPT10.FOR. This file is an input to the ORACLE FORTRAN host language precompiler which generates an intermediate file RPT10.F77. This file is compiled with the PRIME F77 compiler and the PRIME BIND linker utility is used to produce an executable program RPT10.RUN.

a. Functions - The program extracts information from the PJSM ORACLE data base and produces a report showing items which have an Other Service shortfall in at least one of the five POM FYs.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR) for the specified RCN.

(2) From ITEM table: FSC, DODIC, SSN, and item nomenclature.

(3) From RESULT2: DODIC, FY, INV_APP, PROD_APP, and SFALL_QTY. Only rows where RCN = the specified RCN and SFALL-QTY > 0 are retrieved.

c. Processing -

(1) Selected columns from the ITEM tables are retrieved and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(2) The data from RESULT2 are extracted and stored in a dynamic data structure consisting of three data arrays (INVT_APP, PROD_APP, and SFALL_QTY) and an index array. Each index entry contains a DODIC and a pointer to the columns in the data arrays. Each data array has five rows -- one for each of five POM FYs.

(3) Each column of accumulated data in the data arrays built in step 5.46.1c(2) is printed out in DODIC sequence along with related information from the ITEM_ARRAY. The Total Other Service requirement is the sum of Production applied, Inventory applied, and shortfall quantity. All quantities are then converted to thousands by dividing by 1,000.

5.46.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values from the data base usually consist of the table column name with a '#' character added. The name of the output file is RPT10.RCNx where 'x' is a particular RCN (one or two digits).

5.46.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.46.4 Error Conditions. Error messages will be printed in the output file.

5.46.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT10.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.	1. DATE 30 Nov 87
2. PROGRAM ID 5.46	3. PROGRAM NAME RPT10.FOR
4. REV NO./DATE	5. DESCRIPTION OF REVISION

DA FORM 4792-R, APR 83 REPLACES DA FORM 5752, NOV 78, WHICH IS OBSOLETE.

5.47 Program. PDIP Package Structure by Item (RPT11.FOR)

5.47.1 Program Description. The RPT11.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT11.FOR. This file is an input to the ORACLE FORTRAN host language precompiler which generates an intermediate file RPT11.F77. This file is compiled with the PRIME F77 compiler and the PRIME BIND utility is used to produce an executable program RPT11.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces a detailed report that shows the quantities produced and the $ value by item, package level, and FY.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY (NFY), training level (LVTRNG), and depot reserves (DEPOTR) for specified RCN.

(2) From ITEM table: FSC, DODIC, SSN, SEQ, and item nomenclature.

(3) From RESULT1 table: DODIC, FY, PKGL, PROD_APP. and PROD_DV. Only rows with the speicifed RCN and DODICs that do not begin with 'x', FY between BFY and BFY + 4, type = 'E' and PROD_APP > 0 are returned.

c. Processing -

(1) Selected columns from the ITEM table are extracted and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(2) The data from RESULT1 are retrieved and stored in a dynamic data structure consisting of two data arrays (PROD_ARRAY and COST_ARRAY) and an index array. Each index entry contains a concatenated key consisting of a sequence number, DODIC, and package level followed by a pointer to the appropriate columns in the data arrays. Each data array has five rows -- one for each of five POM fiscal years.

(3) Quantities in the PROD_ARRAY are converted to thousands and each element in the COST_ARRAY is converted to millions. Each column of accumulated data in the data arrays is printed out in DODIC sequence along with related information from the ITEM_ARRAY.

(4) The HQ, AMC Sequence Number is inserted into the first two bytes of each index key. The index array is then resorted and 5.47.1(3)(c) is repeated except now the items are listed in HQ, AMC Sequence Number sequence.

5.47.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values from the data base usually consist of the table column name with a '#' character added. The name of the output files is RPT11.RCNx where 'x' is a particular RCN (one or two digits).

5.47.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.47.4 Error Conditions. Error messages will be printed in the output file.

5.47.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT11.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.47	3. PROGRAM NAME RPT11.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.48 Program. PDIP Cost Report (RPT12.FOR)

5.48.1 Program Description. The RPT12.FOR program is written in structured FORTRAN 77. The source code is in a file named RPT12.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT12.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND utility is used to produce an executable program RPT12.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces a detailed PDIP Cost Report. This report displays quantities produced and cost by package level, item, and FY. Subtotals are printed for each package level, non-AAO hardware, non-AAO production base projects, total non-AAO, total hardware, and grand totals.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY, training level (LVTRNG), and depot reserves (DEPOTR) for specified RCN.

(2) From PACKAGE table: Package level and package description. Only rows where PKGL > 0 are retrieved.

(3) From ITEM table: FSC, DODIC, SSN, family, subfamily, new materiel fielding code, and item nomenclature.

(4) From RESULT1 table: DODIC, FY, PKGL, PROD_APP, and PROD_DV. Only rows having the specified RCN, FY between BFY and BFY + 4, PKGL between 2 and 17, and PROD_DV > 0 are returned.

(5) From ITEM_DATA table: DODIC, FY, and POM cost. Only rows having FY between BFY and BFY + 4, and RCN = 0 or the specified RCN, and POM_COST > 0 are returned. Any rows returned where RCN = specified RCN will replace the data returned with RCN = 0.

c. Processing -

(1) Package names are extracted from the PACKAGE table and stored in an array.

(2) Selected columns are obtained from the ITEM table and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(3) Data are retrieved from the RESULT1 table and summed in a dynamic data structure consisting of two data arrays (COST_ARRAY and PROD_ARRAY) and an index array. Each index entry contains package level, DODIC, and a pointer to the appropriate columns in the data arrays. Each data array has five rows -- one for each of the five FYs in the POM cycle. For each row of data retrieved from the RESULT1 table the following steps are performed:

(a) If the package level is 3 through 7 the item is looked up in the ITEM_ARRAY built in step 5.48.1c(2) to determine if the item is a NMF item. If the item is a NMF item, package level is set to 1.

(b) The production quantities (PROD_APP) and dollar values (PROD_DV) are added to the appropriate elements in the PROD_ARRAY and the COST_ARRAY, respectively. The column is determined by the package level/DODIC combination and the row is determined by the FY.

(4) Production and cost data accumulated in 5.48.1c(3) are converted to thousands and millions, respectively.

(5) The POM cost is extracted from the ITEM_DATA table and added to in the COST_ARRAY. A pseudo package level of 18 is used if the sub-family is 1 or 2, and 19 is used for sub-family 3 (non-AAO production base projects).

(6) Each column of data accumulated in steps 5.48.1c(3) and 5.48.1c(5) is printed out along with related information obtained in steps 5.48.1c(1) and 5.48.1c(2). Form feeds and headers are printed so each package starts on a new page and to prevent printing over page breaks.

5.48.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '$' character added. The name of the output file is RPT12.RCNx where 'x' is a particular RCN (one or two digits).

5.48.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.48.4 Error Conditions. Error messages will be printed in the output file.

5.48.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT12.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.		1. DATE 30 Nov 87
2. PROGRAM ID 5.48	3. PROGRAM NAME RPT12.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4792-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.49 Program. Work-years Available to Increase Plant Utilization Report (RPT13.FOR)

5.49.1 Program Description. The RPT13.FOR program is written in structured FORTRAN 77. The source code contains about 1,100 lines including comments and is in a file named RPT13.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces the embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT13.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable program RPT13.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces a detailed plant utilization report for commercial and each active Army Ammunition Plant. This report displays a summary of work-years available, workload goal, actual work-years, and percent line utilization for each plant for the 5-year POM cycle. Additionally, detailed information is displayed for each item which has remaining requirements.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY, training level (LVTRNG), and depot reserves (DEPOTR) for specified RCN.

(2) From PLANT table: Plant codes and plant names.

(3) From STAFF table: Plant code, FY, workload goal, and direct labor. Only rows where FY is between BFY and BFY + 4, and RCN = 0 or the specified RCN are returned. The order by clause assures that the exceptional data (RCN = the specified data) is retrieved last so it will override the base data (RCN = 0).

(4) From ITEM table: FSC, DODIC, SSN, and item nomenclature.

(5) From ITEM_DATA table: DODIC, FY, and unit price. Only rows where FY is between BFY and BFY + 4, and RCN = 0 or the specified RCN are returned. The order by clause assures that the exceptional data (RCN = the specified data) is retrieved last so it will override the base data (RCN = 0).

(6) From PRODT table: Plant code, line number, DODIC, 1-8-5 production rate, 1-8-5 direct labor staffing, and line availability. Only rows where AVAIL > 0 are retrieved.

(7) From PRODT_FY table: Plant code, line number, DODIC, FY, 1-8-5 production rate, 1-8-5 direct labor staffing, and line availability. Only rows where FY is between BFY and BFY + 4, and RCN = 0 or the specified RCN are returned. The order by clause assures that the exceptional data (RCN = the specified data) is retrieved last so it will override the base data (RCN = 0).

(8) From RESULT1 table: DODIC, FY, and package level. Only rows where FY is between BFY and BFY + 4, TYPE = 'E', and SFALL_QTY = 0 are retrieved.

(9) From RESULT3 table: DODIC, FY, and other customer production. Only rows where RCN = the specified RCN, FY is between BFY and BFY + 4, and PROD_OTHER > 0 are retrieved.

(10) From RESULT4 table: DODIC, FY, Army production, other customer shortfall quantity, and Army shortfall quantity. Only rows where RCN = the specified RCN, FY is between BFY and BFY + 4, TYPE = 'E', are retrieved.

(11) From RESULT6 table: FY, plant code, line number, work-years, and percent line utilization. Only rows where RCN = the specified RCN, and FY is between BFY and BFY + 4 and total shortfall > 0 are retrieved.

c. Processing -

(1) Plant codes and names are extracted from the PLANT table and stored in arrays. A cross reference array is built which is subsequently used to convert plant codes into plant numbers. Plant numbers are assigned in a manner that assures that the plant names will be in alphabetical order on the output report.

(2) Data are obtained from the STAFF table and stored in two data arrays -- GOALS and DIRECT. GOALS is the percent of direct labor which should be achieved to balance workload. Array DIRECT contains the number of direct labor employees by plant and FY.

(3) Selected columns are extracted from the ITEM table and stored in an array -- ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(4) Unit price data are retrieved from the ITEM_DATA table and stored in an array. Data in this array are organized by DODIC and FY. Unit prices are used to compute the remaining requirement or capacity dollar value.

(5) Selected columns are obtained from the PRODT table and stored in an array -- PD_ARRAY.

(6) FY plus the same columns selected in 5.49.1b(5) are retrieved from the PRODT_FY table and stored in the PD_ARRAY.

(7) The PD_ARRAY is sorted so that the exceptional data from PRODT_FY immediately follows the base data from the PRODT table. This is accomplished by making FY the last part of the sort key and setting FY to zero for the data from the PRODT table.

(8) The RESULT1 table is extracted to get the last package filled for each DODIC and FY.

(9) Other customer production is retrieved from the RESULT3 table and stored in an array by DODIC and FY.

(10) The RESULT4 table is obtained to get Army production, other customer shortfall, and Army shortfall data for each DODIC and FY.

(11) Table RESULT6 is extracted to get total work-years for each plant and percent line utilization for each line.

(12) The information for each plant in the PD_ARRAY obtained in steps 5.49.1c(5) to 5.49.1c(7) is printed out along with related data obtained in steps 5.49.1c(1), 5.49.1c(2), and 5.49.1c(11). Percent plant utilization is computed as 100 * work-years/direct labor staffing at plant.

(13) Detailed information is printed for each line/DODIC combination in the PD_ARRAY where there is an unfilled requirement. Some of the quantities printed in this part of the report are computed in the program using the following formulas:

(a) Remaining Capacity = ((Line Availability * 100 - Percent Line Utilization) * 1-8-5 Production Rate * 12) / 100.

(b) Remaining Requirement or Capacity $ Value = Minimum (Army Shortfall + Other Service Shortfall, Remaining Capacity) * Unit Price / 1000000.

(c) Work-years Required to Produce = Minimum (Army Shortfall + Other Service Shortfall, Remaining Capacity) * 1-8-5 Staffing / (1-8-5 Production Rate * 12).

(14) Form feeds and headers are printed so each plant starts on a new page and to prevent printing over page breaks.

5.49.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '#' character added. The name of the output file is RPT13.RCNx where 'x' is a particular RCN (one or two digits).

5.49.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.49.4 Error Conditions. Error messages will be printed in the output file.

5.49.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT13.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.49	3. PROGRAM NAME RPT13.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-E, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.50 Program. Secondary Item Status Report (RPT14.FOR)

5.50.1 Program Description. The Secondary Item Status Report program is written in structured FORTRAN 77. Source code can be found in the RPT14.FOR file which is input to the ORACLE precompiler that creates the RPT14.F77 file. This file in turn is compiled with the F77 compiler and then loaded using the BIND utility. The executable file will be named RPT14.RUN. The following paragraphs describe the RPT14.FOR program:

a. Identification - Source code for this program is located in a file called RPT14.FOR. Its executable equivalent is RPT14.RUN. Users will have access to this program only through the JSM Master Menu.

b. Functions - The RPT14.FOR program produces the Secondary Item Status Report which is a detailed listing of requirements for secondary components by FY and by end item or primary component. Also included is beginning asset posture, production data, and shortfall information by year for the component.

c. Input -

(1) Interactive: The menu will pass the user ID, password, and RCN to the program in the command line.

(2) By program:

(a) From REVTAB table: Beginning FY and number of FYs.

(b) From REASON_CODE table: Read list of reason codes and description.

(c) From ICT table: Read list of distinct secondary components, DODIC, and procurement factors.

(d) From RESULT4 table: Read Army inventory applied to support test and training, and ending asset for the end item or primary component by FY. Also read inventory applied to test and training, ending assets, shortfall for test and training, shortfall for days of supply, and production amount for secondary item.

(e) From RESULT3 table: Read Army production to support test and training by FY for end items and primary components. Also read production for test and training for secondary components.

(f) From ITEM table: Read nomenclature.

(g) From RESULT1 table: Read reason code for shortfall associated with secondary component, by FY.

(h) From RAMP_ITEM: Read ending assets for secondary component for year = beginning FY minus 1.

d. Processing - The following paragraphs describe the processing sequence:

(1) User enters ORACLE information and password.

(2) User enters desired RCN.

(3) Read REVTAB for beginning FY and number of FYs and write to screen.

(4) Read REASON_CODE table for list of reason codes and descriptions.

(5) Read ICT table for list of distinct secondary components and their procurement factors.

(6) For each secondary component:

(a) Read the nomenclature from the ITEM table.

(b) Declare and open a cursor that will return from ICT table each end item and primary component that employs the current secondary component.

(c) For each DODIC selected with this cursor, do the following:

(1) From RESULT4: Read the Army inventory applied to test and training, and end assets, both by FY.

(2) From RESULT3: Read the Army production applied to test and training by FY.

(3) Write the DODIC, the sum of Army inventory and Army production applied to test and training, and ending assets, both by year, to the output report. In addition, apply the components procurement factor to the two previous values to compute the requirement of component to support test and training, and the ending asset posture of the component, both by year. Write these values to the output report. Also keep a cumulative total of test and training support and ending assets across end items and primary components.

(4) Loop to next end item or primary component.

(d) Write the totals for the accumulated test and training losses, and ending asset with the applied procurement factor, by year.

(e) Read RESULT4 for the Army inventory applied to test and training, ending assets, shortfall for test and training, shortfall for Army days of supply and Army production, by FY.

(f) Read RESULT3 for Army production to support test and training by FY.

(g) Read RESULT1 for first non-zero reason code.

(h) Read RAMP_ITEM table for ending assets for FY = beginning FY minus 1.

(i) Beginning asset posture for beginning FY plus 1 and beyond will be equal to the previous year's ending assets.

(j) Write the component data to the output by year.

(k) The first page will list valid reason codes and descriptions.

(l) Loop to next component.

(7) Close report, end program.

e. Output - Secondary Item Status Report.

f. Interfaces - This program requires no interfacing with any other program. Users can access this program at any time but only through the JSM Master Menu.

g. Tables - See Annex B.

5.50.2 Conventions. This program was written with the convention that all local variables will end with a '$' character. This will help to distinguish from similarly names table columns. In addition, the output needs no special format controls when spooled. The output will signal the printer so that format controls are recognized. This program also does not prompt for user ID, password, or RCN because they are entered as part of the command line to initiate the program.

5.50.3 Verification. Verification will be done using the ORACLE UFI utility to access the data base.

5.50.4 Error Conditions. Errors will be listed in the COMO file produced at each execution of the program.

5.50.5 Listings. Program listing is located in <SYSSA>SAXPGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.50	3. PROGRAM NAME RPT14.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.51 Program. Summary of Package Requirement/Filled (RPT15.FOR)

5.51.1 Program Description. The Summary of Package Requirement/ Filled Report is written in structured FORTRAN 77. Source code is found in the RPT15.FOR file which is input to the ORACLE precompiler that builds the RPT15.F77 file. This file is compiled with the F77 compiler and loaded using the BIND utility which builds the executable program RPT15.RUN. The following paragraphs describe the RPT15.FOR program:

a. Identification - Source code for the program is found in the RPT15.FOR file. Its executable equivalent is RPT15.RUN. Users will have access to this program only through the JSM Master Menu.

b. Functions - RPT15.FOR is a summary report of number of items with requirements versus those not filled, by package, for all items and also a separate summary for New Materiel Items only. Included in the latter is a list of items not filled, broken out by year.

c. Input -

(1) Interactive: The user enters his ORACLE identification, password, and RCN as part of the command line - no prompts are used.

(2) By Program:

(a) From REVTAB: Read beginning FY and number of FYs.

(b) From PACKAGE table: Read list of package levels and names.

(c) Form ITEM table: Read list of DODICs and their new materiel fielding code.

(d) From RESULT1 table: Read the shortfall by DODIC, FY, and package level.

d. Processing -

(1) Read beginning FY and number of FYs from REVTAB.

(2) Read list of packages and package descriptions from PACKAGE table.

(3) Read list of DODICs and their new materiel fielding code from ITEM table.

(4) Read the RESULT1 table for shortfalls by DODIC, FY, and package level.

(a) Keep track of the number of items read by package and year.

(b) If there is a shortfall, keep track of this number also by package and FY.

(c) If the item is a new materiel item, keep track of this number separately from the others.

(d) Also, if the new materiel item has a shortfall, keep a running total of this number by package level and FY.

(e) If the item is a new materiel item and it has a shortfall, add this item to a list of new materiel fielding items that will be output last. This list is by year.

(f) Read through the RESULT1 table.

(5) Write the summary for all items, the number of items with a requirement and the number of items with a shortfall by package level and FY.

(6) Give column totals in each case.

(7) Produce a similar summary for new materiel fielding items, but only for package levels 3 through 7. Also, provide column totals.

(8) For new materiel fielding items, write out the list of items with shortfalls in package levels 3 through 17, by FY.

(9) Close report, end program.

e. Output - RPT15.RCN**

f. Interfaces - This program can be run any time after the RESULT1 table has been loaded. No interface with any other program is required.

g. Tables - See Annex B.

5.51.2 Conventions. This program was written with the convention that all local variables end with a '$' character to distinguish them from similarly named table columns. Also, no format control is required when output is spooled. The report will signal the printer to recognize the format controls.

This program does not prompt for user identification, password, or RCN. This information is entered as part of the command line to initiate the program.

5.51.3 Verification Procedures. Verification must be done using the ORACLE UFI utility to access the data base.

5.51.4 Error Conditions. All errors will be documented in the COMO file produced at each execution of the program.

5.51.5 Listings. Program listing is located in <SYSSA>PGM>$PGM. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.51	3. PROGRAM NAME RPT15.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.52 Program. Conventional Ammunition Acquisition Plan (RPT16.FOR)

5.52.1 Program Description. The RPT16.FOR program is written in structured FORTRAN 77. The source code contains about 710 lines including comments and is in a file named RPT16.FOR. This file must be precompiled using the ORACLE FORTRAN host language precompiler which replaces embedded SQL statements with calls to library routines. The precompiler generates an intermediate file RPT16.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce an executable program RPT16.RUN.

a. Function - The program extracts information from the PJSM ORACLE data base and produces the Conventional Ammunition Acquisition Report for both Army and other customers. This report displays quantities and dollar values by family, item, and customer for the 5-year POM cycle plus two RAMP years. Subtotals are printed for each family, non-AAO hardware, non-AAO production base projects, total non-AAO, total hardware, and grand totals.

b. Input - The program requires a valid ORACLE user name, a password, and a RCN. All other data are obtained from the ORACLE data base:

(1) From REVTAB table: Beginning FY, next FY, training level (LVTRNG), and depot reserves (DEPOTR) where RCN = specified RCN.

(2) From FAMILY table: Family names and family where sub-family > 0.

(3) From ITEM table: FSC; DODIC; SSN; HQ, AMC SEQ; family; sub-family; and item nomenclature.

(4) From RESULT1 table: DODIC, FY, Army production, and dollar value. Only rows where RCN = specified RCN, FY is between BFY and BFY + 4, TYPE = 'E', and PROD_DV > 0 are retrieved.

(5) From RESULT2 table: DODIC, FY, service code, production quantity, and dollar value. Only rows where RCN = specified RCN, FY is between BFY and BFY + 4, and PROD_DV > 0 are retrieved.

(6) From RAMP_ITEM table: DODIC, FY, Army buy quantity, and dollar value. Only rows where FY is between BFY - 2 and BFY - 1, and DV_ARMY > 0 are retrieved.

(7) From ITEM_DATA table: DODIC, FY, and POM cost. Only rows where RCN = 0, FY is between BFY - 2 and BFY + 4, and POM_COST > 0 are retrieved.

c. Processing -

(1) Family names are obtained from the FAMILY table and stored in an array.

(2) Selected columns are retrieved from the ITEM table and stored in ITEM_ARRAY. This array is sorted in ascending DODIC sequence to facilitate the subsequent use of a fast binary search routine.

(3) Data are extracted from the RESULT1 table and stored in a dynamic data structure consisting of two data arrays (PROD_ARRAY and COST_ARRAY) and an index array. Each index entry consists of the family, sub-family, DODIC, customer code, and a pointer to the appropriate columns in the data array. Each data array has seven rows to provide space for five POM years plus two RAMP years.

(4) Other customer production and dollar value data are obtained from the RESULT2 table and stored in the same data structure described in 5.52.1c(3).

(5) RAMP data for Army only are extracted from the RAMP_ITEM table and added to the same data structure described in 5.52.1c(3).

(6) Quantities and cost data are converted to thousands and millions, respectively.

(7) POM cost data for Army only are retrieved from the ITEM_DATA table and added to the same data structure described in 5.52.1c(3).

(8) Each column of accumulated data obtained in steps 5.52.1c(3), 5.52.1c(4), 5.52.1c(5), and 5.52.1c(7) is printed out along with related data from the ITEM_ARRAY. Form feeds and headers are printed so each family starts on a new page and to prevent printing over page breaks.

(9) In each index entry, the HQ, AMC SEQ is inserted into the first two bytes replacing the family/sub-family numbers. The index is resorted and 5.52.1c(8) is repeated except the items are now printed in HQ, AMC SEQ sequence.

5.52.2 Conventions. The program is written in structured FORTRAN 77. Program variables that receive values directly from the data base usually consist of the corresponding table column name with a '#' character added. The name of the output file is RPT16.RCNx where 'x' is a particular RCN (one or two digits).

5.52.3 Verification Procedures. The program can be verified by spot checking some of the output against values manually retrieved from the data base using the ORACLE UFI.

5.52.4 Error Conditions. Error messages will be printed in the output file.

5.52.5 Listings. The program listing contains comments to assist in making program changes. The program listing (RPT16.LIS) is located in <SYSSA>SAXPGM>#RPT. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.52	3. PROGRAM NAME RPT16.FOR	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.53 Program. Army Requirement Allocation Output Screen (RESULT1.INP)

5.53.1 Program Description. The Army Requirement Allocation screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 4, Review Output Data, then option 1, Screens for Reviewing Output Data, and finally as option 1.

a. Identification - The source code for the Army Requirement Allocation screen is identified by a file labeled RESULT1.INP. After being compiled through ORACLE IAG, a run file is created named RESULT1.FRM, both located in SAXPGM)*INP.

b. Functions - The output screen, Army Requirement Allocation, retrieves item information from three PJSM data base tables titled ITEM, REVTAB, and RESULT1. The purpose of this screen is for reviewing the Army production from various specified PJSM runs. None of the information that appears on this screen can be updated, inserted, or deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Title of PJSM Run (REMARKS) --- CHARACTER (40).

(c) From RESULT1 table:

(1) FY --- INTEGER (2).

(2) Package Level (PKGL) --- INTEGER (2).

(3) Army Inventory Applied (INV_APP) --- INTEGER (10).

(4) Dollar Value of Army Inventory (INV_DV) --- INTEGER (10).

(5) Army Production Applied (PROD_APP) --- INTEGER (10).

(6) Dollar Value of Army Production (PROD_DV) --- INTEGER (10).

(7) Work-Years Used (WRKYRS) --- NUMBER (F7.2).

(8) Percent of Line Used (PCT_LINE) --- NUMBER (F7.2).

(9) Shortfall Quantity (SFALL_QTY) --- INTEGER (10).

(10) Reason Code for Shortfall (RC) --- CHARACTER (1).

d. Processing -

(1) The Army Requirement Allocation screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second block, which relates to the REVTAB table, also uses a default WHERE and ORDER BY clause which is by RCN. This means the data will be displayed in ascending order by RCN.

(c) The third block, which relates to the RESULT1 table, also uses a default WHERE and ORDER BY clause which is by FY. The data, which relate to the DODIC in block 1 and the RCN in block 2, will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Army Requirement Allocation screen.

(4) From PJSM data base ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 which is designed for the user to select which PJSM run they wish to review results from. The information in block 2 is retrieved from the REVTAB table.

(6) After values are displayed in block 1 and 2, a query is executed on block 3 with the tie between the three blocks being the DODIC of the item and the RCN. The information in block 3 is retrieved from the RESULT1 table.

(7) Certain rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.53.1-1).

f. Security - The Army Requirement Allocation screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the output screens.

Army Requirement Allocation

FSC: 1320 DODIC: D505 SSN: E26900

Nomenclature: PROJ 155MM ILLUM M485 SERIES W/O FUZE

RCN Version

RCN: 11 Title: MARCH 15 PRES. BUDGET

		Army Inventory		Army Production		Work	Percent	Shortfall	
FY	Pk	Applied	Dollars	Applied	Dollars	Years	of Line	Quantity	RC
88	2	0	0	1	229	.3	0	0	U
88	3	9	2064	0	0	0	0	0	U
88	5	0	0	37	8484	10.96	0	0	U
88	7	37	8484	0	0	0	0	0	U
88	8	25	5732	0	0	0	0	0	U
88	9	23	5274	0	0	0	0	0	U
88	13	28	6420	8	1834	2.37	0	0	U

Figure 5.53.1-1. Example of Army Requirement Allocation Output Screen Filled with Information

(2) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(3) RESULT1 - contains output results from a PJSM model run. This table stores the output that is associated with the Army requirements.

5.53.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.53.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.53.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.53.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.53	3. PROGRAM NAME RESULT1.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.54 Program. Other Customer Requirement Allocation Output Screen (RESULT2.INP)

5.54.1 Program Description. The Other Customer Requirement Allocation screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 4, Review Output Data, then option 1, Screens for Reviewing Output Data, and finally as option 2.

a. Identification - The source code for the Other Customer Requirement Allocation screen is identified by a file labeled RESULT2.INP. After being compiled through ORACLE IAG, a run file is created named RESULT2.FRM, both located in SAXPGM>$INP.

b. Functions - The output screen, Other Customer Requirement Allocation, retrieves item information from three PJSM data base tables titled ITEM, REVTAB, and RESULT2. The purpose of this screen is for reviewing the other customer production from various specified PJSM runs. None of the information that appears on this screen can be updated, inserted, or deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Title of PJSM Run (REMARKS) --- CHARACTER (40).

(c) From RESULT2 table:

(1) FY --- INTEGER (2).

(2) Other Customer (SERVICE) --- CHARACTER (2).

(3) Army Inventory Applied (INV_APP) --- INTEGER (10).

(4) Dollar Value of Inventory (INV_DV) --- INTEGER (10).

(5) Army Production Applied (PROD_APP) --- INTEGER (10).

(6) Dollar Value of Army Production (PROD_DV) --- INTEGER (10).

(7) Work-Years Used (WRKYRS) --- NUMBER (F7.2).

(8) Percent of Line Used (PCT_LINE) --- NUMBER (F7.2).

(9) Shortfall Quantity (SFALL_QTY) --- INTEGER (10).

(10) Reason Code for Shortfall (RC) --- CHARACTER (1).

d. Processing -

(1) The Other Customer Requirement Allocation screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second block, which relates to the REVTAB table, also uses a default WHERE and ORDER BY clause which is by RCN. This means the data will be displayed in ascending order by RCN.

(c) The third block, which relates to the RESULT2 table, also uses a default WHERE and ORDER BY clause which is by FY then service. The data, which relate to the DODIC in block 1 and the RCN in block 2, will be displayed in order of FY and service.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Other Customer Requirement Allocation screen.

(4) From PJSM data base ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 which is designed for the user to select which PJSM run they wish to review results from. The information in block 2 is retrieved from the REVTAB table.

(6) After values are displayed in block 1 and 2, a query is executed on block 3 with the tie between the three blocks being the DODIC of the item and the RCN. The information in block 3 is retrieved from the RESULT2 table.

(7) Certain rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.54.1-1).

f. Security - The Other Customer Requirement Allocation screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the output screens.

Other Customer Requirement Allocation

FSC: 1320 DODIC: D505 SSN: E26900

Nomenclature: PROJ 155MM ILLUM M485 SERIES W/O FUZE

RCN Version

RCN: 11 Title: MARCH 15 PRES. BUDGET

FY	Sv	Inventory Applied	Inventory Dollars	Production Applied	Production Dollars	Work Years	Percent of Line	Shortfall Quantity	RC
88	OT	0	0	12000	2751474	3.56	0	0	U
90	OT	0	0	12000	3102600	3.56	0	0	U
91	OT	0	0	12000	3173879	3.56	0	0	U
92	OT	0	0	12000	3246839	3.56	0	0	U
93	OT	0	0	12000	3246839	3.56	0	0	U

Figure 5.54.1-1. Example of Other Customer Requirement Allocation Output Screen Filled with Information

(2) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(3) RESULT2 - contains output results from a PJSM model run. This table stores the output that is associated with the other customer requirements.

5.54.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.54.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.54.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.54.5 Listings. Program listing is located in <SYSSA>SAXPGM>*INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.54	RESULT2.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.55 Program. Production Summary Output Screen (RESULT3.INP)

5.55.1 Program Description. The Production Summary screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 4, Review Output Data, then option 1, Screens for Reviewing Output Data, and finally as option 3.

a. Identification - The source code for the Production Summary screen is identified by a file labeled RESULT3.INP. After being compiled through ORACLE IAG, a run file is created named RESULT3.FRM, both located in SAXPGM>$INP.

b. Functions - The output screen, Production Summary, retrieves item information from three PJSM data base tables titled ITEM, REVTAB, and RESULT3. The purpose of this screen is for reviewing the total production from various specified PJSM runs. None of the information that appears on this screen can be updated, inserted, or deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Title of PJSM Run (REMARKS) --- CHARACTER (40)

(c) From RESULT3 table:

(1) FY --- INTEGER (2).

(2) Plant Code (PLANT) --- CHARACTER (2).

(3) Line Number (LINE) --- INTEGER (3).

(4) Army Production for Test and Training (PROD_TATR) --- INTEGER (10).

(5) Army Production for War Reserve (PROD_ADOS) --- INTEGER (10).

(6) Army Work Years Used (WRKYRS_AR) --- NUMBER (F7.2); e.g., XXXXXXX.XX.

(7) Other Service Production (PROD_OTHER) --- INTEGER (10).

(8) Other Service Work Years Used (WRKYRS_OT) --- NUMBER (F7.2); e.g., XXXXXXX.XX.

d. Processing -

(1) The Production Summary screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second block, which relates to the REVTAB table, also uses a default WHERE and ORDER BY clause which is by RCN. This means the data will be displayed in ascending order by RCN.

(c) The third block, which relates to the RESULT3 table, also uses a default WHERE and ORDER BY clause which is by FY then plant. The data, which relate to the DODIC in block 1 and the RCN in block 2, will be displayed in order of FY and plant.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Production Summary screen.

(4) From PJSM data base ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 which is designed for the user to select which PJSM run they wish to review results from. The information in block 2 is retrieved from the REVTAB table.

(6) After values are displayed in block 1 and 2, a query is executed on block 3 with the tie between the three blocks being the DODIC of the item and the RCN. The information in block 3 is retrieved from the RESULT3 table.

(7) Certain rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.55.1-1).

f. Security - The Production Summary screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the output screens.

(2) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(3) RESULT3 - contains output results from a PJSM model run. This table stores the output that is associated with the total production requirements.

Production Summary

FSC: 1320 DODIC: D505 SSN: E26900

Nomenclature: PROJ 155MM ILLUM M485 SERIES W/O FUZE

RCN Version

RCN: 11 Title: MARCH 15 PRES. BUDGET

			ARMY PRODUCTION			OTHER PRODUCTION	
FY	Plant	Line	Test/Trng	War Reserve	Wrk/Yrs	Quanitity	Wrk/Yrs
88	LH	249	38000	152000	56.3	12000	3.56
89	LH	249	0	0	0	12000	3.56
90	LH	249	0	0	0	12000	3.56
91	LH	249	0	0	0	12000	3.56
92	LH	249	0	0	0	12000	3.56
93	LH	249	0	0	0	12000	3.56

Figure 5.55.1-1. Example of Production Summary Output Screen Filled with Information

5.55.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.55.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.55.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.55.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.55	RESULT3.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.56 Program. Item Summary Output Screen (RESULT4.INP)

5.56.1 Program Description. The Item Summary screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 4, Review Output Data, then option 1, Screens for Reviewing Output Data, and finally as option 4.

a. Identification - The source code for the Item Summary screen is identified by a file labeled RESULT4.INP. After being compiled through ORACLE IAG, a run file is created named RESULT4.FRM, both located in SAXPGM>#INP.

b. Functions - The output screen, Item Summary, retrieves item information from three PJSM data base tables titled ITEM, REVTAB, and RESULT4. The purpose of this screen is for reviewing the Army production, shortfalls, and ending assets from various specified PJSM runs. None of the information that appears on this screen can be updated, inserted, or deleted through this screen.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Title of PJSM Run (REMARKS) --- CHARACTER (40).

(c) From RESULT4 table:

(1) FY (FY) --- INTEGER (2).

(2) Army Production Applied (PROD_ARMY) --- INTEGER (10).

(3) Dollar Value of Army Production (DV_ARMY) --- INTEGER (10).

(4) Shortfall Quantity for Test and Training (SFALL_TATR) --- INTEGER (10).

(5) Shortfall Quantity for DOS (SFALL_ADOS) --- INTEGER (10).

(6) Shortfall Quantity for OS (SFALL_OTHER) --- INTEGER (10).

(7) Ending Assets (E_ASETS) --- INTEGER (10).

d. Processing -

(1) The Item Summary screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC.

(b) The second block, which relates to the REVTAB table, also uses a default WHERE and ORDER BY clauses which is by RCN. This means the data will be displayed in ascending order by RCN.

(c) The third block, which relates to the RESULT4 table, also uses a default WHERE and ORDER BY clause which is by FY, then plant. The data, which relate to the DODIC in block 1 and the RCN in block 2, will be displayed in order of FY and plant.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Item Summary screen.

(4) From PJSM data base ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 which is designed for the user to select which PJSM run they wish to review results from. The information in block 2 is retrieved from the REVTAB table.

(6) After values are displayed in block 1 and 2, a query is executed on block 3 with the tie between the three blocks being the DODIC of the item and the RCN. The information in block 3 is retrieved from the RESULT4 table.

(7) Certain rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results. These rules include the use of a predelete and preselect statement, in addition to answering no to the update field prompt.

e. Output - Filled screen (see figure 5.56.1-1).

f. Security - The Item Summary screen displays information which is unclassified.

g. Interfaces - All functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the output screens.

(2) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(3) RESULT4 - contains output results from a PJSM model run. This table stores the output that is associated with the percent of requirements filled, shortfalls, and ending asset levels.

Item Summary

FSC: 1320 DODIC: D505 SSN: E26900

Nomenclature: PROJ 155MM ILLUM M485 SERIES W/O FUZE

RCN Version

RCN: 11 Title: MARCH 15 PRES. BUDGET

	Army Production		Shortfall Quantities			Ending
FY	Quantity	Dollars	Test/Trng	War Reserve	Other	Assets
88	190000	43565000	0	0	0	310000
89	0	0	0	0	0	273000
90	0	0	0	45000	0	234000
91	0	0	0	125000	0	195000
92	0	0	0	150000	0	156000
93	0	0	0	150000	0	156000

Figure 5.56.1-1. Example of Item Summary Output Screen Filled with Information

5.56.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.56.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.56.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing the Display Error function key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.56.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.56	3. PROGRAM NAME RESULT4.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.57 Program. Post Processor Change Screen (RSLT_CHG.INP)

5.57.1 Program Description. The Review or Update Results screen is written using ORACLE IAF. An ORACLE process called IAG is used to create the form necessary to produce a screen. The ORACLE IAP is utilized to convert the coding into a screen. This screen is located on the PJSM Master Menu under option 2, Develop or Alter the Ammunition Program, and then option 2.

a. Identification - The source code for the Review or Update Output Results screen is identified by a file labeled RSLT_CHG.INP. After being compiled through ORACLE IAG, a run file is created named RSLT_CHG.FRM, both located in SAXPGM>$INP.

b. Functions - The output screen, Review or Update Output Results, retrieves item information from three PJSM data base tables titled ITEM, REVTAB, and RESULT4. This screen is to be used in conjunction with the PJSM Post Processor. The purpose of this screen is to alter the output from a PJSM model run by inputting a new quantity or a new dollar value for a specific item. Formulas are executed when a user inserts a new production or dollar level. When a new production value is input to the screen, two formulas are executed. The first formula multiples the new production and the unit price, subtracts the dollar value of the original production, and inserts the total into the change in dollar value field. The second formula subtracts the original production from the new production and inserts the total into the change in production field. When a new dollar value is input to the screen, two formulas are executed. The first one divides the new dollar value by the unit price, subtracts the original production, and inserts the total into the change in production field. The second one subtracts the dollar value of the original production from the new dollar value and inserts the total into the change in dollar value field. These values are stored in the RESULT4 table under the same RCN as the original run. The general item information that appears at the top of this screen cannot be update, inserted, or deleted through this screen. These functions have to be performed through the Item Descriptors screen or a special data base procedure that is found on the JSM Master Menu.

c. Input -

(1) Through JSM menu, user inputs terminal type, ORACLE identification, and password.

(2) From data base:

(a) From ITEM table:

(1) FSC --- CHARACTER (4).

(2) DODIC --- CHARACTER (4).

(3) SSN --- CHARACTER (6).

(4) Nomenclature --- CHARACTER (48).

(b) From REVTAB table:

(1) RCN --- INTEGER (2).

(2) Title of PJSM Run (REMARKS) --- CHARACTER (40).

(c) From RESULT4 table:

(1) FY --- INTEGER (2).

(2) Army Production Applied (PROD_ARMY) --- INTEGER (10).

(3) Dollar Value of Army Production (DV_ARMY) --- INTEGER (10).

(4) Dollar Value Change (DV_CHG) --- INTEGER (10).

(5) Production Change (PROD_CHG) --- INTEGER (10).

d. Processing -

(1) The Review or Update Output Results screen incorporates a few unique rules:

(a) In the first block, which pertains to the ITEM table, this screen uses a default WHERE and ORDER BY clause which is by DODIC then FSC. If a query is executed with no given information, items will be returned to the screen in DODIC order. If a particular FSC is the value being queried, the items will appear in DODIC order within that FSC. The first block also uses a series of rules that will not allow the user to delete, insert, or update general item information.

(b) The second block, which relates to the REVTAB table, also uses a default WHERE and ORDER BY clause which is by RCN. This means the data will be displayed in ascending order by RCN.

(c) The third block, which relates to the RESULT4 table, also uses a default WHERE and ORDER BY clause which is by FY. The data, which relate to the DODIC in block 1 and the RCN in block 2, will be displayed in order of FY.

(2) Read terminal type to set function keys appropriately on each user's terminal.

(3) Read ORACLE identification and password to determine if user has access to the tables being used in the Review or Update Output Results screen.

(4) From PJSM data base ITEM table, read information associated with value query was executed on. Display to input screen.

(5) After values are displayed in block 1, a query is executed on block 2 which is designed for the user to select which PJSM run they wish to review results from. The information in block 2 is retrieved from the REVTAB table.

(6) After values are displayed in block 1 and 2, a query is executed on block 3, with the tie between the three blocks being the DODIC of the item and the RCN. The information in block 3 is retrieved from the RESULT4 table.

(a) Changing information that exists in the RESULT4 table can only be accomplished in block 3. When a user wishes to change the data for a particular item, they insert either the new quantity or total dollar value.

(b) The screen will take whatever value the user inserts and compute the change from the original value. The screen will also compute the change for the value they did not enter based on the information given.

(c) When the user commits the transaction, the values for the changes are directly placed into the RESULT4 table. The user is unable to delete any information from the RESULT4 table.

(7) The rules will not allow the user to delete, insert, or update general item information, RCN information, or any output results.

e. Output - Filled screen (see figure 5.34.1-1).

f. Security - The Review or Update Output Results screen displays information which is unclassified.

g. Interfaces - The functional users of the PJSM System will only have access through the PJSM Master Menu.

h. Tables - Information is retrieved from the following tables in the PJSM data base:

(1) ITEM - contains general information for each item. Since the ITEM table contains the main identifiers for items in the data base, it is used in the majority of the output screens.

(2) REVTAB - contains important information regarding rules the PJSM will use for a model run.

(3) RESULT4 - contains output results from a PJSM model run. This table stores the output that is associated with the percent of requirements filled, shortfalls, and ending asset levels.

5.57.2 Conventions.

a. The data stored in ORACLE are case sensitive. If the same values were input in upper and lower case, ORACLE would treat them separately. The input screens are designed to insert capital letters into all of the fields which are alphabetic.

b. When designing a screen using ORACLE, the user has the option of having WHERE and ORDER BY clauses in their program. These clauses are used to group data in a particular block which relates to a data table. If a screen was designed to have a WHERE and ORDER BY clause set up by FY, the screen would organize the data for that block in ascending order by year.

5.57.3 Verification Procedures. Verification of the program can be accomplished by comparing the data that appear on the input screen with data in the corresponding ORACLE tables.

5.57.4 Error Conditions. When a user receives a message that an ORACLE error has occurred, executing a certain control key will let the user know what the error was. The most common error would be that the table or column does not exist, which simply means that the user does not have access to that table.

5.57.5 Listings. Program listing is located in <SYSSA>SAXPGM>$INP. A hardcopy of the source program is available in AMSMC-IMS-HM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.57	RSLT CHG.INP	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.58 Program. Plant Job Scheduling Model Workloading Program (PJSMWL)

5.58.1 Program Description. The PJSMWL program is written in structured FORTRAN 77. The source code contains approximately 905 lines, including comments, and is in a file named PJSMWL. This file must be compiled using the PRIME F77 compiler to call the appropriate libraries. Like the rest of the PJSM graphics programs this program is written as a module, and will not execute properly as a stand alone program. After compilation, it must then be loaded and linked using the WLLINK command file. This will load and link the program with the other required routines in PJSMDB, PJSMCOM, and PJSMCHT. The PJSMWL program consists of six subroutines which are called when the individual options are selected.

a. Identification - The source code for the program PJSMWL is identified by a file labeled PJSMWL. After being compiled through the PRIME F77 compiler, the PJSMWL.SEG version is created for use in loading and linking the other modules together.

b. Function - The program serves as a supervisor directing all supporting routines, allowing the user to select the type of workload chart desired and the style. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input - This program begins by calling three routines located in PJSMCOM:

(1) GORCLI to retrieve the user's ORACLE ID and password.

(2) GTMTYP to retrieve the terminal type.

(3) GPTSYL to retrieve the character style.

This information is used as input for the six subroutines within this program. All six subroutines require IOX (input/output unit number) and TERM (terminal type) as input.

d. Processing -

(1) Subroutine WRKMEN displays a menu for the five available workload charts. Depending on which of the five are selected, one of the following subroutines is implemented. If another option should be added to these current workloading charts, this routine will have to be changed. The menu appears as follows:

```
NO.  WORK-YEARS
 0 - EXIT
 1 - BY YEAR GROUPED BY SIGNIFICANT ITEMS FOR PLANTS
 2 - BY YEAR GROUPED BY FAMILIES FOR PLANTS
 3 - BY YEAR GROUPED BY PDIP FOR PLANTS
 4 - BY YEAR GROUPED BY SERVICE FOR PLANT
 5 - BY YEAR STRATIFIED BY LABOR TYPE

10> SELECT DESIRED PLOT >
```

If option 1 is selected, routine WRKPT1 is called. If option 2 is selected, routine WRKPT2 is called; option 3 calls WRKPT3, option 4 calls WRKPT4, and option 5 calls WRKPT5.

(2) Subroutine WRKPT1 is implemented when the user requests the chart displaying the 'work-years by year grouped by significant items for plants'. It calls on routines GYEAR and GPLANT, located in PJSMCOM, to retrieve the years requested as well as the plant(s) requested. It then fills the x and y axis labels with the appropriate titles. The program next initiates the routine GSIGIT if data are to be retrieved from the data base, or starts the routine GITEMS if the user is to enter the data. Both routines are located in PJSMCOM. The title for the chart is then filled for the specific plant(s). If the user has retrieved data from the data base, routine DBWRK1 is implemented in PJSMDB to accumulate the data and load them in the appropriate form. After the legend labels are loaded with the appropriate information, the program calls routine BGNPLT in PJSMCOM to begin the actual drawing of the plot. The information retrieved from the above mentioned procedures is passed to this routine and the chart is drawn on the screen.

(3) Subroutine WRKPT2 is implemented when the user requests the chart displaying the 'work-years by year grouped by families for plants'. It calls the routine GYEAR, located in PJSMCOM, to retrieve the years being requested as well as the routine GFAMILY, also located in PJSMCOM, to retrieve the information for the family or families being requested. The program then initiates the routine GPLANT, located in PJSMCOM, to retrieve the desired plant(s) information. It designates the x and y axis titles as well as the title for the chart. Routine DBWRK2, located in PJSMDB, is then called to accumulate the data and load them in the appropriate form. The information retrieved for the above mentioned procedures is passed to the routine BGNPLT and the chart is drawn on the screen.

(4) Subroutine WRKPT3 is implemented when the user requests the chart displaying the 'work-years by year grouped by PDIP for plants'. It calls the routine GYEAR to retrieve the appropriate years requested. If the data are to be manually inserted, the routine GPDIPS, in PJSMCOM, is called to allow the user to insert data. Otherwise, the routine DBWRK3, in PJSMDB, is called to accumulate the data from the data base and load them in the appropriate form. The routine GPLANT, in PJSMCOM, is called to retrieve the required plant. The information retrieved from the above mentioned procedures is passed to the routine BGNPLT in PJSMCOM and the chart is drawn on the screen.

(5) Subroutine WRKPT4 is implemented when the user requests the chart displaying the 'work-years by year grouped by service for plant'. This routine calls GYEAR, GSERVC, and GPLANT in PJSMCOM to retrieve the years, service, and plant information, respectively. The x and y axis labels are filled with the appropriate information as well as the title of the chart. Then the DBWRK4 routine, in PJSMDB, is called to accumulate and load the data from the data base in the appropriate form. Once all of the information is retrieved and loaded, the routine BGNPLT in PJSMCOM is called to draw the chart on the screen.

(6) Subroutine WRKPT5 is implemented when the user requests the chart displaying the 'work-years stratified by labor type'. This routine calls GYEAR and GPLANT, in PJSMCOM, to retrieve the years and plant information, respectively. The x and y axis labels and the title of the chart are loaded in the appropriate arrays. Routine DBWRK5 in PJSMDB is called to accumulate and load the required data in the appropriate form. Once all of the information is retrieved and loaded, the routine BGNPLT in PJSMCOM is used to draw the chart on the screen.

e. Output - Creates graphic charts of workloading according to specified directions.

f. Security - Information portrayed on the charts is unclassified.

5.58.2 Conventions. The program is written in structured FORTRAN 77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.58.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.58.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.58.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>SAXTST>$JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.58	3. PROGRAM NAME PJSMWL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.59 Program. Plant Job Scheduling Model Requirement Program (PJSMRQ)

5.59.1 Program Description. The PJSMRQ program is written in structured FORTRAN 77. The source code contains approximately 450 lines, including comments, and is located in a file named PJSMRQ which must be compiled using the PRIME F77 compiler to call the appropriate libraries. Like the rest of the PJSM graphics programs, this program is written as a module of the whole and will not execute properly as a stand alone program. After compilation, it must be loaded and linked using the RQLINK command file. This will load and link the program with the other required routines in PJSMDB, PJSMCOM, and PJSMCHT. The PJSMRQ program consists of two subroutines which are called when the individual options are selected.

a. Identification - The source code for the program PJSMRQ is identified by a file labeled PJSMRQ. After being compiled through the PRIME F77 compiler, the PJSMRQ.SEG version is created for use in loading and linking with the other modules.

b. Function - The program serves as a supervisor directing all supporting routines, allowing the user to select the type of requirements chart desired and the style. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input - This program begins by calling three routines located in PJSMCOM:

(1) GORCLI to retrieve the user's ORACLE ID and password.

(2) GTMTYP to retrieve the terminal type.

(3) GPTSYL to retrieve the character style.

This information is used as input for the two subroutines within this program. Both subroutines require IOX (input/output unit number) and TERM (terminal type) as input.

d. Processing -

(1) Subroutine REQMEN is a menu for the available requirements charts (currently only one). If another option should be added to the current capability, this routine would have to be modified. The menu appears as follows:

```
NO   REQUIREMENTS
 0 - EXIT
 1 - BY YEAR GROUPED BY SERVICE FOR ITEM

10> SELECT DESIRED PLOT >
```

(2) Subroutine REQPT1 is implemented when the user requests the chart displaying the 'requirements by year grouped by service for item'. It calls on routines GYEAR, GITEMS, and GSERVC to retrieve the years, items, and service respectively, as parameters of the data to be plotted. After it fills the x and y axis labels and titles with the appropriate information, the routine then calls DBINAM, in PJSMDB, to retrieve all the proper item identification data. The routine next invokes the routine DBREQ1, in PJSMDB, to retrieve and accumulate the data from the ORACLE data base and loads them in the appropriate form. Once everything is loaded, the routines UDATIN and BGNPLT, in PJSMCOM, are called to begin the actual drawing of the chart on the screen.

5.59.2 Conventions. The program is written in structured FORTRAN 77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.59.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.59.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.59.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>SAXTST>$JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.59	3. PROGRAM NAME PJSMRQ	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.60 Program. Plant Job Scheduling Model Production Program (PJSMPD)

5.60.1 Program Description. The PJSMPD is written in structured FORTRAN 77. The source code contains approximately 489 lines, including comments, and is located in a file named PJSMPD. This file must be compiled using the PRIME F77 compiler which calls the appropriate libraries. Like the rest of the PJSM graphics programs, this program is written as a module of the whole system, and will not execute properly as a stand alone program. After compilation, it must be loaded and linked using the PDLINK command file which will load and link the program with the other required routines in PJSMDB, PJSMCOM, and PJSMCHT. The PJSMPD program consists of two subroutines which are called when the individual options are selected.

a. Identification - The source code for the program PJSMPD is identified by a file labeled PJSMPD. After being compiled through the PRIME F77 compiler, the PJSMPD>SEG is created for use in loading and linking with the other modules.

b. Functions - The program serves as a supervisor directing all supporting routines, allowing the user to select the type of production chart desired and the style. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input - This program begins by calling three routines located in PJSMCOM:

(1) GORCLI to retrieve the user's ORACLE ID and password.

(2) GTMTYP to retrieve the terminal type.

(3) GPTSYL to retrieve the character style.

This information is used as input for the three subroutines within this program. All three subroutines require IOX (input/output unit number) and TERM (terminal type) as input.

d. Processing -

(1) Subroutine PRDMEN displays a menu for the available production charts. If another option should be added to the current capability, this routine would have to be modified to include the new option. The menu appears as follows:

NO.	PRODUCTION QUANTITY
0	EXIT
1	BY YEAR GROUPED BY SIGNIFICANT ITEMS 'FOR PLANTS'
2	SHORTFALL QUANTITY BY YEAR BY ITEMS

> SELECT DESIRED PLOT >

(2) Subroutine PRDPT1 is implemented when the user requests the chart displaying the 'production by year grouped by significant items for plants'. It calls on routines GPLANT and GITEMS, in PJSMCOM, to retrieve the desired plants and items, respectively. This routine fills the x and y axis labels and plot titles with the appropriate information. It then invokes the routine DBPRD1, in PJSMDB, to retrieve and accumulate the data from the ORACLE data base and loads them in the appropriate form. Once everything is loaded, the routines UDATIN and BGNPLT in PJSMCOM are called to begin the actual drawing of the chart on the screen.

(3) Subroutine PRDPT2 is implemented when the user requests the chart displaying 'production shortfall quantity by year by items'. It calls on the routine GYEAR, in PJSMCOM, to retrieve the years requested. This routine then loads the x and y axis labels and the plot title information. It then invokes the routine DBPRD2, in PJSMDB, to retrieve and accumulate the data from the ORACLE data base and loads them in the appropriate form. Once everything is loaded, the routines UDATIN and BGNPLT in PJSMCOM are called to begin the actual drawing of the chart on the screen.

e. Output - Creates graphic charts of production according to specified directions.

f. Security - Information portrayed on the charts is unclassified.

5.60.2 Conventions. The program is written in structured FORTRAN 77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.60.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.60.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.60.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>SAXTST>#JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.60	PJSMPD	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.61 Program. Plant Job Scheduling Model Dollar Program (PJSMDL)

5.61.1 Program Description. The PJSMDL program is written in structured FORTRAN 77. The source code contains approximately 1,229 lines, including comments, and is located in a file named PJSMDL. This file must be compiled using the PRIME F77 compiler to call the appropriate libraries. Like the previous three PJSM graphics programs, this program is written as a module and will not execute properly as a stand alone program. After compilation, it must then be loaded and linked using the DLLINK command file which will load and link the program with the other required routines in PJSMDL, PJSMCOM, and PJSMCHT. The PJSMWL program consists of nine subroutines which are called when the individual options are selected.

a. Identification - The source code for the program PJSMDL is identified by a file labeled PJSMDL. After being compiled through the PRIME F77 compiler, the PJSMDL.SEG version is created for use in loading and linking the other modules together.

b. Functions - The program serves as a supervisor directing all supporting routines, allowing the user to select the type of expenditure (dollars) chart desired and the style. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input - This program begins by calling three routines located in PJSMCOM:

(1) GORCLI to retrieve the user's ORACLE ID and password.

(2) GTMTYP to retrieve the terminal type.

(3) GPTSYL to retrieve the character style.

This information is used as input for the nine subroutines within this program. All six subroutines require IOX (input/output unit number) and TERM (terminal type) as input.

d. Processing -

(1) Subroutine DOLMEM displays a menu for the eight dollar expenditure charts now available. Depending on which of the eight is selected, one of the following eight subroutines is implemented. If another option should be added to these current dollar expenditure charts, this routine will have to be changed. The menu appears as follows:

NO. DOLLAR VALUES
0 - Exit
1 - BY YEAR GROUPED BY FAMILIES FOR PLANTS
2 - BY YEAR GROUPED BY PDIP'S FOR PLANTS
3 - BY FAMILY FOR PLANT-YEAR
4 - BY YEAR OF TOA, HARDWARE, PRODUCTION BASE
5 - BY PLANTS GROUPED BY YEAR FOR PLANT TYPE
6 - BY YEAR GROUPED BY GOGO, GOCO, AND COCO
7 - BY YEAR OF POM AND TOA
8 - BY YEAR OF PRODUCTION BASE, WAR, RESERVES, AND TRAINING

› SELECT DESIRED PLOT ›

(2) Subroutines DOLPT1 is implemented when the user requests the chart displaying the 'dollars expended by year grouped by families for plants'. Routines GYEAR and GFAMILY, located in PJSMCOM are called to retrieve the years requested as well as the family information and selection requested. The x and y axis and legend labels are then filled with the appropriate titles. The plot title is loaded next. The subroutine DBDOL1 in PJSMDB is used to retrieve and accumulate the data and load them in the appropriate format. Then the program calls the routine BGNPLT in PJSMCOM to begin the actual drawing of the plot.

(3) Subroutine DOLPT2 is implemented when the user requests the chart displaying the 'dollar expenditure by year grouped by PDIPs for plants'. It calls on the routines GYEAR, GPDIPS, and GPLANT to retrieve the years requested, the PDIP selection and information, and the plants desired respectively. DOLPT2 then builds the x and y axis labels, and the title information and loads the information accordingly. The routine DBDOL2 in PJSMDB is used to retrieve and accumulate the data and load them in the appropriate format. Then the program then calls the routine BGNPLT in PJSMCOM to begin the actual drawing of the plot.

(4) Subroutine DOLPT3 is implemented when the user requests the chart displaying the 'dollar expenditure by family for plant year'. Calling on the routines GYEAR and GPLANT to retrieve the years and the plants requested, it fills the x and y label and title information appropriately. DOLPT3 then calls the routine DBDOL1 to retrieve and accumulate the appropriate information requested. The routine BGNPLT, in PJSMCOM, is called to begin the actual drawing of the plot.

(5) Subroutine DOLPT4 is implemented when the user requests the chart displaying the 'dollar expenditure by year of TOA, Hardware, Production Base'. It first calls the routine GYEAR to retrieve the desired years, then fills the x and y labels, the legend labels, and plot title information accordingly. The routine DBDOL4, in PJSMDB, is used to retrieve and accumulate the desired data and load them in the appropriate format. Routine BGNPLT, in PJSMCOM, is called to begin the actual drawing of the plot.

(6) Subroutine DOLPT5 is implemented if the user requests the chart displaying 'dollar expenditure by plants grouped by year for plant type'. It first calls the routine GYEAR, in PJSMCOM, to retrieve the desired year information, then fills the x and y axis labels, the legend labels, and plot title information accordingly. The routine DBDOL5, in PJSMDB, is called to retrieve and accumulate the desired data and load them in the appropriate format. Routine BGNPLT, in PJSMCOM, is then used to begin the actual drawing of the chart.

(7) Subroutine DOLPT6 is implemented if the user requests the chart displaying 'dollar expenditure by year grouped by GOGO, GOCO, COCO'. This routine first calls the routine GYEAR, in PJSMCOM, to retrieve the desired year information, then fills the x and y axis labels, the legend labels, and the plot title information accordingly. The routine DBDOL6, in PJSMDB, is called to retrieve and accumulate the desired data and load them in the appropriate format. Routine BGNPLT, in PJSMCOM, is used to begin the actual drawing of the plot.

(8) Subroutine DOLPT7 is implemented if the user requests the chart displaying 'dollar expenditure of POM and TOA'. This routine just calls the routine GYEAR, in PJSMCOM, to retrieve the desired year information, then fills the x and y axis labels, the legend labels, and the plot title information accordingly. The routine DBDOL7, in PJSMDB, is called to retrieve and accumulate the desired data and load them in the appropriate format. Routine BGNPLT, in PJSMCOM, is then called to begin the actual drawing of the chart.

(9) Subroutine DOLPT8 is implemented if the user requests the chart displaying 'dollar expenditure by year of production base, war reserves, and training'. This routine first calls the routine GYEAR, in PJSMCOM, to retrieve the desired year information, then fills the x and y axis labels, the legend labels and the plot title information accordingly. The routine DBDOL8, in PJSMDB, is then called to retrieve and accumulate the desired data and load them in the appropriate format. Routine BGNPLT, in PJSMCOM, is called to begin the actual drawing of the chart.

e. Output - Creates graphic charts of dollar expenditure according to specified directions.

f. Security - Information portrayed on the charts is unclassified.

5.61.2 Conventions. The program is written in structured FORTRAN 77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.61.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.61.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.61.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>SAXTST>$JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.61	3. PROGRAM NAME PJSMDL	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.62 Program. Plant Job Scheduling Model Data Base Program (PJSMDB)

5.62.1 Program Description. The PJSMDB.FOR program is written as a precompiled form of structured F77 and contains approximately 2,233 lines of code. This program consists of 22 subroutines and must be precompiled using the ORACLE FORTRAN host language precompiler which replaces embedded SQL statements with calls to library routines. The precompiler generates an intermediate file PJSMDB.F77 which is compiled with the PRIME F77 compiler. The PRIME BIND linker utility is used to produce the executable routines in the program PJSMDB as requested from the four major programs.

a. Identification - The source code for the program PJSMDB is identified by a file labeled PJSMDB.FOR. After being compiled with the ORACLE libraries, the F77 source code is produced in a file identified as PJSMDB.F77. This program is then compiled to create the segmented version PJSMDB.SEG for use in loading and linking the modules together.

b. Function - The program is actually one module of the entire graphics program system. It consists of 22 subroutines each of which are executed according to the particular chart requested. Each subroutine performs a specific data base extract under the constraints of the requested parameters. This program can not be executed as an independent program. Its function is to be used in conjunction with the PJSMWL, PJSMDL, PJSMRQ, or the PJSMPD program. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input - The input required is dependent upon the chart requested. Following is the input required for each subroutine and the variable names they are passed in:

(1) Subroutine DBDOL1 requires IY1 (first FY year), NY (number of years), NFAM (array of family codes), NF (number of families in array), PLTNO (plant code), and RCN1 (revision control number).

(2) Subroutine DBDOL2 requires IY1 (first FY year), NY (number of years), RCN1 (revision control number), and PLTNO (plant code).

(3) Subroutine DBDOL4 requires IY1 (first FY year), NY (number of years), and RCN1 (revision control number).

(4) Subroutine DBDOL5 requires IY1 (first FY year), NY (number of years), PLTTYP (plant type (GOGO, GOCO, COCO)), and RCN1 (revision control number).

(5) Subroutine DBDOL6 requires IY1 (first FY year), NY (number of years), and RCN1 (revision control number).

(6) Subroutine DBDOL7 requires IY1 (first FY year), NY (number of years), and RCN1 (revision control number).

(7) Subroutine DBDOL8 requires IY1 (first FY year), NY (number of years), and RCN1 (revision control number).

(8) Subroutine DBINAM requires IDX (item identifier name) and ICODE (item identifier type).

(9) Subroutine DBGPLT requires no parameters to be initiated.

(10) Subroutine DBLOGI requires NAME (the user's ORACLE name) and OPASS (the user's ORACLE password).

(11) Subroutine DBLOGO requires no parameters to be initiated.

(12) Subroutine DBPRD1 requires IY1 (first FY year), NY (number of years), DODIC (array of items), NITM (number of items), PLTNO (plant code), and RCN1 (revision control number).

(13) Subroutine DBPRD2 requires IY1 (first FY year), NY (number of years), RCN1 (revision control number), IMODE (last returned item), and NITM (maximum number of items requested).

(14) Subroutine DBREQ1 requires IY1 (first FY year), NY (number of years), NSERV (array of service codes requested), NS (number of service codes), DODIC1 (DOD identification requested), and RCN1 (revision control number).

(15) Subroutine DBSIGC requires PLTNO (plant code), RCN1 (revision control number), IY1 (first FY year), IY2 (last FY year), and NITM (maximum number of items).

(16) Subroutine DBSIGQ requires PLTNO (plant code), RCN1 (revision control number), IY1 (first FY year), IY2 (last FY year), and NITM (maximum number of items).

(17) Subroutine DBSIGW requires PLTNO (plant code), RCN1 (revision control number), IY1 (first FY year), IY2 (last FY year), and NITM (maximum numoer of items).

(18) Subroutine DBWRK1 requires IY1 (first FY year), NY (number of years), DODIC (array of selected items), NITM (number of items), PLTNO (plant code), and RCN1 (revision control number).

(19) Subroutine DBWRK2 requires IY1 (first year), NY (number of years), NF (number of families in family array), PLTNO (plant code), and RCN1 (revision control number).

(20) Subroutine DBWRK3 requires IY1 (first FY year), NY (number of years), RCN1 (revision control number), and PLTNO (plant code).

(21) Subroutine DBWRK4 requires IY1 (first FY year), NY (number of years), SERV (array of selected services), NS (number of services), RCN1 (revision control number), and PLTNO (plant code).

(22) Subroutine DBWRK5 requires IY1 (first FY year), NY (number of years), RCN1 (revision control number), and PLTNO (plant code).

d. Processing - The PJSMDB program utilizes the input specified above as constraints for extracting the appropriate data from the ORACLE data base which are then passed to the calling routines. The processing logic for each subroutine is given below:

(1) Subroutine DBDOL1 selects and accumulates its data in one of two ways. If the user selected a specific plant it extracts the DV_ARMY value and DV_OTHER values from the RESULT3 table for the given input parameters and sums them to create a single value for each family and year. If the user selected the option which is a total of all plants, it extracts the DV_PROD values from RESULT5 for the given input parameters. With either option, these values are then divided by 1,000 and converted to integer values and placed in DATA (NF, NY) which is the output data array diminished by number of families and number of years. This array is then feed back to the calling routines DOLPT1 in PJSMDL.

(2) Subroutine DBDOL2 selects and accumulates its data in one of two ways. If the user selected a specific plant, it then extracts the DV_PROD values from RESULT5 for the given input parameters and sums them to create a single value for each package and year. If the user selected the option which is the total of all plants, it extracts the DV_PROD for all plants for the given input parameters. Regardless of the option entered, the routine then goes

thru a process to eliminate all zero packages, divides all values by 1,000 and converts them to integer values, then sorts package totals in each year in descending order. For legend purposes, the package titles are then extracted from the PACKAGE table and the last package (8) will be the sum of all other packages. The values are then fed into the DATA array diminished 8 by 20. This array, as well as N (number of non-zero packages) and PDIPNM (legend titles), is returned to the calling routine DOLPT2 in PJSMDL.

(3) Subroutine DBDOL4 extracts from the GOALS table the ACTIVITY_I and ACTIVITY_II values for the given input parameters. It then divides all values by 1,000 and converts them to integers. The output data array, DATA, dimensioned three by NY (number of years), is passed these values into positions one and two. The values in position one and two are summed for each year to arrive at the third placed in the third parameter for each year. This DATA array is then fed back to the calling routine DOLPT4 in PJSMDL.

(4) Subroutine DBDOL5 extracts and accumulates by plant type from the RESULT6 table the DV_ARMY and DV_OTHER values for the given input parameters. These values are then divided by 1,000 and converted to integer values and fed into the output data array DATA, dimensioned by NY (number of years) with the second dimension set to 20. However, only three positions of the 20 will be used. The DATA array, as well as NP (number of plants in group) and PLTNM (plant codes for name) are passed back to the calling routine DOLPT5 in PJSMDL.

(5) Subroutine DBDOL6 extracts and accumulates by plant type from RESULT6 the DV_PROD value from the given input parameters. These values are then divided by 1,000 and converted to integer values and fed into the output data array, DATA, dimensioned three (1 for each plant type) by NY (number of years). The DATA array is passed back to the calling routine DOLPT in PJSMDL.

(6) Subroutine DBDOL7 extracts and accumulates DV_PROD values form the RESULT5 table by year and divides by 1,000. Also extracted and accumulated is the ACTIVITY_I values from the GOALS table by year and multiplied by 1,000. These values are then fed into the output data array, DATA, dimensioned by two (position one being the DV_PROD values, position two being TOA values), and NY (number of years). This array is then passed back to the calling routine DOLPT7 in PJSMDL.

(7) Subroutine DBDOL8 extracts data from two tables. The first select statement extracts and accumulates the ACTIVITY_II values from the GOALS table. The second select statement extracts and calculates data as follows: SUM((PROD_TATR/(PROD_TATR+PROD_ADOS))*DV_ARMY) and extracts and accumulates the DV_ARMY and DV_OTHER values for the give input constraints. All values are then divided by 1,000 and converted to integer values. The data are then fed into the output data array, DATA, dimensioned by four (positions one thru four, respectively, are training. DOS, production base, and other service) and NY (number of years). The DATA array is passed back to the calling routine DOLPT8 in PJSMDL.

(8) Subroutine DBINAM extracts from the ITEM table and the information pertaining to the specific item requested by the input data. The item number, standard study number, the Federal Supply Class, DOD Identification Code, National Stock Number, and item sequence number is passed back to the calling routine in the following variables respectively: ITEM1, SSN1, FSC1, DODIC1, NSN1, and ISN1.

(9) Subroutine DBGPLT is another generic routine that can be implemented from any calling routine. It extracts from the PLANT table, the plant codes, plant names, the length of the plant names, and the number of plants returned. These data are placed in the following variables respectively: PCODE, PNAME, PLEN, and NPLT, and passed back to the calling routine.

(10) Subroutine DBLOGI is implemented each time any graphics program is implemented and the user wishes to extract data from the ORACLE data base. This routine does nothing other than log the user into ORACLE and verify the user's password. It returns nothing other than an error code if needed. Control is then returned back to the calling routine.

(11) Subroutine DBLOGO is called whenever the user exits the program and has been attached to the ORACLE data base. This routine 'commits' and 'releases' the data base.

(12) Subroutine DBPRD1 selects and accumulates the PROD_OTHER, PROD_TATR, and PROD_ADOS values and divided them by 1,000 and converts them to integers. The values are fed into the output data array entitled DATA, dimensioned by NITMS (number of items) and NY (number of years). This data array is passed back to the calling routine PRDPT1 in PJSMPD.

(13) Subroutine DBPRD2 selects DODICs from RESULT3 table where the sum of the values SFALL_OTHER, SFALL_TATR, and SFALL_ADOS from RESULT4 is greater than zero. It selects only those DODICs and orders them in descending order of those sums. It then invokes another select statement, and selects and accumulates the values SFALL_OTHER, SFALL_TATR, and SFALL_ADOS for the DODIC previously flagged in the calculated order. It then divides these by 1,000. These values are passed into the output array, DATA, dimensioned by NITM (maximum number of items) and NY (number of years). The DATA array, as well as IMODE (accumulative total of SFALL items returned), NITM (number of items returned), and DODIC (array of DODIC numbers), is passed back to the calling routine PRDPT2 in PJSMPD.

(14) Subroutine DBREQ1 - If the user selected only the Army requirements, this routine selects and accumulates the QUANTITY value from the REQTS_ARMY of the selected DODIC. If the user selected any other option; i.e., a particular service or all services, the routine selects and accumulates the QUANTITY value from the REQTS_OTHER table for the selected service and DODIC. These values are then fed into the output data array, DATA, dimensioned by NS (number of services) and NY (number of years). This array is then passed back to the calling routine REQPL1 in PJSMRQ.

(15) Subroutine DBSIGD is called from the routine GSIGIT in PJSMCOM. Its function is to select and order, in descending order, the dollar value of items produced at a selected plant. It selects and accumulates the DV_ARMY values from the RESULT3 table. Once the values are summed and placed in descending order, the respective DODICs are placed into the DODIC array in the same order. This array, diminished by the NITM (number of items), is fed back to the calling routine along with the variable NITM.

(16) Subroutine DBSIGQ is called from the subroutine GSIGIT in PJSMCOM. Its function is to select and order, in descending order, the QUANTITY value of items produced at a selected plant. It selects and accumulates the PROD_OTHER, PROD_TATR, and PROD_ADOS values from the RESULT3 table. Once the values are summed and placed in descending order, the respective DODICS are placed into the DODIC array in that same order. This array, dimensioned by NITM (number of items), is passed back to the calling routine along with the variable NITM.

(17) Subroutine DBSIGW is called from the routine GSIGIT in PJSMCOM. Its function is to select and order, in descending order, the work-year values of items produced at a selected plant. It selects and accumulates the WRKYRS_AR and WRKYRS_OT values from the RESULT3 table. Once the values are summed and placed in descending order, the respective DODICs are placed into the DODIC array

in the same order. This array is dimensioned by NITM (number of items) and passed back to the calling routine along with the variable NITM.

(18) Subroutine DBWRK1 provides the eight most significant producers at each plant or a total of all plants, and a ninth value which is the sum of all other items not included in the first eight. This routine selects and accumulates the WRKYRS_AR and WRKYRS_OT values from the RESULT3 table for the given DODICs and plant(s). These values are converted to integers and placed in the output data array, DATA, dimensioned by NITM+1 (number of items+1) and NY (number of years). This array is then passed back to the calling routine DOLPT1 in PJSMWL.

(19) Subroutine DBWRK2 provides the nine most significant families of items produced for a given plant or for the total of all plants. Also the user is allowed to select the families desired. This routine selects and accumulates the WRKYRS_AR and the WRKYRS_OT values from the RESULT3 table. These values are converted to integer values and placed in the output data array, DATA, dimensioned by NF (number of families) and NY (number of years). This array is then passed back to the calling routine WRKPT2 in PJSMWL.

(20) Subroutine DBWRK3 provides the seven most significant PDIP packages produced for a given plant or for all plants totaled and a total for all other packages not included in the first seven. The user is also allowed to select the packages desired. This routine selects and accumulates the WRKYRS value from the RESULT5 table. These values are converted to integers and loaded into an intermediate array, PDIP. This array is then sorted to eliminate zero packages. The routine then places seven most significant packages, as well as one value for the total of the rest, into the output data array, DATA, dimensioned eight by 20. This array, as well as variable N (number of non-zero PDIPs) and PDIPNM (legend name of data), is then passed back to the calling routine WRKPT3 in PJSMWL.

(21) Subroutine DBWRK4 provides the work-years expended by year, by service for a plant, or a total of all plants. This routine selects and accumulates the WRKYRS values from RESULT5 according to the user given constraints. These values are converted to integer values and placed into the output data array, DATA, dimensioned by NS (number of services) and NY (number of years). This data array is passed to the calling routine WRKPT4 in PJSMWL.

(22) Subroutine DBWRK5 provides the labor utilization by plant stratified by each labor type; i.e., production over head, non-production overhead, and direct labor; and a total of all three for a total work-year value for each year. The routine selects and accumulates the WRKYRS values from the RESULT6 table to obtain the direct labor values for a given plant. It then selects the OMA_OH values from the STAFF table to obtain the production overhead values by selecting and accumulating the WRKYRS_TOT values from the RAMP_PROD table the prior year direct labor data are retrieved. The routine then selects the PROD_OH_FAC and NON_PROD_OH_FAC values from the PLANT table needed to calculate the respective values. The needed data elements are calculated as follows: Direct labor = direct labor; production overhead = direct labor * production overhead factor; non-production overhead = production overhead + direct labor + OMA; OMA = OMA. These values are converted to integer values and placed in the output data array, DATA, dimensioned by four (one position for each value mentioned previously) and NY (number of years). The legend titles are fed into an array entitled PNAME dimensioned by four. These two arrays are passed back to the calling routine WRKPT5 in PJSMWL.

e. Output - Creates no output in report form.

f. Security - All information generated is unclassified.

5.62.2 Conventions. The program is written in structured F77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.62.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.62.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.62.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>SAXTST>$JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID 5.62	3. PROGRAM NAME PJSMDB	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.63 Program. Plant Job Scheduling Model Charting (PJSMCHT)

5.63.1 Program Description. The PJSMCHT program contains the structured F77 code for seven display routines. These routines use the Computer Associates Inc. DISSPLA graphics software to generate the actual graphic charts requested. The program contains approximately 1,560 lines of source code and is located in a file named PJSMCHT. This program, which must be compiled with the PRIME F77 compiler, is loaded and linked with each of the four basic programs. It is a module and not a stand alone program.

a. Identification - The source code for the program PJSMCHT is identified by a file labeled PJSMCHT. After being compiled through the F77 compiler, the PJSMCHT.SEG version is created for use in loading and linking the other modules together.

b. Function - The program is used in conjunction with the four basic programs. It is considered a module and is not run in itself. When executing one of the four main PJSM graphics programs, this program is called by the program being used. Its function is to call the appropriate subroutine the user requests. There are seven options and seven subroutines, one for each of the following:

(1) A Stacked Bar Chart.

(2) A Grouped Bar Chart.

(3) A Grouped Bar Chart with a Total Line.

(4) A Line Chart.

(5) A Pie Chart.

(6) A Horizontal Bar Chart.

(7) A Tabular Listing.

c. Input - The program requires data to feed the parameters of the display routines. These data are received from the four main programs as the user selects the desired chart type.

d. Processing -

(1) Subroutine STKBAR - invoked if the user requests a stacked bar style chart. The appropriate parameters are taken from the major program and utilized to form a chart.

(2) Subroutine GRPBAR - invoked if the user requests a grouped bar chart. The appropriate parameters are taken from the major program and utilized to form a chart.

(3) Subroutine GTOBAR - invoked if the user requests a grouped bar chart with a total line chart. The appropriate parameters are taken from the major program and utilized to form a chart.

(4) Subroutine LINCHT - invoked if the user requests a line chart. The appropriate parameters are taken from the major program and utilized to form a chart.

(5) Subroutine PIECHT - invoked if the user requests a pie chart. The appropriate parameters are passed from the major program and utilized to form a chart.

(6) Subroutine HORBAR - invoked if the user requests a horizontal bar chart. The appropriate parameters are passed from the major program and utilized to form a chart.

(7) Subroutine TABLST - invoked if the user requests a tabular listing of the data. The appropriate parameters are passed from the major program and utilized to form a chart.

e. Output - Creates no output in report form.

f. Security - All information generated is unclassified.

5.63.2 Conventions. The program is written in structured FORTRAN 77. The program does not create any output in report form, but produces data arrays that are passed to the calling routines. This program is a module of the whole and independently serves no purpose.

5.63.3 Verification Procedures. This program can be verified by spot checking some of the output reflected in charts against values manually retrieved from the data base using the ORACLE UFI.

5.63.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur it will be reflected in the plotted chart.

5.63.5 Listings. The program listing contains many comments to assist in making program changes. The program listing PJSMDB.FOR is located in <CAS2SA>S4XTST>$JSM>PJSM.

PROGRAM REVISION		1. DATE
For use of this form, see TB 18-111; the proponent agency is DCSOPS.		30 Nov 87
2. PROGRAM ID	3. PROGRAM NAME	
5.63	PJSMCHT	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.64 Program. Plant Job Scheduling Model Common (PJSMCOM)

5.64.1 Program Description. The PJSMCOM is written in structured FORTRAN F77. The source code contains approximately 1,792 lines, including comments, and is located in a file named PJSMCOM. This file must be compiled using the PRIME F77 compiler to call the appropriate libraries. Like the rest of the PJSM graphics programs this program is written as a module and will not execute properly as a stand alone program. After compilation, it must be loaded and linked using one of the four main programs WLLINK, RQLINK, DLLINK, or PKLINK. It will then be loaded and linked with these programs and other required routines in PJSMDB, PJSMWL, PJSMDL, PJSMPD, and PJSMRQ. The PJSMCOM program consists of 24 subroutines which are called when the different options are selected.

a. Identification - The source code for the program PJSMCOM is identified by a file labeled PJSMCOM. After being compiled through the F77 compiler, the PJSMCOM.SEG version is created for use in loading and linking the other modules together.

b. Function - The program serves as a program being directed by one of the four supervisor programs. It contains common routines which may be selected by any of the supervisors. Refer to Annex F for a cross-reference of subroutines in each of the PJSM graphics modules.

c. Input -

(1) Subroutine ADDDLL requires no input.

(2) Subroutine BGNPLT requires: IOX (input-output unit number), TERM (terminal number), MODL (module from which routine was called), and IMODE (process mode - 0 = inquiry, other = plot only).

(3) Subroutine GETFAM requires NUM (the family number).

(4) Subroutine GETPLN requires ICODE (0: given number, return the name, or, 1: given number, return the number), NUM (the plant number if ICODE = 0), and NAME (the plant name if ICODE = 1).

(5) Subroutine GETPTY requires IOX (input-output unit number), TERM (terminal type), and GPT (current plot type number).

(6) Subroutine GFAMILY requires IOX (input-output unit number), TERM (terminal number), and MODL (module from which routine is called).

(7) Subroutine GITEMS requires IOX (input-output unit number), TERM (terminal type), MODL (module from which routine was called), NIM (maximum number of items to select), and IMODE (0-ask for user input else use set parameters).

(8) Subroutine GORCLI requires IOX (input-output unit number).

(9) Subroutine GPDIPS requires IOX (input-output unit number), TERM (terminal type), and MODL (module from which routine was called.

(10) Subroutine GPLANT requires IOX (input-output unit number), TERM (terminal type), and MODL (module from which routine was called).

(11) Subroutine GPTSYL requires IOX (input-output unit number) and MODL (module from which routine was called).

(12) Subroutine GSERVE requires IOX (input-output unit number), TERM (terminal type), and MODL (module from which routine was called).

(13) Subroutine GSIGIT requires IOX (input-output unit number), TERM (terminal type), MODL (module from which routine was called), PLTNO (plant code), RCN (revision control number), IY1 (first FY year), IY2 (last FY year), NITM (maximum number of items to select), and IMODE (0-ask for user input else use set parameters).

(14) Subroutine GTMTYP requires IOX (input-output unit number) and MODL (module from which routine was called).

(15) Subroutine GUPDAT requires IOX (input-output unit number), TERM (terminal type), and MODL (module from which routine was called).

(16) Subroutine GYEAR requires IOX (input-output unit number), TERM (terminal type), MODL (module from which routine was called), and NYR (maximum number of years on plot).

(17) Subroutine LSTDAT requires IOX (input-output unit number) and MODL (module from which routine was called).

(18) Subroutine MYPIE requires no input.

(19) Subroutine MYSPEC requires no input.

(20) Subroutine NOTDON requires IOX (input-output unit number) and MODL (module from which routine was called).

(21) Subroutine PLOTIT requires IOX (input-output unit number) and MODL (module from which routine was called).

(22) Subroutine SETCOL requires ICOL (color index number).

(23) Subroutine ZAESTH requires no input.

(24) Subroutine ZREADI requires IUN (input unit number).

d. Processing -

(1) Subroutine ADDDLL is called from each of the charting routines in the program PJSMCHT. The function of this routine is to check for user entered chart labels (legend, axis, titles) and be sure there is a '$' appended at the end of the character string. If there is not a '$', the routine adds one. These '$'s are required by DISSPLA to correctly utilize DISSPLA's 'self terminating string' option.

(2) Subroutine BGNPLT is implemented by the routines DOLPT1, DOLPT2, DOLTP3, DOLPT4, DOLPT5, DOLTP6, DOLPT7, and DOLPT8 in PJSMDL; WRKPT1, WRKPT2, WRKPT3, WRKPT4, and WRKPT5 in PJSMWL; PRKPT1 and PRKPT2 in PJSMPD; and REQPT1 in PJSMPQ. The function of this routine is to begin the actual plotting by displaying the PLOT/UPDATE menu or directly plotting the data by calling the routine PLOTIT in PJSMCOM. The PLOT/UPDATE menu appears as follows.

```
NO.  MENU OPTION
 0 - EXIT
 1 - DRAW PLOT
 2 - UPDATE PLOT PARAMETERS
 3 - REVIEW PLOT PARAMETERS

21> ENTER OPTION NUMBER>
```

(a) If option '0' is selected, the routine returns to the previously called menu by using the variable MODL (module from which routine is called).

(b) If option 1 is selected, the routine PLOTIT is called to draw the plot.

(c) An option 2 selection calls the routine GUPDAT, in PJSMCOM, to allow the user to change the plot parameters.

(d) An option 3 selection calls the routine LSTDAT, in PJSMCOM, to allow the user to review the current plot parameters.

(3) Subroutine GETFAM is a short routine whose only function is to return a family name (FAM) and the number of characters in the selected family name (NCH). Stored in a data statement are nine family names coded in a manner which DISSPLA can recognize as denoting upper and lower case. These will be used by the plots in legends.

(4) Subroutine GETPLN is a short routine that retrieves a number or name for a selected plant and returns it to the calling routine. GETPLN is called from GPLANT in PJSMCOM. GETPLN returns NUM (the plant number if ICODE = 1), PCODE (plant two letter code), NAME (the plant name if ICODE = 0), and NCH (number of characters in the name defined only if ICODE = 0).

(5) Subroutine GETPTY is a routine that displays on the screen the current (or default) chart type and the possible options as shown below:

```
CURRENT PLOT TYPE NUMBER = 4
NO.  MENU OPTION
 0 - EXIT NUMBER UNCHANGED
 1 - STACKED BAR CHART
 2 - GROUPED BAR CHART
 3 - GROUPED BAR CHART & TOTAL LINE
 4 - ONE OR TWO AXIS LINE PLOT
 5 - PIE CHART
 6 - HORIZONTAL STACKED BAR CHART
 7 - TABULAR LISTING TO SCREEN

SELECT MENU OPTION >
```

If the user should indicate an option number not listed, the menu returns a message that states that an invalid entry has occurred. The option selected is returned in the variable GPT (plot type number) and returned to the calling routine GUPDAT in PJSMCOM.

(6) Subroutine GFAMLY is called from DOLPT2 in PJSMDL and WRKPT2 in PJSMWL. GFAMLY implements the routine GETFAM and together they display to the user a list of the families and the ability to select any combination or all of the families to be displayed on the chart. The menu is displayed below:

```
 0  ALL FAMILIES
 1  S<MALL & .50 CALIBER>$
 2  40<MM - 120MM & 4.2IN>$
 3  120<MM - 3IN>$
 4  R<OCKETS>$
 5  G<RENADES> & <MINES>$
 6  S<IGNALS & SIMULATORS>$
 7  N<ON> - AAO$
 8  M<ISC HARDWARE>$
 9  C<OMPONENTS>$

21>  ENTER ALL FAMILY NUMBERS DESIRED>
```

The routine verifies that a valid option was entered. If not, the message 'Invalid entry. Reenter' is displayed. The routine then calls the routines ZREADI in PJSMCOM to read the entered options in a free format manner and converts them to integers. It then returns NFAM (array of selected family numbers), N (number of family groups selected), and FAMNM (array of family name) to the calling routines.

(7) Subroutine GITEMS is called from GSIGIT in PJSMCOM, WRKPT1 in PJSMWL, PRDPT1 in PJSMPD, and REQPT1 in PJSMRQ when the user desires a chart concerning significant items. This routine displays a list of the different item identifiers as follows:

```
NO.  IDENTIFICATION MODE
 1 - DODIC
 2 - ITEM NUMBER
 3 - LEGEND BY SSN
 4 - LEGEND BY DODAC
 5 - LEGEND BY NSN
 6 - LEGEND BY ISN
 7 - LEGEND BY NOMENCLATURE (1-31)

21> ENTER MENU NUMBER>
```

It then calls the subroutine ZREADI in PJSMCOM to read the free formatted input and convert to integers. GITEMS verifies that a valid option was entered. If not, a message appears stating that an invalid option was entered and you must reenter. It utilizes the user input selections and then calls the routine DBINAM in PJSMDB to retrieve all the item identification data desired. GITEMS takes this information and returns to the original calling routines ITEM (array of item names to use for legend), DODIC (array of DODICS for data base searches), and NITM (number of selected items).

(8) Subroutine GORCLI is called from PJSMRQ, PJSMWL, PJSMDL, and PJSMRQ. This routine asks the user if the ORACLE data base is to be used to retrieve the data to be charted. If so, the ORACLE name and password are entered and the login occurs. It then asks the user for the RCN that will be used to select the desired data. The menu appears as follows:

```
 00> DATA TO BE READ FROM DATA BASE? Y/N >
Y
 00> ENTER YOUR ORACLE USERNAME >

 00> ENTER YOUR ORACLE PASSWORD >

 00> ENTER REVISION CONTROL NUMBER (RCN) >
```

The actual login is accomplished by calling another routine DBLOGI in PJSMDB. All user inputs are verified for accuracy. Then by using the routine ZREADI in PJSMCOM, the entered responses are read and converted to integers.

(9) Subroutine GPDIPS is called from subroutines DOLPT2 in PJSMDB and WRKPT3 in PJSMWL, when the user requests a chart depicting PDIPs. It first displays the option on the screen as well as the possible selections as follows:

NO.	PDIP	NO.	PDIP
01	NEW MATERIAL FIELDING	10	ANNUAL TRAINING REQUIREMENTS
02	SUPPORT TEST REQUIREMENTS	11	DEPOT LEVEL 2
03	AIIQ	12	60 DAY MOB TRAINING
04	OPERATIONAL PROJECTS	13	30 DOS WRSA
05	MINIMUM ANNUAL TRAINING	14	90 DOA
06	DEPOT LEVEL 1	15	60 DOS WRSA
07	30 DOS	16	90 DAY MOB TRAINING
08	45 DOS	17	180 DOS
09	60 DOS		

```
> ENTER ALL DESIRED PDIPS
```

The user now has the ability to enter up to eight packages. If the user should enter too many, a message will be shown and the user is given the opportunity to reenter. Numbers are converted to integers by calling the routine ZREADI in PJSMCOM. The PDIPs information is then loaded into the appropriate data arrays. NPDIP (array of selected PDIP numbers), NP (number of PDIPs requested), and PDIPNM (PDIP name for legend).

(10) Subroutine GPLANT is implemented by DOLPT1, DOLPT2, and DOLPT3 in PJSMDL; WRKPT1 in PJSMWL; and PRDPT1 in PJSMPD. The purpose of this routine is to retrieve the selected plant and its name. GPLANT calls the routine DBGPLT, in PJSMDB, if data are to be read from the data base, or GETPLN if data are to be entered manually. Data values retrieved are converted to integers by using the routine ZREADI in PJSMCOM. The routine displays the menu and gives the user the opportunity to make the selection. The menu appears as follows:

```
NO.  PLANT NAME                NO.  PLANT NAME
 0 - TOTAL OF ALL PLANTS
 1 - IOWA                       2 - LONE STAR
 3 - MILAN                      4 - HOLSTON
 5 - COMMERCIAL                 6 - KANSAS
 7 - SCRANTON                   8 - LAKE CITY
 9 - LONGHORN                  10 - LOUISIANA
11 - PINE BLUFF                12 - INDIANA
13 - MISSISSIPPI               14 - RADFORD
15 - CRANE                     16 - HAWTHORNE
17 - SUNFLOWER                 18 - MCALESTER
19 - ALL PLANTS

21> ENTER PLANT OR OPTION NUMBER >
```

Once the selections are made the appropriate data are stored in the arrays PLTNO (plant code selected), PLTNM (plant name), NCH (length of plant name), and NPLT (number of plants selected). These arrays are passed back to the appropriate calling routine.

(11) Subroutine GPTSYL is implemented by the programs PJSMWL, PJSMPD, PJSMDL, PJSMRQ, and GUPDAT. Its purpose is to display the menu which describes the font style (style of the letters and numbers) of the charts so the user may select the one desired. The menu appears as follows:

```
NO.  FONT          / PLOT STYLE
 1 - STICK FONT    - QUICK DRAY
 2 - SCMPLX FONT   - PRESENTATION QUALITY, NO FILL
 3 - SWISS FONT    - PRESENTATION QUALITY, SOLID

00> SELECT MENU NUMBER >
2
```

The entered selection is converted to integer form by calling the routine ZREADI in PJSMCOM. The information is stored in the variable PLTYP (plot style number) and passed back to the calling routine.

(12) Subroutine GSERVC is implemented by the routine WRKPT4 in PJSMWL and REQPT1 in PJSMRQ. This routine provides a menu of the services available and allows the user to get the desired option. The menu is as follows:

```
NO.  SERVICE
 0 - ALL SERVICES
 1 - A<RMY>$
 2 - N<AVY SEA>$
 3 - N<AVY AIR>$
 4 - M<ARINE> C<ORPS>$
 5 - A<IR> F<ORCE>$
 6 - C<OAST> G<UARD>$
 7 - O<THER>$

24> ENTER MENU NUMBER >
```

The user entry is verified to be of an integer nature by the routine ZREADI in PJSMCOM. The selected information is stored in the arrays NSERV (array of selected service codes), NS (number of selected services), and SERVNM (name for use in legend). This information is passed back to the appropriate calling routine.

(13) Subroutine GSIGIT is implemented by the routines WRKPT1 in PJSMWL and PRDPT1 in PJSMPD. Purpose of the GSIGIT routine is to retrieve significant items from the data base. The user determines the selection criteria (or what is significant to the user). Options are the most dollars spent, the most work years expended, or the most quantity produced. The user also has the option of entering items from the screen. This can be useful if items are requested that would not fall into the first three options. This routine first displays the following menu which gives the user the selection criteria:

```
21 > ENTER PLANT OR OPTION NUMBER >

NO.  OPTION
 1 - USER ENTERED ITEMS
 2 - SIGNIFICANT ITEMS BY DOLLARS
 3 - SIGNIFICANT ITEMS BY QUANTITY
 4 - SIGNIFICANT ITEMS BY WORK YEARS
21> ENTER MENU NUMBER >
```

(a) If option 1 is selected, the routine GITEMS in PJSMCOM is implemented.

(b) If option 2 is selected, the routine DBSIGD in PJSMDB is implemented.

(c) If option 3 is selected, the routine DBSIGQ is called.

(d) If option 4 is selected, the routine DBSIGW is called.

The routine then displays the following menu which gives the user a choice of how the items will be identified on the charts in the legend:

```
NO.  OPTION
 1 - LEGEND BY DODIC
 2 - LEGEND BY SEQUENCE NUMBER
 3 - LEGEND BY SSN
 4 - LEGEND BY DODAC
 5 - LEGEND BY NSN
 6 - LEGEND BY ISN
 7 - LEGEND BY NOMENCLATURE (1-31)

21> ENTER MENU NUMBER >
1
```

All user responses are verified to be integer values by calling the routine ZREADI in PJSMCOM. Once the user has selected how the items selected will be identified, the routines DBINAM in PJSMDB, are called to select the appropriate identifier for the legend titles. All information is stored in the arrays DODIC (array of selected DODIC), ITEM (array of legend titles), and NITM (total number of items returned) and passed back to the calling routine.

(14) Subroutine GTMTYP is implemented by the routines PJSMDL, PJSMWL, PJSMPD, and PJSMRQ. Its function is to establish the terminal type the user is running on and intialize the DISSPLA post processor routine. The routine displays the following menu:

```
NO.  TERMINAL TYPE
 1 - TEKTRONIX 4006
 2 - TEKTRONIX 4010 COMPATIBLE
 3 - TEKTRONIX 4014
 4 - TEKTRONIX 4025
 5 - TEKTRONIX 4107
 6 - TEKTRONIX 4115
 7 - TEKTRONIX 4125
 8 - BLACK/WHITE META FILE
 9 - COLOR META FILE

00> ENTER TERMINAL MENU NUMBER >
```

The user identifies the terminal being used and the selection entered is verified for being an integer value by the routine ZREADI in PJSMCOM. If one of the options (1 through 4) are selected, the COLFG array is set to 0. This designates that a monochrome (black/white) terminal is being used and tells DISSPLA to call the subroutine PTEKAL, requesting the user to enter the exact terminal model number. If the user selects option 5 thru 7, the COLFG array is set to 1. This designates that a color capable terminal is being used and to call the DISSPLA subroutine PTK41 which asks the user to enter the exact terminal model number. An example of a color response follows:

```
 ENTER MODEL NUMBER
4107
```

This information is stored in TERM (terminal type) and COLFG (color/black and white flag) and passed back to the calling routine.

(15) Subroutine GUPDAT is a large subroutine which stores and allows the users to update the graphics variables. Following is a list of the graphics variables and a brief description of each:

GTITLE - TITLE AT TOP OF PLOT OR GRAPH. THREE LINES MAX.
GXTTLE - X-AXIS TITLE LINE
GYTTL1 - Y-AXIS TITLE LINE LEFT AXIS
GYTTL2 - Y-AXIS TITLE LINE RIGHT AXIS
GLGTTL - LEGEND TITLE LINES 9 MAX
GXATTL - X-AXIS TICK MARK LABELS
GXSCLE - X-AXIS SCALE MIN,STEP,MAX
GYSCL1 - Y-AXIS SCALE MIN,STEP,MAX LEFT AXIS
GYSCL2 - Y-AXIS SCALE MIN,STEP,MAX RIGHT AXIS
GCOLOR - FILL COLORS FOR EACH DATA ELEMENT 9 MAX
GFILL - FILL PATTERNS FOR EACH DATA ELEMENT 9 MAX
GBGCOL - BACKGROUND COLOR
GPLTYP - SELECTED PLOT TYPE
GLNTYP - LINE TYPES FOR EACH DATA ELEMENT 9 MAX
GCHRSZ - CHARACTOR SIZE FOR PLOT LABELS
GLNTHK - LINE THICKNESS
GCOLFG - COLOR TERMINAL FLAG
GSTYLE - CHARACTOR TYPE OR STYLE
GIOFLG - I/O FLAG DATE SOURCE DATA BASE OR USER INPUT
GNUM1 - NUMBER OF BARS OR CURVES LEFT AXIS
GNUM2 - NUMBER OF BARS OR CURVES RIGHT AXIS
GXNUM - NUMBER OF X VALUES
GXPAGE - X PAGE SIZE
GYPAGE - Y PAGE SIZE

GXORG - X ORIGIN POSITION
GYORG - Y ORIGIN POSITION
GXLEN - X-AXIS LENGTH
GYLEN - Y-AXIS LENGTH
GDATA - DATA STORAGE ARRAY

This routine is called from BGNPLT in PJSMCOM. Once it is implemented, the routine displays the following menu:

NO.	UPDATE OPTION	NO.	UPDATE OPTION
0	- EXIT	1	- PLOT TITLE
2	- X-AXIS TITLE	3	- Y-AXIS TITLE
4	- X-AXIS SCALE	5	- Y-AXIS SCALE
6	- LEGEND TITLES	7	- FILL COLORS
8	- FILL PATTERNS	9	- LINE TYPES
10	- CHARACTOR SIZE	11	- LINE THICKNESS
12	- COLOR FLAG	13	- DASTA I/O FLAG
14	- PAGE SIZE	15	- PLOT SIZE
16	- PLOT ORIGIN	17	- PLOT TYPE
18	- PLOT STYLE	19	- TICK TITLES

26› ENTER ALL MENU NUMBERS DESIRED ›

If desired, the user may enter several options or one option at a time. For any option entered, the routine will display its current value for that parameter. Following, it will ask for the new values. To alter a few of these settings will require the use of the Computer Associates DISSPLA Manual or Pocket Guide. Settings such as 'FILL PATTERNS', 'FILL COLORS', and 'LINE TYPES' would require the user to refer to the DISSPLA Manual. Most of the changes are verified for being integer values by the routine ZREADI in PJSMCOM. Once all of the graphics parameters are as desired, they are loaded back into the variable listed above and passed back to the calling routine via a COMMON BLOCK. This common block is loaded in PJSMGC and inserted into programs if required.

(16) Subroutine GYEAR is called from the graphics module. It allows the user to enter the first year for the data to be plotted as well as the number of years desired. The program prompts the user as follows:

24› ENTER FIRST YEAR FOR PLOT ›

24› ENTER TOTAL NUMBER OF YEARS TO PLOT ›

Once the user has entered the prompted data and the responses are verified by ZREADI in PJSMCOM, for being integers, the data are loaded into the variables IYR (first year for plotting) and NYR (number of years or plot) and passed back to the calling routine.

(17) Subroutine LSTDAT is called from the routine BGNPLT in PJSMCOM. It allows the user to review the current values store in the graphics common block. Once the user selects this option the following will appear on the screen:

```
GRAPHICS PARAMETERS                                          LEGEND
  PAGE SIZE = 11.0 X  8.5  PLOT SIZE =  8.5 X  6.2   1 GOGO <PLANTS>$
  CHAR SIZE =  0.180          ORIGIN =  1.5 ,  1.0   2 GOCO <PLANTS>$
  THICKNESS =  0.020      COLOR FLAG =  TRUE         3 COCO <PLANTS>$
  CHAR FONT = SCMPLX        I/O FLAG =  TRUE         4
  PLOT TYPE =  3 , GROUPED BAR CHART & TOTALS LIN    5
   PATTERNS =  1,  5,  7, 16, 19, 4,  3, 12          6
FILL COLORS = 12,  9,  7, 16,  2, 6, 18,  4          7
LINE STYLES = 18, 15, 17, 16,  2, 3,  4,  7          8
 TITLE  = TOTAL DOLLAR EXPENDITURE                   9
          GROUPED BY PLANT TYPE              X - 1990.0, 1.0, 1991.0
                                             Y1 -    0.0, 0.0,    0.0
     X-AXIS TITLE = F<ISCAL YEAR>$           Y2 -    0.0, 0.0,    0.0
     Y-AXIS TITLE = M<ILLIONS OF DOLLARS>r
TICK TITLES = FY90     FY91

ENTER 'C' TO CONTINUE
```

The user must keep in mind that several of these values are set by a default. For example: 'page size', 'plot size', 'patterns', 'origin', etc. However, the user has the option to change any of these values. This routine returns nothing to the calling routine.

(18) Subroutine MYPIE is required by DISSPLA to place color in the legend of a color pie chart. The user should not be concerned with this routine because it is for internal use only.

(19) Subroutine MYSPEC is required by DISSPLA to place color in the legend of a color line or bar chart. The user should not be concerned with this routine because it is for internal use only.

(20) Subroutine NOTDON is called from any routine and its function is to display a message if the selected plot is not available. The message appears as follows:

PLOT SELECTED IS NOT YET AVAILABLE.

(21) Subroutine PLOTIT is called from the routine BGNPLT in PJSMCOM. This routine begins the set up processing of the chart. It calls a few DISSPLA routines to set up the page size, the physical origin, the font styles and character size in height. The routine also checks to see what type chart has been selected and calls the appropriate routine in PJSMCHT.

(22) Subroutine SETCOL is called internally to set the hue, intensity, and saturation for the colors selected. This routine makes a call to the DISSPLA routine HWHSHI which sets the aforementioned color parameters. The user should not be concerned with this routine because it is for internal use only.

(23) Subroutine ZAESTH is called from the display routines to determine the aesthetic minimum, maximum, major and minor values for the charts. The calculated values are returned in the variables XMN (aesthetic minimum), XMX (aesthetic maximum), DXMJOR (major internal), and DXMNOR (minor internal) and are passed back to the calling routine.

(24) Subroutine ZREADI is called from many of the other routines and its function is to read integer values from an input string in a free form manner. The user should not be concerned with this routine because it is for internal use only.

e. Output - Creates no output in report form.

f. Security - All information generated is unclassified.

5.64.2 Conventions. The program is written in structured F77. The program does not create any output in report form, but produces a specific module or data element needed for the calling routine. The program is a module of the whole and independently serves no purpose.

5.64.3 Verification Procedures. The verification of this program is quite difficult. Any errors occurring would have to be traced through the entire procedure. It would be doubtful that the error would be literally within this module.

5.64.4 Error Conditions. There are no error messages printed to an output file or to the screen. However, if an error does occur, it will be reflected in the plotted chart.

5.64.5 Listings. The program listing contains many comment statements to assist in making program changes. The program listing PJSMCOM is located in <CAS2SA>SAXTST>#JSM>PJSM.

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.	1. DATE 30 Nov 87
2. PROGRAM ID 5.64	3. PROGRAM NAME PJSMCOM
4. REV NO./DATE	5. DESCRIPTION OF REVISION

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

5.65 Program. Post Post Processor for the Plant Job Scheduling Model (POST-POST)

5.65.1 Program Description. The Post-Post program is written in structured FORTRAN 77. It should be run immediately following the Post Processor for the Plant Job Scheduling (JSMPP) to produce a summary report.

a. Identification - The source code for the Post-Post program is in a file named POST-POST.F77. This file is compiled and linked to produce an executable run file -- POST-POST.RUN -- as follows:

```
F77 POST-POST -SI 2 -INTS -LIST -XREF -RANGE
BIND -LO POST-POST -LI SYNCLIB -LI
```

b. Functions - The Post-Post program reads the JSMPP.COMO file, produced by the Post Processor for the Plant Job Scheduling Model, and writes a summary report. This report is sorted by DODIC and FY and shows each change that was entered into the PJSM RESULT4 table, the change that was allowed, and any reason(s) for not allowing a full change to take place. Also, there is a reference line number that is the line number in the JSMPP.COMO file where the processing of the change begins.

c. Input - The JSMPP.COMO file.

d. Processing - The JSMPP.COMO file contains a line beginning with '.S ' for each change. This line contains the DODIC, FY, and the change that was entered. The change that was allowed appears on a similar line except it begins with '.F '. Between each of these could be one or more remarks lines which begin with '.R '. These lines are stripped out and processed in the following steps:

(1) SyncSort parameters are defined and the application is initiated by calling subroutine SS#SET.

(2) The JSMPP.COMO file is opened, and the first line is read.

(3) The following steps are performed until the end of file is reached.

(a) The input line number is incremented by one.

(b) If the line is a '.S ' line, it is combined with the line number and the resulting record is released to SyncSort.

(c) If the line is a '.R ' line, it is combined with the DODIC and FY from the last '.S ' line and the resulting record is released to SyncSort.

(d) If the line is a '.F ' line, it is released to SyncSort.

(e) The next line is read.

(4) The released records are sorted by calling subroutine SS$CMB.

(5) The sorted records are then returned from SyncSort one at a time and are processed in the following steps:

(a) The information for each set of records is accumulated in buffers until the '.F ' record is received. All the remarks from the '.R ' records are edited and strung together to form a single remarks line.

(b) Depending upon the length of the remarks line, one or more lines are printed in the output file for each set of records. Remarks are not printed if the full change was allowed.

(c) For improved readability, a blank line is inserted to separate DODICs.

(d) Form feeds and headers are printed out as necessary to prevent printing over page breaks.

(6) The program signals SyncSort that it is finished by calling subroutine SS$CLN.

e. Output - Post Processor summary report (POST-POST.COMO file).

f. Security - The Post-Post program processes information that is unclassified.

g. Interfaces - The program interfaces with the SyncSort/ PRIME system.

5.65.2 Conventions. The program was written in structured FORTRAN 77.

5.65.3 Verification Procedures. None needed.

5.65.4 Error Conditions. Not applicable.

5.65.5 Listings. Program listings are located in <SYSSA>SAXPGM>*PS

PROGRAM REVISION For use of this form, see TB 18-111; the proponent agency is DCSOPS.		1. DATE 30 Nov 87
2. PROGRAM ID 5.65	3. PROGRAM NAME POST-POST	
4. REV NO./DATE	5. DESCRIPTION OF REVISION	

DA FORM 4752-R, APR 83 REPLACES DA FROM 5752, NOV 78, WHICH IS OBSOLETE.

ANNEX A

DOCUMENTED PROGRAMS

UTILITY PROGRAMS

JSM1.CPL	PJSM Master Menu Program
ARCHIVE.CPL	PJSM Table Export Program
BIND3IX.CPL	ORACLE Precompile, Compile, Link, and Execute Program
CREV.FOR	Change Revision Program
PJSM_MAINT.CPL	PJSM Data Base Maintenance Program
BASECHK.FOR	Data Base Consistency Check
VALID.CPL	ORACLE User Validation Program
BATCH CPL SUBPGMS	Batch CPL Subprograms Interfacing JSM1.CPL

INPUT SCREENS

JSM_1.INP	Item Descriptors Input Screen
JSM_2.INP	Item Planning Parameters Input Screen
JSM_3.INP	Item Production Capacity/Staffing Input Screen
JSM_4.INP	General Plant Data Input Screen
JSM_5.INP	Primary Component Information Input Screen
JSM_6.INP	Yearly Consumption Requirements Input Screen
JSM_7.INP	Production Project Specification Input Screen
JSM_8.INP	Army Requirements Stratification Input Screen
JSM_9.INP	Other Customer Requirements-Order Input Screen
JSM_10.INP	Secondary Component Information Input Screen
JSM_11.INP	Line Descriptors Input Screen
JSM_12.INP	Update or Display Army Requirements Input Screen
JSM_13.INP	Update or Display Other Service Requirements Input Screen
JSM_14.INP	DODIC/Project Cross Reference Input Screen
JSM_15.INP	RAMP Item Input Screen
JSM_16.INP	RAMP Production Input Screen
GUID_1.INP	JSM Simulation Guidelines Input Screen

INPUT PROGRAMS

PKG.FOR	RDAISA (Army Requirements) Input Program
ICAPP.FOR	ICAPP Unit Prices (Including Army's) and Other Customer Requirement Input Program
APS.FOR	Alternate Priority Scheme Generator Program

INPUT DATA REPORT GENERATORS

IIDR.FOR	Individual Item Data Report
PIDR.FOR	Plant Item Data Report
AEMS.FOR	Data Extract for Ammunition Executive Management System (AEMS)
IWIR.FOR	Increased Workload Impact Report

AMMUNITION SCHEDULING MODELS

PS.FOR	Ammunition Scheduling Model
JSMPP	Plant Job Scheduling Model Post Processor
BUILD_PTR.FOR	Build Pointer Arrays Program (for PS and PPJSM)
TREQ.FOR	Requirement Summarization Program
POST-POST	Post Post Processor for the Plant Job Scheduling Model

OUTPUT REPORT GENERATORS

RPT1.FOR	Army Acquisition Report
RPT2.FOR	Workload Summary Report by Item (by Plant and Line)
RPT3.FOR	Plant Workload Utilization Detailed Report
RPT4.FOR	Plant Workload Summary Report
RPT5.FOR	Analysis Worksheet Report
RPT6.FOR	Program Development Incremental Package(PDIP) Summary Report
RPT7.FOR	Compare Program
RPT8.FOR	Summary of Training and Test Items not Achieving 100 Percent
RPT9.FOR	Readiness Report
RPT10.FOR	Other Service Shortfall Report
RPT11.FOR	PDIP Package Structure by Item
RPT12.FOR	PDIP Cost Report
RPT13.FOR	Work-years Available to Increase Plant Utilization Report
RPT14.FOR	Secondary Item Status Report
RPT15.FOR	Summary of Package Requirement/Filled Report
RPT16.FOR	Conventional Ammunition Acquisition Plan

OUTPUT SCREENS

RESULT1.INP	Army Requirement Allocation Output Screen
RESULT2.INP	Other Customer Requirement Allocation Output Screen
RESULT3.INP	Production Summary Output Screen
RESULT4.INP	Output Screen
RSLT_CHG.INP	Post Processor Change Screen

ANNEX B

DATA BASE TABLES

TABLE NAME	DESCRIPTION
REVTAB	Revision Table
ITEM	Cross Reference Table
SERVICE	Service Table
PACKAGE	Package Table
FAMILY	Family Table
PLANT	Plant Table
STAFF	Staff Table
LINE	Line Table
GOALS	Goals Table
PFSTAB	PDIP Funding Sequence Table
ICT	Item Component Table
PRODT	Production Table
PROJECT	Project Table
PRODT_FY	Production Change Table
ITEM_DATA	Item Data Table
RAMP_ITEM	RAMP Years Item Table
RAMP_PROD	RAMP Years Production Table
REQTS_ARMY	Army Requirements Table
REQTS_OTHER	Other Requirements Table
TREQ	Total Requirements Table
REASON_CODE	Reason Code Table
RESULT1	Army Requirements Output Table
RESULT2	Other Requirements Output Table
RESULT3	Production Output Table
RESULT4	Item Output Table
RESULT5	Summary by Package and Family
RESULT6	Summary by Plant and Line

REVTAB (Revision Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
STATUS	Char	1	0	Status of current revision. NOTE: Only baseline plus one other record can contain a 'Y'. Y = Yes N = No
BFY	Number	2	0	Beginning FY
NFY	Number	2	0	Number of FYs
LVTRNG	Number	1	0	Level of Training 1 = Level 1 Training 2 = Level 2 Training
DEPOTR	Number	1	0	Depot requirements needed to fill 1 = Provide during the first year (200 DOS) 3 = Provide over a 3 year (90 DOS, 55 DOS, and 55 DOS)
ASSETS	Char	1		Asset indicator Y = Use initial assets N = Do not use initial assets
SHARING	Char	1		Sharing indicator Y = Use excess Army inventory to fill Other Customer requirements N = Do not share
CREATED	Date	14		Date this study run was created
CUSER	Char	12		User who created this study run
ARCHIVED	Date	14		Date this study was pruged from the DBMS
AUSER	Char	12		User who archived this study
REMARKS	Char	40		General comments on this study

ITEM

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
FSC	Char	4		Federal Supply Class
DODIC	Char	4		Department of Defense Identification Code
SSN	Char	6		Standard Study Number. Numbers preceeded by the symbol '*' identify Other Service/FMS unique ammunition items
SEQ	Number	3	0	Sequence Number to indicate ordering on output reports
NSN	Char	16		National Stock Number
ISN	Char	5		Item Sequence Number
FAMILY	Number	1	0	Family Code
SUB_FAMILY	Number	1	0	Sub Family Code
USE	Char	1		Use Code W = War Reserve T = Training X = Both N = Not assigned
NMF	Char	1		New materiel fielding code Y = Yes N = No
NOMENCLATURE	Char	48		Description of the item
UOM	Char	4		Unit of Measure Thou = Thousands Each = Units

SERVICE

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
Service	Char	2		Service Code
Name	Char	15		Service Name

AR = Army
FM = Foreign Military Sales/Grant Aid
AM = Army Miscellaneous (includes DEMIL)
NS = Navy Sea
NA = Navy Air
MC = Marine Corps
AF = Air Force
CG = Coast Guard
OT = Other

PACKAGE

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
PKGL	Number	2	0	Package Level
NAME	Char	30		Package Name

Package Level (for Army use only)*
1 = New Materiel Fielding (NMF) = summation of PKGl 3 thru 7
2 = Test
3 = AIIQ
4 = OPS
5 = Level 1 Training
6 = Depot 1
7 = War Reserve 1.0
8 = War Reserve 2.0
9 = War Reserve 3.0
10 = Level 2 Training
11 = Depot 2
12 = MOB A
13 = WRSA 1.0
14 = War Reserve 4.0
15 = WRSA 2.0
16 = MOB B
17 = War Reserve 5.0

*NOTE: Package level 0 is used in RESULT1 table for undelivered prior year quantities which are obtained from the RAMP_PROD table.

FAMILY

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
FAMILY	Number	1	0	Family Number
SUB_FAMILY	Number	1	0	Sub Family Number
NAME	Char	40		Name of Family

FAMILY	SUB-FAMILY	DESCRIPTION
1	0	Small arms thru 30mm
2	0	Light cal thru 120mm (motor/tank)
3	0	Heavy cal 155mm thru 8"
4	0	Rockets
5	0	Grenades/mines/demo
6	0	Miscellaneous (smoke/illum)
7	0	Non-AAO
8	0	Miscellaneous Hardware
9	0	Components

PLANT

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
Plant	Char	2		Plant Code
Name	Char	15		Plant Name
Type	Char	4		Plant Type
Prod_oh_fac	Number	5	4	Production overhead factor
Non_prod_oh_fac	Number	5	4	Non-production overhead factor

Plant type:

GOGO = Government Owned, Government Operated

GOCO = Government Owned, Contractor Operated

COCO = Contractor Owned, Contractor Operated

IA	GOCO
LS	
LC	
LH	
HS	
LA	
MS	
SC	
MI	
RA	
KS	
IN	
PB	GOGO
CN	
MC	
CO	COCO

STAFF

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
Plant	Char	2		Plant Code
FY	Number	2	0	Fiscal Year
RCN	Number	2	0	Revision Control Number
GOAL	Number	3	0	Percent of direct labor work-force which should be achieved to balance workload
Direct	Number	5	0	Number of direct labor employees
OMA_OH	Number	5	0	Number of OMA overhead employees

LINE

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
Line	Number	3	0	Line Number
Plant	Char	2		Plant Code
Desc	Char	40		Line Description

GOALS*

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
FY	Number	2	0	Fiscal Year
RCN	Number	2	0	Revision Control Number
DRAWDN	Number	3	0	Drawdown in days of supply (DOS)
BUILDUP	Number	3	0	Buildup (target) in DOS
PCT_TRNG	Number	3	0	Percent of training requirements to be filled (usually = 100%) (only applied to Level 1)
ACTIVITY_I	Number	7	1	Total obligation authority for hardware (in millions)
ACTIVITY_II	Number	7	1	Total obligation authority for projects (in millions)

*The drawdown, buildup, and PCT_TRNG values in this table will be applied to all items with requirements unless specific values are entered for an item in the DRAWDN and BUILDUP columns of the ITEM_DATA table.

PFSTAB
(PDIP Funding Sequence Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
PKGL	Number	2	0	Package Level
RCN	Number	2	0	Revision Control Number
SEQ	Number	2	0	Sequence in which the PKGLs are funded

ICT
(Item Component Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
COMP	Char	4		Component DODIC
TYPE	Char	1		Type of Component P = Primary component, such as explosives, black powder, combustible cases, metal parts, and grenades. (These are possible pacing components.) S = Secondary component such as primers, prop charges, and fuzes.
PF	Number	12	6	Procurement Factor. For secondary components this is actually the training procurement factor.
WRPF	Number	12	6	War Reserve Procurement Factor. This is used only for secondary components.

PRODT
(Production Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
PLANT	Char	2		Plant Code
LINE	Number	3	0	Line Number
R185	Number	9	0	Production rate for a 1-8-5 operation (1 shift, 8 hours a day, 5 days a week)
R285	Number	9	0	Production rate for a 2-8-5 operation (2 shifts, 8 hours a day, 5 days a week)
S185	Number	4	0	Staffing needed to support a 1-8-5 production rate
S285	Number	4	0	Staffing needed to support a 2-8-5 production rate
AVAIL	Number	1	0	Availability 0 = Line not available 1 = 1st shift only 2 = 1st and 2d shifts available
MPQ	Number	7	0	Minimum Procurement Quantity
MPA	Number	10	0	Maximum Production Allowed

PROJECT

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
PROJECT	Char	8		Project Code
PLAN_YR	Number	2	0	Planning Year
COST	Number	7	1	Cost of Project (in millions)
TITLE	Char	30		Title of Project

PRODT FY

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
PROJECT	Char	8		Project Code
DODIC	Char	4		Department of Defense Identification Code
PLANT	Char	2		Plant Code
LINE	Number	3	0	Line Number
FY	Number	2	0	Fiscal Year
MO	Number	2	0	Month
RCN	Number	2	0	Revision Control Number
R185	Number	9	0	Production rate for 1-8-5
R285	Number	9	0	Production rate for 2-8-5
S185	Number	4	0	Staffing for 1-8-5
S285	Number	4	0	Staffing for 2-8-5
AVAIL	Number	1	0	Availability 0 = Not available 1 = 1st shift only 2 = 1st and 2d shifts available
MPQ	Number	7	0	Minimum procurement quantity
MPA	Number	10	0	Maximum production allowed

ITEM DATA

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Number	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
RCN	Number	2	0	Revision Control Number
UNIT_PRICE	Number	12	0	Army's Unit Price
PRI	Number	3	0	Priority - As assigned by the Deputy Chief of Staff for Operations (DCSOPS). The lower the number, the higher the priority. Priority number 777 denotes component items for the end items, commercially procured items, and items for which DCSOPS did not assign a priority
ALT_PRI	Number	3	0	Alternate Priority Scheme
DRAWDN	Number	3	0	Drawdown in DOS a. When blank, the value obtained from the GOALS table is applied b. When positive, this number is applied
BUILDUP	Number	3	0	Buildup in DOS a. When blank, the value obtained from the GOALS table is applied b. When positive, this number is applied
PCT_TRNG	Number	3	0	Percent of Training requests to be filled
POM_COST	Number	6	1	Projected production cost, in millions of dollars, for classified and non-AAO items. These costs are included to reflect the total budget costs where quantities are not available and/or considered classified
MIA	Number	10	0	Maximum inventory allowed

RAMP ITEM

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Program year of the Funded Delivery Period (FDP)
BEG_CY	Number	2	0	Beginning CY of the FDP
BEG_MO	Number	2	0	Beginning calendar month of the BEG_CY
END_CY	Number	2	0	Ending CY of the FDP
END_MO	Number	2	0	Ending calendar month of the END_CY
NO_MOS	Number	2	0	Number of months in the FDP
BEG_ASSETS	Number	10	0	Beginning assets
PROD_ARMY	Number	10	0	Total production for Army
PROD_TOT	Number	10	0	Total production
LOSSES	Number	10	0	Test/Training/Other losses
OC_DEL	Number	10	0	Other customer deliveries
END_ASSETS	Number	10	0	Ending assets at end of FDP
ARMY_BUY_QTY	Number	10	0	Army buy quantity
UNIT_PRICE	Number	12	4	Army unit price
DV_ARMY	Number	10	0	Dollar value of the funded Army production quantity

RAMP PROD

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
PLANT	Char	2		Plant Code
LINE	Number	3	0	Line Number
UNDEL_AR	Number	10	0	Undelivered RAMP year quantity for Army
UNDEL_TOT	Number	10	0	Undelivered RAMP year quantity for other customers
PROD_ARMY	Number	10	0	Army and production for Army
DV_ARMY	Number	10	0	Dollar value of the funded Army production quantity
WRKYRS_AR	Number	7	2	WRKYRS of the Army production quantity
PROD_TOT	Number	10	0	Total production for Army and other customers
DV_TOTAL	Number	10	0	Dollar value of the Army and other customer production
WRKYRS_TOT	Number	10	2	WRKYRS of the Army and other customer production
PROD_YR_QTY	Number	10	0	Production year quantity (program year + 1)

REQTS ARMY

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
PKGL	Number	2	0	Package Level
Quantity	Number	10	0	Required Quantity
Change	Number	10	0	Indicated the change from the last set of requirements

REQTS OTHER

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
Service	Char	2		Service Code
UNIT_PRICE	Number	12	4	Unit Price
Quantity	Number	10	0	Required Quantity
Change	Number	10	0	Indicated the change from the last set of requirements

TREQ
(Total Requirements)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
OTHER_REQ	Number	10	0	Total other service requirements: (NS + NA + MC + AF + CG + OT + FM + AM)
ARMY_TATR	Number	10	0	Total Army test and training requirements
DRAWNDN_QTY	Number	10	0	Drawdown quantity
BUILDUP_QTY	Number	10	0	Build up quantity
NMF_QTY	Number	10	0	New Materiel Fielding (NMF) quantity. Equals the sum of PKGL 3 thru 7 for NMF items.
TOT_AREQ	Number	10	0	Total Army requirement

REASON CODE

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RC	Char	1		Reason Code
NAME	Char	40		Reason for shortfall

CODE	DESCRIPTION
A	Lack of sufficient funds
B	Production capacity constraint
C	Staffing constraint
D	Cannot produce a primary component
E	Requirement exceeds buildup target
F	Level 2 training not allowed
G	Pipeline filled over 3-year period
H	No production facilities available
I	Requirement exceeds percent of training allowed
J	Maximum inventory allowed constraint
K	Maximum inventory allowed constraint

RESULT1
(Army Requirements Output Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
PKGL	Number	2	0	Package Level
TYPE	Char	1		Type of item E = End item P = Primary comp S = Secondary comp U = Prior year undelivered production ('U' is used in conjunction with PKGL = 0)
PACER	Char	1		For primary components, this indicated whether the comp was a pacing item: Y = Yes N = No
INV_APP	Number	10	0	Inventory applied
INV_DV	Number	10	0	Dollar value of inventory applied
PROD_APP	Number	10	0	Production applied
PROD_DV	Number	10	0	Dollar value of production applied. (This is 0 when PKGL = 0 and TYPE = 'U')
WRKYRS	Number	7	2	Work-years used
PCT_LINE	Number	7	2	Percent of line used to produce the quantity, based on a 1-8-5 shift
SFALL_QTY	Number	10	0	Shortfall (unfilled requirement)
RC	Char	1		Reason Code for shortfall

RESULT2
(Other Requirements Output Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
SERVICE	Char	2		Service
TYPE	Char	1		For primary component, this indicates whether the comp was a pacing item: Y = Yes N = No
INV_APP	Number	10	0	Inventory applied
INV_DV	Number	10	0	Dollar value of inventory applied
PROD_APP	Number	10	0	Production applied
PROD_DV	Number	10	0	Dollar value of production applied
WRKYRS	Number	7	2	Work-years used
PCT_LINE	Number	7	2	Percent of line used to produce the quantity, based on a 1-8-5 shift
SFALL_QTY	Number	10	0	Shortfall (unfilled requirements)
RC	Char	1		Reason Code for shortfall

RESULT3
(Production Output Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
DODIC	Char	4		Department of Defense Identification Code
PLANT	Char	2	0	Plant Code
LINE	Number	3	0	Line Number
FY	Number	2	0	Fiscal Year
TYPE	Char	1		Type of item
PROD_OTHER	Number	10	0	Amount produced to support other service requirements
PROD_TATR	Number	10	0	Amount produced to support Army test and training
PROD_ADOS	Number	10	0	Amount produced to support Army DOS (includes prior years undelivered production)
DV_ARMY	Number	10	0	Dollar Value of Army production (does not include cost of prior year undelivered production)
WRKYRS_AR	Number	7	2	Work-years used to produce the Army quantity
PCT_LINE_AR	Number	7	2	Percent of line used to produce the Army quantity, based on a 1-8-5 shift
DV_OTHER	Number	10	0	Dollar value of other customer production
WRKYRS_OT	Number	7	2	Work-years used to produce other customer quantity
PCT_LINE_OT	Number	7	2	Percent of line used to produce other customer quantity, based on a 1-8-5 shift

RESULT4
(Item Output Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
DODIC	Char	4		Department of Defense Identification Code
FY	Number	2	0	Fiscal Year
TYPE	Char	1	0	Type of item also includes 'X' for unassigned items with remaining assets
AINV_OTHER	Number	10	0	Army inventory applied to support other service requirements
AINV_TATR	Number	10	0	Army inventory applied to support Army test and training requirements
AINV_ADOS	Number	10	0	Army inventory applied to support Army DOS requirements
E_ASSETS	Number	10	0	Ending Assets
PFILL_OTHER	Number	7	2	Percent of other service requirements filled (by inventory + production)
PFILL_TATR	Number	7	2	Percent of Army test and training requirements filled (by inventory + production)
PFILL_ADOS	Number	7	2	Percent of Army DOS requirements filled (by inventory + production)
SFALL_OTHER	Number	10	0	Shortfall of other service requirements
SFALL_TATR	Number	10	0	Shortfall of Army test and training requirements
SFALL_ADOS	Number	10	0	Shortfall of Army DOS requirements
PROD_ARMY	Number	10	0	Total produced for Army (includes prior year undelivered production)

RESULT4
(Item Output Table - Continued)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
DV_ARMY	Number	10	0	Dollar value of Army production (does not include cost of prior year undelivered production)
PROD_CHG	Number	10	0	Manual change (+ or -) to PROD_ARMY
DV_CHG	Number	10	0	Manual change (+ or -) to DV_ARMY
CUM_CHG	Number	10	0	Cumulative manual change (+ or -) to PROD_ARMY

RESULTS
(Summary by Plant and Package/Service Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
FY	Number	2	0	Fiscal Year
PLANT	Char	2		Plant Code
PKGL	Number	2	0	Package Level
SERVICE	Char	2		Service Code
DV_PROD	Number	10	0	Dollar Value of Production
WRKYRS	Number	7	2	Work-years used

RESULT6
(Summary by Plant and Line Table)

ELEMENT	TYPE	WIDTH	SCALE	DESCRIPTION
RCN	Number	2	0	Revision Control Number
FY	Number	2	0	Fiscal Year
PLANT	Char	2	0	Plant Code
LINE	Number	3	0	Line Number
DV_PROD	Number	10	0	Dollar Value of Production
WRKYRS	Number	7	2	Work-years used
PCT_LINE	Number	7	2	Percent of line used based on a 1-8-5 shift

ANNEX C

DECISION FLOW PROCESS FOR PRODUCTION SCHEDULING MODEL

1. Obtain the Revision Control Number (RCN).

2. Obtain guidance from the Revision Table (REVTAB) - applied to all items.

    a. Beginning Fiscal Year (FY).

    b. Number of FYs for the study.

    c. Level of Training Requirements.

    d. Depot Training Requirements.

    e. Are assets used or is everything produced.

    f. Are excess assets applied to satisfy Other Service Requirements.

3. Clear results out of tables for this RCN.

4. Get initial data for all items.

    a. From PTR_ARRAY file (generated from the BUILD_PTR program):

        (1) Obtain all items that will be considered.

        (2) Obtain primary components and procurement factors.

        (3) Obtain secondary components and procurement factors.

        (4) Obtain list of plants in the data base.

        (5) Obtain all locations where an item can be produced.

        (6) Obtain list of end items that use each primary component.

    b. From LINE table (LINE): Get valid line/plant combinations.

    c. From RAMP YEARS table (RAMP_ITEM): Get beginning assets.

d. From PRODUCTION table (PRODT): Get production data (R185, R285, S185, S285, number of shifts (available, MPA) for each DODIC/ LINE combination.

e. From SERVICE table: Obtain list of valid service codes.

5. Develop yearly production schedule.

a. Update yearly data.

(1) From ITEM DATA table (ITEM_DATA):

(a) Get unit prices for Army items.

(b) Get maximum inventory allowed (MIA) constraints for each item.

(2) From STAFF table (STAFF):

(a) Get direct labor (actual available) for each plant.

(b) Get goal (percent of the direct labor workforce, for each plant, which should be achieved to balance workload - usually 90 percent).

(3) From PRODUCTION CHANGE table (PRODT_FY):

Update production data (add or delete lines, change production rates, staffing, shift availability, MPA).

(4) From TOTAL REQUIREMENTS table (TREQ).

(a) Get drawdown quantity for each item.

(b) Get buildup quantity for each item.

(5) From RAMP YEARS PRODUCTION table (RAMP_PROD):

(a) Schedule the undelivered RAMP year quantities and their primary components.

(b) Costs are not charged for the actual year of production because the item was previously funded.

(c) Undelivered quantities that were not produced in the first FY are carried forward to the remaining years.

(d) Scheduled production is written to the RESULT1 table.

b. Process Other Customer Requirements. (This routine processes the Other Customer Requirements in DODIC order, writes results to the RESULT2 table, and accumulates data for other output tables.)

(1) From OTHER REQUIREMENTS table (REQTS_OTHER):

(a) Get requirements and unit prices for each item, for each service.

(b) Starts with the first item listed in the table.

(2) From the SERVICE table (SERVICE): Check for valid service codes against the codes listed in the SERVICE table.

(3) Check if other customers are allowed to use Army inventory:

(a) If 'Yes', check asset posture versus drawdown quantity. Apply excess assets toward satisfying Other Service requirements. If Other Service requirements are now satisfied, go to the next item (return to step 5b(1) and process the next item).

(b) If 'No', check if the item can be produced; i.e., could have a production capacity constraint, staffing constraint, cannot produce a primary component, or no production facilities available. If a constraint exists, go to the next item (return to step 5b(1) and process the next item).

(4) For each plant/line combination where this item can be produced, the following checks will be performed:

(a) Check remaining line capacity.

(b) Check remaining staffing level.

(c) If an Army requirement is being processed, then check if there is an MPA constraint on this item. If there is, production will be limited.

(d) Compute and schedule the quantity permitted by the remaining production capacity and staffing.

(e) If requirements were not satisfied, check for other production facilities (return to step 5b(4)(a) and continue processing).

(5) Schedule primary components to support the end item. (This routine controls the processing of primary components, given that an end item is produced.):

(a) Select the first primary component and its procurement factor.

(b) Compute the primary component's requirements.

(c) Check if the requirements for this component can be satisfied from inventory, and apply what is possible. If the inventory can satisfy the total requirements for this component, select the next primary component (return to step 5b(5)).

(d) If the inventory cannot satisfy the requirements for the component, then schedule production of this component (return to step 5b(4)). If production satisfied the requirements, then check the components for the primary component, and then select the next primary component (return to step 5b(5)).

(e) If the requirements for the primary component cannot be satisfied by inventory or production, then the component is identified as a pacer. Other end items will thus be constrained if they require this primary component.

(6) Go process next Other Customer requirement. (Repeat loop 5b until all Other Customer requirements are satisfied.)

c. Process Army Data. (This routine processes the Army requirements in a specified priority order within each PDIP, writes results to RESULT1 table, and accumulates data for other output tables.)

(1) Get Army related yearly data. (This routine gets the TOA Activity I dollars, the POM costs for specified items, the PDIP funding sequence, and the percent of training requirements to be filled for each item.)

(a) From the GOALS table, obtain the Total Obligation Authority for hardware dollars (TOA Activity I), and the Percent of Training Requirements to be Filled (PCT_TRNG), for all items.

(b) From the ITEM DATA table (ITEM_DATA), select the POM costs. The POM costs are then subtracted from the TOA Activity I costs to obtain the dollar balance available for the remaining program.

(c) From the PDIP Funding Sequence table (PFSTAB), obtain the PDIP funding sequence. This establishes the funding priority of the various package levels.

(d) From the ITEM DATA table (ITEM_DATA), obtain the percent of training requirements to be filled for items that are exceptions to the values obtained from the GOALS table (in 5c(1)(a) above).

(2) Get item priority sequence. (This routine builds the priority array which is used to establish the sequence in which items are processed within each PDIP.)

From the ITEM DATA table (ITEM_DATA), assign to each item either the DCSOPS priority or alternate priority number. (The DCSOPS priority number is the only priority number currently used. The alternate priority number can be used for 'what-if' studies.)

(3) For each PDIP (process PDIPs in order of funding).

(a) For each item within that PDIP (process the items in priority sequence):

(1) Check if Level 2 Training is allowed (packages 10 and 11). The level of training indicator is selected from the REVISION table (REVTAB). If Level 2 Training is not allowed, the shortfall reason code is set equal to 'F' (Level 2 Training not allowed), and the shortfall records are written out. If Level 2 Training is allowed, continue processing.

(2) For packages 6 and 11, check if the pipeline is filled over 3 years. If the pipeline is filled over a 3 year period, write the shortfall for the first 2 years with a reason code 'G' (pipeline filled over a 3 year period). If not filled over a 3 year period, continue processing.

(3) For package 5, check if training is restricted. If training is restricted, write the shortfall for the restricted portion with a reason code of 'I' (requirement exceeds percent of training allowed). If training is not restricted, continue processing.

(4) Fill requirements from inventory, if possible.

(a) If it is a test or training requirement, apply inventory in excess of the drawdown requirement.

(b) If it is a DOS requirement, then inventory not already allocated is assigned to the appropriate DOS package.

(5) If the requirement could not be satisfied from inventory, then:

(a) If the requirement exceeds the buildup goal, write the shortfall for the excess portion with a reason code of 'E' (requirement exceeds buildup target).

(b) Check if funds for production are available for shortfall requirements due to inventory or buildup constraints. If a funding constraint exists, write a shortfall record for the unfunded portion, with a reason code 'A' (lack of sufficient funds).

(c) Check if the item's unfilled requirement exceeds the MIA constraint. If it does, write a shortfall record for the excess portion with a reason code 'K' (maximum inventory allowed constraint).

(d) Schedule production for the unfilled requirement (repeat step 4b(4)).

(6) Schedule primary components to support the end item (repeat step 5b(5)).

(7) Accumulate secondary component requirements.

(8) Loop to the next item in priority sequence (return to step 5c(3)(a)).

(b) Process the secondary components for this package. For each secondary component with a requirement:

(1) Fill the requirement with inventory, if possible.

(2) If the requirement was not satisfied with inventory, then check if funds for production are available for the shortfall quantity. If a funding constraint exists, write a shortfall record for the unfunded portion with a reason code 'A' (lack of sufficient funds).

(3) Schedule production for the unfilled requirement (repeat step 5b(4)).

(4) Schedule primary components to support the secondary item (repeat step 5b(5)).

(5) Loop to the next secondary item with a requirement.

d. Write results. Results are written to the following tables:

(1) RESULT3 (Production Output Table).

(2) RESULT4 (Item Output Table).

(3) RESULT5 (Summary by Plant and Package or Service).

(4) RESULT6 (Summary by Plant and Line).

NOTE: The RESULT1 (Army Requirements Output Table) and RESULT2 (Other Requirements Output Table) tables are continually updated as each item is processed.

e. Process the Next Year's Requirements (repeat Loop 5 until all years are processed).

ANNEX D

FILE FORMATS

ICAPP Extract File

DATA ELEMENT	FORMAT	COLUMN	REMARKS
DODAC	A8	1-8	
Service Code	A2	9-10	NS = Navy Sea NA = Navy Air MC = Marine Corps AF = Air Force CG = Coast Guard OT = Other AR = Army*
SSN	A6	11-16	
Nomenclature	A50	17-66	
Requirements:			
Prior year	I10	67-76	
Current year	I10	77-86	
Budget year	I10	87-96	
POM year 1	I10	97-106	
POM year 2	I10	107-116	
POM year 3	I10	117-126	
POM year 4	I10	127-136	
POM year 5	I10	137-146	
Unit Prices:			
Prior year	F11.4	147-157	
Current year	F11.4	158-168	
Budget year	F11.4	169-179	
POM year 1	F11.4	180-190	
POM year 2	F11.4	191-201	
POM year 3	F11.4	202-212	
POM year 4	F11.4	213-223	
POM year 5	F11.4	224-234	

*Note that these Army requirements are not used since we got them from RDAISA.

☒ MAGNETIC TAPE
☐ MASS STORAGE

FILE LAYOUT

CONTROL NO.

PROJECT TITLE	PREPARED BY/DATE	PAGE
AAO DATA TAPE (MORGAN)	L.B.H. / 9 JAN 86	1 OF 5

LUN NO	FILE NAME/LABEL	CLASSIFICATION	CHAR/RECORD	WDS/PHYSICAL RECORD	LOG REC/PHY REC
	RECORD 1 FORMAT	U	132	22	1

1–5	6–8	9–11	12	13–132
SSN	YR	1		NOMENCLATURE

1–6	72	78	84	90	96	102	108	114
UOM	CATEGORY FUNDED PROFILE	MAJOR ELEMENT PROFILE	BEGINNING ASSETS	FY88 TEST LOSSES	FY89 TEST LOSSES	FY90 TEST LOSSES	FY91 TEST LOSSES	FY92 TEST LOSSES	BLANK

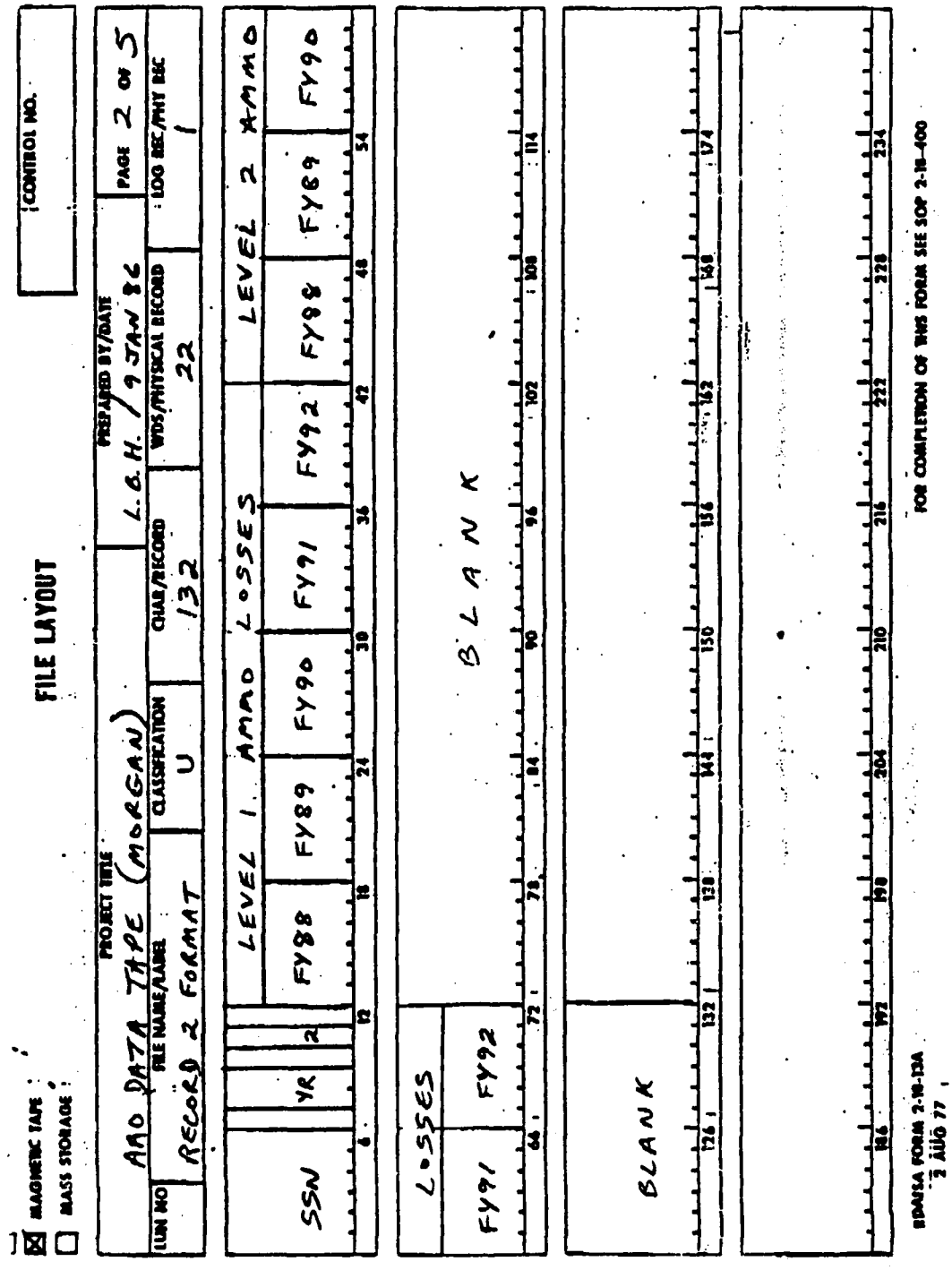
115–126	127–132	133–138	139–144	145–150	151–156	157–162	163–168	169–174
BLANK								

175–184	185–192	193–198	199–204	205–210	211–216	217–222	223–228	229–234

NOTE: EACH SSN AND YR HAS 5 RECORDS (DENOTED BY 1, 2, 3, 4, 5 IN COL. 11).

RDAISA FORM 2-18-13A
2 AUG 77

FOR COMPLETION OF THIS FORM SEE SOP 2-18-400


FILE LAYOUT
CONTROL NO.
MAGNETIC TAPE
MASS STORAGE
PROJECT TITLE
AMO DATA TAPE (MORGAN)
PREPARED BY/DATE
L.G.H. / 9 JAN 86
PAGE 2 OF 5
LUN NO
FILE NAME/LABEL
RECORD 2 FORMAT
CLASSIFICATION
U
CHAR/RECORD
132
WDS/PHYSICAL RECORD
22
LOG REC/PHYT REC
1
SSN
YR
2
LEVEL 1 AMMO LOSSES
FY88
FY89
FY90
FY91
FY92
LEVEL 2 AMMO
FY88
FY89
FY90
LOSSES
FY91
FY92
BLANK
BLANK
FOR COMPLETION OF THIS FORM SEE SOP 2-18-400
DRAFSA FORM 2-18-13A
2 AUG 77

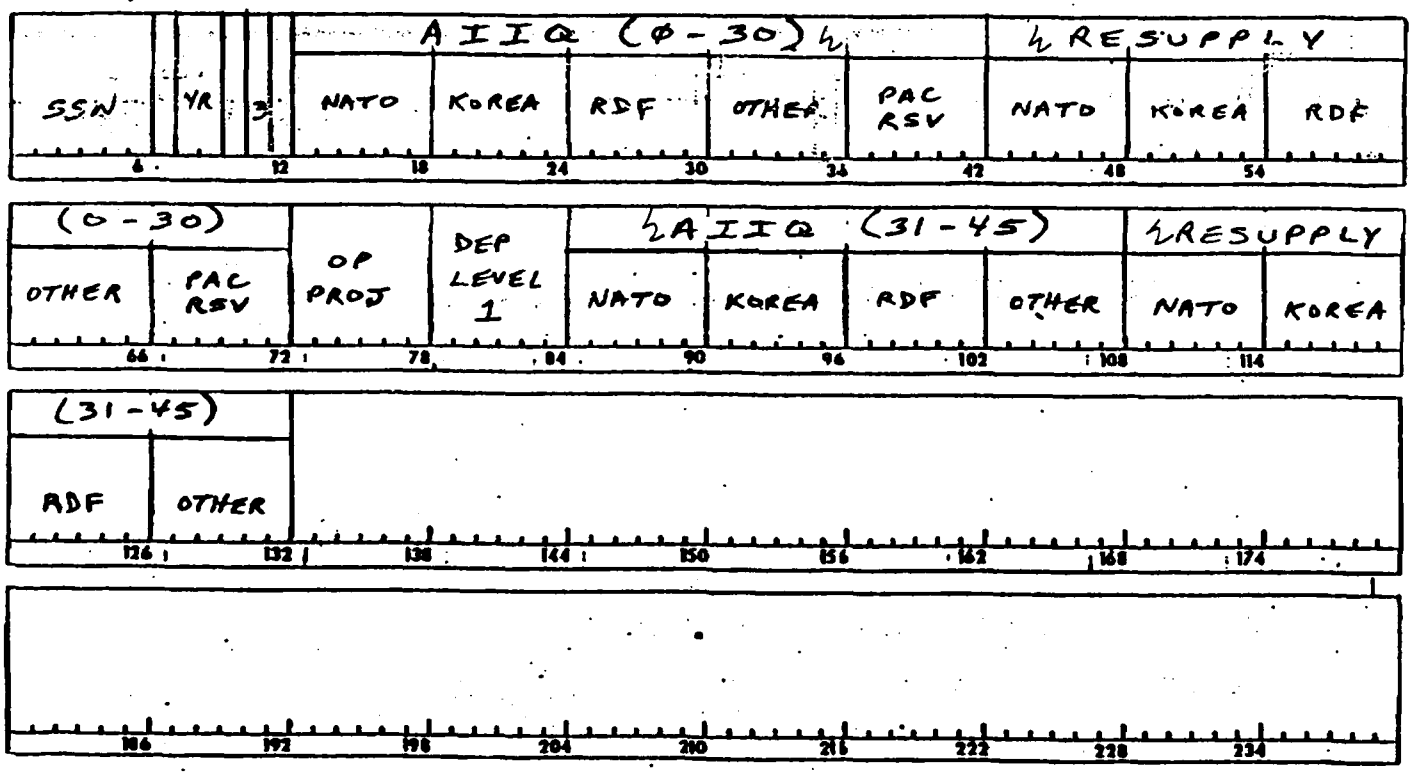
☒ MAGNETIC TAPE
☐ MASS STORAGE

FILE LAYOUT

CONTROL NO.

PROJECT TITLE	PREPARED BY/DATE	PAGE
AAO DATA TAPE (MORGAN)	L.B.H. / 9 JAN 86	3 OF 5

LUN NO	FILE NAME/LABEL	CLASSIFICATION	CHAR/RECORD	WDS/PHYSICAL RECORD	LOG REC/PHY REC
	RECORD 3 FORMAT	U	132	22	1



BDAMSA FORM 7-18-13A
2 AUG 77

FOR COMPLETION OF THIS FORM SEE SOP 2-18-400

ADSM 18-L62-LAT-ZZZ-MM-2604
30 November 1987

☒ MAGNETIC TAPE
☐ MASS STORAGE

FILE LAYOUT

CONTROL NO.

PROJECT TITLE	PREPARED BY/DATE	PAGE
AAO DATA TAPE (MORGAN)	L.B.H. / 9 JAN 86	4 OF 5

LUN NO	FILE NAME/LABEL	CLASSIFICATION	CHAR/RECORD	WDS/PHYSICAL RECORD	LOG REC/PHY REC
	RECORD 4 FORMAT	U	132	22	1

SSN	YR	M	½AIIQ (46-60) NATO	½AIIQ (46-60) KOREA	½AIIQ (46-60) RDF	½AIIQ (46-60) OTHER	½RESUPPLY (46-60) NATO	½RESUPPLY (46-60) KOREA	½RESUPPLY (46-60) RDF	½RESUPPLY (46-60) OTHER
6	12		18	24	30	36	42	48	54	

DEPOT LEVEL 2	½MOB TNG A 0-30	½MOB TNG A 31-45	½MOB TNG A 46-60	WRSA 0-30	½AIIQ (61-90) NATO	½AIIQ (61-90) KOREA	½AIIQ (61-90) RDF	½AIIQ (61-90) OTHER	BLANK
66	72	78	84	90	96	102	108	114	

BLANK
126 132 138 144 150 156 162 168 174

186 192 198 204 210 216 222 228 234

RDAISA FORM 2-18-12A
2 AUG 77

FOR COMPLETION OF THIS FORM SEE SOP 2-18-400

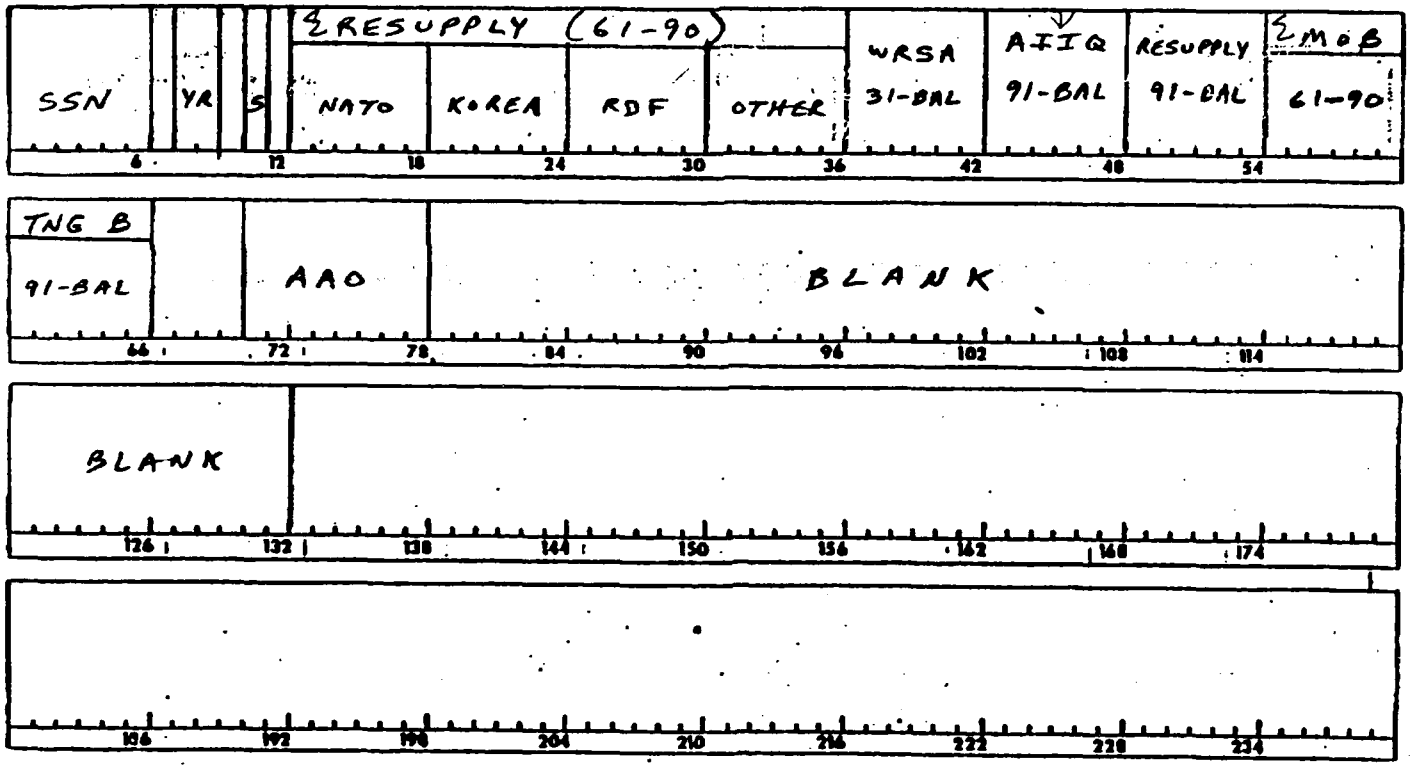
☒ MAGNETIC TAPE
☐ MASS STORAGE

FILE LAYOUT

CONTROL NO.

PROJECT TITLE	PREPARED BY/DATE	PAGE
AAO DATA TAPE (MORGAN)	L.B.H. / 9 JAN 82	5 OF 5

LUN NO	FILE NAME/LABEL	CLASSIFICATION	CHAR/RECORD	WDS/PHYSICAL RECORD	LOG REC/PHY REC
	RECORD 5 FORMAT	U	132	22	1



BRAISA FORM 2-10-13A
2 JUL 77

FOR COMPLETION OF THIS FORM SEE SOP 2-10-400

ADSM 18-L62-LAT-ZZZ-MM-2604
30 November 1987

ANNEX E

SUBROUTINE CROSS-REFERENCE

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
(Financial Program - PJSMDL)			
PJSMDL	Main program for the financial program. Initiates ORACLE login, terminal type, and character styles. Calls for graphics menu and exits program.		GORCLI GTMTYP GPTSYL DOLMEN
DOLMEN	Plot menu for financial plots.	PJSMDL	ZREADI
DOLPT1	Graphics routine for dollars grouped by families for plants. This routine sets up graphics common areas and defaults for display, and reads in data.	DOLMEN	GYEAR GFAMLY GPLANT DBDOL1 UDATIN BGNPLT
DOLTP2	Graphics routine for dollars grouped by PDIP packages for plants. This routine sets up graphics common areas and defaults for display and reads in data.	DOLMEN	GYEAR GPDIPS GPLANT DBDOL2 UDATIN BGNPLT
DOLTP3	Graphics routine for dollars by family for a plant-year. This routine sets up graphics common areas and defaults for display and reads in data.	DOLMEN	GYEAR GFAMLY GPLANT DBDOL1 UDATIN BGNPLT
DOLPT4	Graphics routine for dollars by year expended for TOA, hardware, and production base. This routine sets all plot parameters and reads in the data.	DOLMEN	GYEAR DBDOL4 UDATIN BGNPLT

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
DOLPT5	Graphics routine for dollars by year of plant of same type. This routine sets all plot parameters and gets data from the data base only. A plot for each of the three plant types is generated.	DOLMEN	GYEAR DBDOL5 BGNPLT
DOLPT6	Graphics routine for dollars by year grouped by the three plant types. This routine sets all plot parameters and reads in the data.	DOLMEN	GYEAR DBDOL6 UDATIN BGNPLT
DOLPT7	Graphics routine for dollars by year expended for TOA and POM. This routine sets all plot parameters and reads in the data.	DOLMEN	GYEAR DBDOL7 UDATIN BGNPLT
DOLPT8	Graphics routine for dollars by year grouped by training, war reserve, and production base. This routine sets all plot parameters and reads in the data.	DOLMEN	GYEAR DBDOL8 UDATIN BGNPLT
(Workload Program - PJSMWL)			
PJSMWL	Main program for the workload program. Initiates ORACLE login, terminal type, and character styles. Calls for graphics menu and exits the program.		GORCLI GTMTYP GPTSYL WRKMEN DBLOGO
WRKMEN	Plot menu for workload plots.	PJSMWL	ZREADI WRKPT1
WRKPT1	Graphics routine for work-year grouped by significant items for plants. This routine sets all plot parameters and reads in the data.	WRKMEN	GYEAR GPLANT GSIGIT GITEMS DBWRK1 UDATIN BGNPLT

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
WRKPT2	Graphics routine for work-years grouped by families for plants. This routine sets all plot parameters and reads in the data.	WRKMEN	GYEAR GFAMLY GPLANT DBWRK2 UDATIN BGNPLT
WRKPT3	Graphics routine for work-years grouped by PDIP packages for plants. This routine sets all plot parameters and reads in the data.	WRKMEN	GYEAR GPDIPS GPLANT DBWRK2
WRKPT4	Graphics routine for work-years grouped by service for plants. This routine sets all plot parameters and reads in the data.	WRKMEN	GYEAR GSERVC GPLANT DBWRK4 UDATIN BGNPLT
(Production Quantity Program - PJSMPD)			
PJSMPD	Main program for the production quantity program. Initiates ORACLE login, terminal type, and character styles. Calls for graphics menu and exits program.		GORCLI GTMTYP GPTSYL PRDMEN DBLOGO
PRDMEN	Plot menu for production quantity plots.	PJSMPD	ZREADI PRDPT1 PRDPT2
PRDPT1	Graphics routine for quantity by significant items for plants. This routine sets all plot parameters and reads in data.	PROMEN	GYEAR GPLANT GSIGIT GITEMS DBPRD1 UDATIN BGNPLT

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
(Requirements Program - PJSMRQ)			
PJSMRQ	Main program for the requirements program. Initiates ORACLE login, terminal types, and character styles. Calls for graphics menu and exits the program.		GORCLI GPTSYL REQMEN DBLOGO
REQMEN	Plot menu for requirements plots.	PJSMRQ	ZREADI REQPT1
REQPT1	Graphics routine for requirements by year grouped by service. This routine sets all plot parameters and reads in the data.	REQMEN	GYEAR GITEMS GSERVC DBREQ1 UDATIN BGNPLT
(Common Subroutine Package - PJSMCOM)			
ADDDLL	Adds a dollar sign to the end of a string for use by DISSPLA's self-terminating string option.	Display	
BGNPLT	Begins plotting at the end of a graphics routine by showing the PLOT/UPDATE menu or directly plotting the data.	Graphics	PLOTIT GUPDAT LSTDAT ZREADI
GETFAM	Returns family names from family number.	GFAMLY	
GETPLN	Contains a list of plant names and codes. Used in the code when not reading from the data base.	GPLANT	
GETPTY	Gets plot type from menu.	GUPDAT	ZREADI
GFAMLY	Selects family groups by menu.	DOLPT1 WRKPT2	ZREADI GETFAM
GITEMS	Gets user selected items.	GSIGIT WRKPT1 PRDPT1 REQPT1	ZREADI DBINAM

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
GORCLI	Queries user whether data base is to be used. If so, the ORACLE name and password is entered and login occurs.	PJSMDL PJSMWL PJSMPD PJSMRQ GUPDAT	ZREADI DBLOGI
GPDIPS	Gets user entered PDIP package number with the package name.	DOLPT2 WRKPT3	ZREADI
GPLANT	Gets the list of available plants and presents those plants as a menu for user selection.	DOLPT1 DOLPT2 DOLPT3 WRKPT1 PRDPT1	ZREADI GETPLN DBGPLT
GPTSYL	Gets desired plot style by selection of a character font from a menu.	PJSMDL PJSMWL PJSMPD PJSMRQ GUPDAT	ZREADI
GSERVC	Provides a menu of services and gets user selection.	WRKPT4 REQPT1	ZREADI
GSIGIT	Gets most significant items from a plant based on user selection of criteria. Set item legend information.	WRKPT1 PRDPT1	ZREADI GITEMS DBSIGD DBSIGQ DBSIGW DBINAM
GTMTYP	Presents menu and gets terminal type. Initializes DISSPLA postprocessor routine and sets color flag.	PJSMDL PJSMWL PJSMPD PJSMRQ	ZREADI
GUPDAT	Provides a menu and means to update graphics parameter in the graphics common block.	BGMPLT	ZREADI GETPTY GPTSYL GORCLI
GYEAR	Gets the first year for plots. If more than 1 year is requested, asks for number of years.	Graphics	ZREADI

ROUTINE NAME	DESCRIPTION	CALLED FROM	CALLS TO
LSTDAT	Lists to the screen a formatted display of the graphics common block.	BGMPLT	
MYPIE	Routine for use by DISSPLA to set colors for pie selections in pie charts.		
MYSPEC	Routine for use by DISSPLA to set legend colors to match data colors.		
NOTDON	Prints message to user that selected option is not available for use.		
PLOTIT	Begins plotting by making appropriate calls according to plot type selected. Initializes DISSPLA plotting if desired.	BGMPLT	Display
SETCOL	Makes DISSPLA calls to set color of user's choice.	Display	
UDATIN	Asks for a user data value. Must be entered as an integer.	Graphics	ZREADI
ZAESTH	Determines an aesthetic scale for plotting from a given minimum and maximum value.	Display	
ZREADI	Reads integer values from an input string in a free from manner.	Various	

DISTRIBUTION

NO. OF
COPIES

Commander
U.S. Army Materiel Command
5001 Eisenhower Avenue
Alexandria, VA 22333-0001

1 ATTN: AMCCA-CA
1 AMCPEO-AMMO

Commander
U.S. Army Armament, Munitions and Chemical Command
Rock Island, IL 61299-6000

1 ATTN: AMSMC-AP
6 AMSMC-SA

1 Director
Defense Logistics Studies Information Exchange
U.S. Army Logistics Management Center
Fort Lee, VA 23801-6043

1 Defense Logistics Agency
ATTN: DLA-LO
Cameron Station
Alexandria, VA 22314

2 Defense Technical Information Center
ATTN: DDA
Cameron Station
Alexandria, VA 22314

1 Commander
U.S. Army Research, Development and Engineering Center
ATTN: SMCAR-ASH-S
Picatinny Arsenal, NJ 07801-5001

ADSM 18-L62-LAT-ZZZ-MM-2604
30 November 1987

NO. OF
COPIES

1 Director
U.S. Army Materiel Systems Analysis Activity
ATTN: AMXSY-MP (Mr. Cohen)
Aberdeen Proving Ground, MD 21005-5071

1 Mr. Stephen M. Drezner
The RAND Corporation
1700 Main Street
P.O. Box 2138
Santa Monica, CA 90406-2138