This report provides a narrative description (Volume I) and PC SIMSCRIPT II.5 source code (Volume II) for a PC-based simulation of wholesale inventory management of clothing and textile (C&T) items as practiced at the Defense Logistics Agency's Clothing and Textiles Directorate, Defense Personnel Support Center, Philadelphia, Pennsylvania. The C&T simulation system includes a simulation model, a SIMSCRIPT program for preparation of input data, and an analytic variable safety level (VSL) model (also programmed in SIMSCRIPT) for computation of C&T VSLs. Simulated functions include demand forecasting (both program-oriented and historical-demand-based); C&T inventory control (the setting of quantitative levels, e.g., reorder point, procurement cycle quantity, acquisition objective); customer demand ("customers" being DoD retail supply points); and supplier responsiveness (leadtime variability).
SIMULATING CLOTHING AND TEXTILE OPERATIONS AT THE DEFENSE LOGISTICS AGENCY

VOLUME II: SIMSCRIPT SOURCE CODE

Report DL703R1

January 1989

Robert C. Kline
Christopher H. Hanks

Prepared pursuant to Department of Defense Contract MDA903-85-C-0139.
The views expressed here are those of the Logistics Management Institute at
the time of issue but not necessarily those of the Department of Defense.
Permission to quote or reproduce any part must – except for Government
purposes – be obtained from the Logistics Management Institute.

LOGISTICS MANAGEMENT INSTITUTE
6400 Goldsboro Road
Bethesda, Maryland 20817-5886
PREFACE

This report is published in two volumes. Volume I is a narrative description of the clothing and textiles (C&T) simulation system, which includes the C&T simulation itself, a "capture" program for preparing input data, and an analytic inventory model for computing variable C&T safety levels.

Volume II presents a listing of the PC SIMSCRIPT II.5 source code for each of the three parts of the system, alphabetical listings and brief descriptions of each procedure, and outline descriptions of the flow and interactions among procedures.
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preface</td>
<td>iii</td>
</tr>
<tr>
<td>Chapter 1. Introduction</td>
<td>1-1</td>
</tr>
<tr>
<td>Chapter 2. The C&amp;T Simulation Program</td>
<td>2-1</td>
</tr>
<tr>
<td>Flow Outline of Program</td>
<td>2-1</td>
</tr>
<tr>
<td>Listing of Procedures</td>
<td>2-2</td>
</tr>
<tr>
<td>Description of Procedures</td>
<td>2-3</td>
</tr>
<tr>
<td>Source Code</td>
<td>2-8</td>
</tr>
<tr>
<td>Chapter 3. The Capture Program</td>
<td>3-1</td>
</tr>
<tr>
<td>Flow Outline of Program</td>
<td>3-1</td>
</tr>
<tr>
<td>Listing of Procedures</td>
<td>3-1</td>
</tr>
<tr>
<td>Description of Procedures</td>
<td>3-1</td>
</tr>
<tr>
<td>Source Code</td>
<td>3-3</td>
</tr>
<tr>
<td>Chapter 4. C&amp;T Variable Safety Level Model</td>
<td>4-1</td>
</tr>
<tr>
<td>Flow Outline of Program</td>
<td>4-1</td>
</tr>
<tr>
<td>Listing of Procedures</td>
<td>4-1</td>
</tr>
<tr>
<td>Description of Procedures</td>
<td>4-2</td>
</tr>
<tr>
<td>Source Code</td>
<td>4-4</td>
</tr>
</tbody>
</table>
CHAPTER 1
INTRODUCTION

This volume contains the PC SIMSCRIPT II.5 source code for the three programs in the clothing and textiles (C&T) simulation system: the C&T simulation itself, the data "capture" program, and the C&T variable safety level (VSL) model. Narrative descriptions of those programs are presented in Volume I of this report. This volume lists source code, describes procedures, and outlines the general structure and flow of the system.

The simulation, capture, and VSL programs are treated in Chapters 2, 3, and 4, respectively. Each chapter has four sections. The first section outlines the flow of the program by listing the sequence of procedures that compose the program. Procedures are blocks of code that are called by the program in their entirety. (Procedures are often referred to as subroutines, routines, or modules in other computer languages.) In the outline, indented procedures are called by the procedure immediately above. Words in capitals are procedure names. Words in lower case are branches or loop instructions. Procedures followed by the phrase "every blank.interval", are time-dependent processes. Processes are executed after a specific time interval has elapsed in the simulation. For example, the "REVIEW.INVENTORY" process is executed after every "review.interval" has elapsed (every 2 days).

The second section in each chapter lists all the procedures in the program in alphabetical order. The list serves as a table of contents for the program. Procedures that start with the prefix "PRINT" serve only to print or display a simulation result.

The third section of each chapter provides a brief description of each procedure. Procedures listed in the third section also are in alphabetical order.

Together the first three sections give the general flow, location, and description of each procedure in the program, to make finding specifics in the code and understanding the program easier. The fourth section in each chapter contains the SIMSCRIPT source code for the program. After the "PREAMBLE" and "MAIN," which are required procedures in any SIMSCRIPT program, remaining procedures
appear in alphabetical order in the source listings. The convention in SIMSCRIPT is that code itself appears in capitals, while comments appear in lower case and are preceded by two single apostrophes (' '). Besides the code listing, any procedure can be printed individually from the PC once in SimLab. The command for this is "PRINT procedure.name."

SIMSCRIPT is designed to be more easily understood than most computer languages because of its "English-like" code. The three programs are written in PC SIMSCRIPT II.5. SIMSCRIPT is a structured, general-purpose language with specific features to support simulation. SIMSCRIPT is described in five separate manuals available from CACI, Inc.:

- **SIMSCRIPT II.5 Programming Language**
- **Building Simulation Models with SIMSCRIPT II.5**
- **SIMSCRIPT II.5 Reference Handbook, Second Edition**
- **SIMANIMATION User's Guide and Casebook.**

---

1SIMSCRIPT is a product of CACI, Inc. The Defense Logistics Agency (DLA) owns two copies of the PC SIMSCRIPT II.5 software package (with compilers) and two run-time copies (without compilers). This makes it possible for DLA to run four copies of the C&T simulation simultaneously. Additional run-time packages may be purchased from CACI for relatively small cost (from $200 to $500 per copy, depending on the number of copies purchased).
CHAPTER 2
THE C&T SIMULATION PROGRAM

The C&T simulation program is displayed in this chapter. The program resides in the subdirectory C:\SIM\DLA on the PC's hard disk. The chapter contains four sections: the outline flow of the program, the list of all procedures, a short description of each procedure, and the program source code.

FLOW OUTLINE OF PROGRAM

PREAMBLE
MAIN
SET.OPTIONS
PRINT.QUERIES
INPUT.SSCF.DATA
INPUT.MPT011.DATA
PGC.INITIALIZE
    If Nevassump.opt = true
        OPTIONAL.ASSUMPTIONS
PRINT.SSCF.DATA
    IF VSL.opt=true
        INPUT.VSL.DATA
XYZ.PLTS
MATRIX.DELIVERY.SCHEDULE
    If Delivery.opt = 1
        METHOD1.SCHEDULE
    else
        LAYINTO.MATRIX
PRINT.DELIVERY.MATRIX
DISTRIBUTION.DATA
RTC.REQUISIT.CUTOFF
GRAPH.INITIALIZE
SIMULATION.RUN now
    WARMUP.RESET after Warmup.period
    If ICC = "P" then POI item
        UPDATE.CTREQ.MAT now
    COMPUTE.ROP.PCP every ROP.review
        SUM.FORECAST.OVER.TIME
    PRINT.ASSUMPTIONS
    SET.SIMULATED.DDR every 30 days
    REVIEW.INVENTORY every Review.interval
    If breach
        PLACE.PGC.ORDER
            CALC.ORDER.QTY
            wait LT
            For all deliveries
RECEIVE.PGC.ORDER
PRINT.ORDER
COVAR.SAMPLING every Covar.interval
CONFIDENCE INTERVAL
PRINT.QUICK.COVAR every Quick.interval
For all MSNs
DEMAND.GENERATOR every Demand.interval
REQ.TO.INVENTORY
PRINT.LEVELS every Trace.interval
PRINT.DEMANDS every Trace.interval
PRINT.ATEND
GET.PLOT.DATA every Plot.interval
Wait End.of.simulation days
PLOT.ATEND
PRINT.PGCSTATS
ADD.ALL.PGCS
Stop

LISTING OF PROCEDURES

PREAMBLE
MAIN
ADD.ALL.PGCS
CALC.ORDER.QTY
COMPUTE.ROP.PCP
CONFIDENCE INTERVAL
COVAR.SAMPLING
DEMAND.GENERATOR
DISTRIBUTION.DATA
GET.PLOT.DATA
GRAPH.INITIALIZE
INPUT.MPT011.DATA
INPUT.SSCF.DATA
INPUT.VSL.DATA
LAYINTO.MATRIX
MATRIX.DELIVERY.SCHEDULE
METHOD1.SCHEDULE
OPTIONAL.ASSUMPTIONS
PGC.INITIALIZE
PLACE.PGC.ORDER
PLOT.ATEND
PRINT.ASSUMPTIONS
PRINT.ATEND
PRINT.DELIVERY.MATRIX
PRINT.DEMANDS
PRINT.LEVELS
PRINT.ORDER
PRINT.PGCSTATS
PRINT.QUERIES
PRINT.QUICK.COVAR
PRINT.SSCF.DATA
RECEIVE.PGC.ORDER
REQ.TO.INVENTORY
DESCRIPTION OF PROCEDURES

PREAMBLE

CLOTHING AND TEXTILE SIMULATION MODEL (directory DLA) basic features:

1) options: requisitions/unit demands, constant monthly average DDR/Poisson DDR,

2) NSN and recruit training centers separate

3) ROP & PCP computations

4) variables: stock, EBO, AVBOD, fill rates, demands

5) Normalized CTREQ.MAT that is shifted & filled in to allow multi-year runs

6) Dynamic graphics include: dynamic plot of net stock by PGC or NSNs, histogram of % time with backorders, 4 fill rate meters, and EBO and demand pie charts.

7) PLT distribution as random linear variable from CLIN report

8) Covariance and confidence interval

9) Restoring statistics after a warmup period

10) MAD by NSN for demand generation

11) Phase Deliveries

12) PGC plot and histogram

13) PLT Knob to vary the CLIN distribution shape & variance

14) Demand Knob makes the mean demand a % > or < the forecast mean

15) QFD considered alone or with POI

16) 4 matrix delivery schedules with assumptions from MPT011 table

17) a VSL option with VSL months read in from external file

18) 2 options accumulates stats over many runs same PGC or different

19) A maximum of 12 queries

MAIN

This routine has the basic initialization steps and data input before the actual simulation starts stepping through time

ADD.ALL.PGCS

This routine reads previous PGC tallied results and adds current PGC results to it. If results in the file are from different PGCs, it also sums all PGCs and prints grand total to a file

CALC.ORDER.QTY

This routine calculates the order quantity for a NSN in a PGC. The quantity equals the difference between the current inventory position (onhand + onorder - backorders) and the PTAO (peace time acquisition objective)
COMPUTE.ROP.PCP
This process computes PCP in months, ROP for all NSNs every 30 days.

CONFIDENCE INTERVAL
This routine calculates the covariances for all lags and the confidence intervals.
General covariance formula for the kth lag (Ck):
\[ C_k = \frac{1}{N-k} \sum (X_i - \text{ave}X)(X_{i+k} - \text{ave}X) \] : sum is for 1 to N-k
The one pass formula
\[ C_k = \frac{1}{N-k} \left\{ \sum X_i X_{i+k} - (k+N) \text{ave}X^2 + \text{ave}X \left[ \sum X_{1..k} + \sum X_{n-k+l..N} \right] \right\} \]
where \[ \text{PRODUCT.MAT(NSN,K)} \]
\[ \text{SUM.K.ENDS(NSN,K)} \]

COVAR.SAMPLING
This process samples and updates required variables to estimate the covariances for each NSN and the PGC. (see Gross p418.)

DEMAND.GENERATOR
This process generates demands and requisitions for a given NSN

DISTRIBUTION.DATA
This routine initializes random variables distributions for the
PLT.DAY.DELAY (CLIN report), REQUISITION.RATIO.F(UUSIMS < 5
requisition distribution), and DEMAND.MAPE.F from Orchowsky's POI report pg. 6

GET.PLOT.DATA
This process gets or calculates several plotted variables for
dynamic net stock plot & static plotted histogram graphics.

GRAPH.INITIALIZE
This routine initializes graphics at the start of program: shows
the pie charts, determine histogram intervals (.5 of safety level)
and displays the dynamic traces

INPUT.MPT011.DATA
This routine Reads management policy table 11 and gets the minimum
procurement cycle, PGC delivery percents for all delivery
increments, 1 of 4 methods of delivery, PGC first delivery in days.

INPUT.SSCF.DATA
This routine reads the required input data to run the simulation
original captured from the Special Supply Control File Report via
a SIMSCRIPT program in directory DLADATA. The routine finds the
desired PGC number, and reads in the data into the appropriate
variables. If the PGC number is not found the program prints
error message and stops

INPUT.VSL.DATA
This routine is called if user specifies the VSL option as true
question (4). It searches the "VSL.DAT" file for the PGC number
and overrides the fix safety level values from the SSCF with the
VSL values in months in the file.
LAYINTO.MATRIX
This routine develops X, Y, Z amounts of delivery for all delivery months. Example: It takes the Z item target of 100% in 6th month and the total percent SUM of Z items for the PGC (e.g., 20%). It then makes sure the target times the SUM will not exceed the PGC DELIVERY.PERCENT (e.g., 15%) by month. Since it does exceed the PGC.DELIVERY.PERCENT (20>15), it takes the overflow (5%) and moves it into the previous month (the 5th month). The 5% now becomes the target for the 5th month and the cycle repeats.

MATRIX.DELIVERY.SCHEDULE
This routine lays in the different X, Y, Z percent deliveries per month vector into the XYZ.MATRIX for each of the 3 delivery methods. Method 1 does not use the X,Y,Z percents but a sort in routine METHOD1.SCHEDULE

METHOD1.SCHEDULE
For method 1, the delivery is made in clumps, not spread over several months like other methods. So take first months delivery percent and bring in as many NSNs as month can handle (e.g. 10%). Set the XYZ.MONTH value at 1 to mean bring entire order in first month. If an NSN can have 50% of its order brought in in the current month do so, if not have NSN be brought in next month.

OPTIONAL.ASSUMPTIONS
This routine lets the user override the standard assumptions, options or trace settings found in PGC.INITIALIZE & SET.OPTIONS and lets the user specify there own by editing the file ASSUMP.MOD & entering 1 in the user query (10), select alternate Assumption file.

PGC.INITIALIZE
This routine initializes some of the basic PGC variables such as time intervals between processes, PC variables, mean FORECASTs, and covariance information.

PLACE.PGC.ORDER
This process checks the inventory position of all NSNs in the PGC at the time of breach. The process determines whether any of the other NSNs will breach their ROP within the next minimum procurement cycle. It then calls the CALC.ORDER.QTY to determine the specific NSN order quantity of the NSNs that will be ordered. The process then waits an ALT + 1ST delivery days + a PLT delay before calling RECEIVE.PGC.ORDER. It then waits 30 days for each additional phased order (again calling RECEIVE.PGC.ORDER) until the entire order is received.

PLOT.ATEND
This routine plots the histogram, BO & demand pie charts, fill rate meters at the end of program run.

PRINT.ASSUMPTIONS
Prints all pertinent assumptions and variables for the run including options, query answers, safety level, OWRM, PLT, ALT, M1, M2, T, COST ARS, RTC CUTOFF, VIP, XYZ. MONTH ETC.
PRINT.ATEND
This routine prints the table of summary statistics during and at the end of simulation: requisition vs unit, total vs RTC, for EBO, AVBOD, fill rates, and demands/yr.

PRINT.DELIVERY.MATRIX
This routine prints the delivery matrix: the 3 rows are X, Y, Z; columns are for the number of deliveries. Values are the fraction of the total item (X,Y,Z) order delivered that month (i.e. each row sums to 1).

PRINT.DEMANDS
This process gives requisition size and interval, AMP, AMP/AMD ratio, onhand, onorder information at specified intervals, calls PRINT.ATEND, and gives cumulative BOS and demands.

PRINT.LEVELS
This process prints the requirements, PC, MIN.PC, stock, backorders at specified intervals.

PRINT.ORDER
Prints the order quantity placed for all NSNs, PLT delays, when will come in, and inventory positions all at time of breach and when order completely delivered.

PRINT.PGCSTATS
This routine prints the summary statistics at the end of the simulation at a PGC level. Specifically, backorders, availabilities, and annual demands (requisition & unit levels); onhand, onorder, orders/yr values; and calibration information confidence intervals, % onorder to total stock, etc.

PRINT.QUERIES
This routine prints the answers entered by user during interactive session.

PRINT.QUICK.COVAR
This process uses an approximation formula to estimate the covariance continuously at intervals throughout the simulation. Used primarily to determine end of warmup period and length of run as well as the rate of confidence interval change. Uses info. collected by COVAR.SAMPLING process and automatically printed for long runs.

PRINT.SSCF.DATA
This routine prints the SSCF data read in by routine INPUT.SSCF.DATA.

RECEIVE.PGC.ORDER
This routine adjusts stock and backorders when a stock shipment is received from suppliers. It is called by PLACE.PGC.ORDER. It uses the XYZ.MATRIX for methods 2, 3, 4 and XYZ.MONTH for method 1.

REQ.T0.INVENTORY
This routine updates on hand STOCK and if necessary updates NSN and recruit backorders (BO) when ever a requisition/customer demand is felt.

REVIEW.INVENTORY
This process reviews the inventory every REVIEW.INTERVAL days to see if inventory position IP (onorder + stock + BO) < ROP. If so will activate PLACE.PGC.ORDER to determine which NSNs and how much to buy

RTC.REQUISIT.CUTOFF
This routine automatically determines the requisition size cutoff. All requisition sizes above cutoff will be assumed to come from the Recruit Training Centers and if summed their percent demand would equal the PER.RTC.DEMAND. This routine finds the point in the Requisition distribution where those conditions are meet.

SET.OPTIONS
This key routine is where all options are set, queries are asked, traces are defined and set, and I/O units are defined.

SET.SIMULATED.DDR
This process updates monthly DDR for the simulation. First converts the forecast value to simulated monthly demand via MAPE, MAD, and demand KNOB factors if activated. Then divides the monthly value (30 days/demand interval) to get a daily demand rate (DDR). Note if demand.interval > 1 could be demand for 2, 10, 15 days, whatever.

SIMULATION.RUN
This process gives the general structure of the simulation and starting point for all processes.

SUM.FORECAST.OVER.TIME
This routine sums the CT POX and QFD requirements over the given period (TIME.V to TIME.V + PERIOD) to get a total FORECAST. The PERIOD is in months or month fractions, a real number, and is used to sum PCP, Safety level, ROP, MIN.PC, values. With POX items, this routine can sum 8 years of monthly data, however for non POI items the monthly demand does not change over time but is QFD/3.

UPDATE.CTREQ.MAT
This process makes sure there are enough future months of POI forecasts so that all levels (ROP and PTAO) can be calculated. This process determines the mean and standard deviation for normal distribution from the input CTREQ.MAT. Also, Every CTREQ.period this process shifts the CTREQ values a period up in the matrix so that old values are discarded. It then fills in the empty last period spots in the matrix with newly generated CTREQ from the normal distribution.

WARMUP.RESET
This process resets all appropriate statistics back to zero once the initial warmup period is over and the transient effects have apparently been washed out of simulation. This is so the final statistics at end of simulation are not affected by warmup
period.

XYZ.PLTS

This routine determines which NSN are X, Y, or Z items, and based on delivery method 1 to 4, the PLTs for each NSN.

SOURCE CODE

PREAMBLE

"'CLOTHING AND TEXTILE SIMULATION MODEL (directory DLA) basic features:
  1) options: requisitions/unit demands, constant monthly average
  2) NSN and recruit training centers separate
  3) ROP & PCP computations
  4) variables: stock, EBO, AVBOD, fill rates, demands
  5) Normalized CTREQ.MAT that is shifted & filled in to allow multi-year runs
  6) dynamic graphics include: dynamic plot of net stock by PGC or NSNs, histogram of % time with backorders, 4 fill rate meters, and EBO and demand pie charts.
  7) PLT distribution as random linear variable from CLIN report
  8) Covariance and confidence interval
  9) Restoring statistics after a warmup period
 10) MAD by NSN for demand generation
 11) Phase Deliveries
 12) PGC plot and histogram
 13) PLT Knob to vary the CLIN distribution shape & variance
 14) Demand Knob makes the mean demand a % > or < the forecast mean
 15) QFD considered alone or with POI
 16) 4 matrix delivery schedules with assumptions from MPT011 table
 17) a VSL option with VSL months read in from external file
 18) 2 options accumulations stats over many runs same PGC or different
 20) A maximum of 12 queries

NORMALLY MODE IS UNDEFINED

Since this period is undefined.

PROCESSES INCLUDE

UPDATE.CTREQ.MAT, "'shifts & inserts the CTREQ values in a period
COMPUTE.ROP.PCP, "'computes procurement cycles (PCP.MONTH) & ROP
SET.SIMULATED.DDR, "'sets monthly DDR (given: demand knob & MAD)
PRINT.LEVELS, "'start of months ROP, PCP, BOs, orders, stock
PRINT.DENDS, "'prints end of month sim. DMD & EBOs: NSNs & RTC
REVIEW.INVENTORY, "'continuously checks inventory for breaches
PLACE.PGC.ORDER, "'orders NSNs stock, waits a LT (PLT+ALT+DELAY)
SIMULATION.RUN, "'starting sequences of all simulation processes
GET.PLOT.DATA, "'gets net stock plot variables every x days
COVAR.SAMPLING, "'samples & performs some Covariance calculations
PRINT.QUICK.COVAR, "'approximates covar continuously over run
WARMUP.RESET "'after warmup period, a reset cum. statistics
EVERY DEMAND.GENERATOR"generates demand & requisition given a NSN
HAS A NSN.D

DEFINE NSN.D AS AN INTEGER VARIABLE

2-8
PRIORITY ORDER IS UPDATE.CTREQ.MAT, DEMAND.GENERATOR, PLACE.PGC.ORDER, REVIEW.INVENTORY, PRINT.DEMANDS, GET.PLOT.DATA, WARMUP.RESET, COMPUTE.ROP.PCP, SET.SIMULATED.DDR, COVAR.SAMPLING, PRINT.QUICK.COVAR, SIMULATION.RUN, PRINT.LEVELS

PERMANENT ENTITIES
EVERY NSN.ATTRIBUTES HAS ''key attributes for each NSN
A PLT.DAY, ''procurement leadtimes in days
A ARS, ''average requisition size
A FORECAST.MTH, ''the current months forecast of both POI & QFD
A AVE.FORECAST, ''AMF over course of simulation: CTREQ+QFD/3
A ROP.QTY, ''reorder point in units
A PCP.MONTH, ''procurement cycle period in months
A SAFETY.MONTH, ''safety level in months either VSL or FSL
A DDR, ''mean daily (or few day) demand rate demand for the month
A SIM.DDR, ''actual daily demand from poisson (else=DDR)
A ONORDER, ''outstanding onorder items yet to be received
A STOCK, ''in stock items or onhand at inventory
A NET.STOCK, ''STOCK-BO at a point in time for plots
A OWRM, ''other war reserve material protectable units
A RECRUIT.SIZE.CUTOFF, ''requisition sizes above cut are from RTCs
A PER.RTC.DEMAND, ''the percentage of recruit to total demand
A VIP.ITEM, ''1=yes VIP(monthly ROPT), 0 Not VIP (quarterly)
A MAD, ''mean absolute deviation in QTR demand (monthly if VIP)
A QFD, ''quarterly forecast demands directly from SSCF
A ALPHA, ''alpha factor from SSCF
A XYZ.MONTH, ''X,Y,Z items have a 1,2,3 so that proper %
''delivery vector is used (if method 1 means month delivered)
A NSN.NO ''the NSN number
OMNS A REQ.BO.QUEUE ''requisition backorder queue

DEFINE PLT.DAY, ROP.QTY, PCP.MONTH, PC.EOQ, ARS, MAD, XYZ.MONTH, DDR, RECRUIT.SIZE.CUTOFF, PER.RTC.DEMAND AS REAL VARIABLES
DEFINE ALPHA,QFD, SAFETY.MONTH, SIM.DDR, AVE.FORECAST, FORECAST.MTH, OWRM, NET.STOCK, STOCK, ONORDER AS REAL VARIABLES
DEFINE VIP.ITEM AS INTEGER VARIABLE
DEFINE REQ.BO.QUEUE AS A FIFO SET
DEFINE NSN.NO AS TEXT VARIABLE

''statistical information on backorders for total & RTCs
EVERY DEMAND.BO AND NSN.DETAIL HAS ''detail dimension for recruits
A REQ.SIZE, ''requisition & sum is unit demand for NSN & RTCs
A REQ.INTERVAL, ''requisition time interval for NSN & recruits
A REQ.BO, ''requisition backorders for EBOs (NSN and recruits)
A UNIT.BO, ''total unit backorders for EBOs (NSN & recruits)
A SUM.REQ.BO, ''sum of backorder requisitions used in fill rate
A SUM.UNIT.BO''sum of unit BO used in fill rate calc.
DEFINE REQ.SIZE, REQ.BO, UNIT.BO AS REAL VARIABLES
DEFINE SUM.UNIT.BO, SUM.REQ.BO, REQ.INTERVAL AS REAL VARIABLE

EVERY PLOTNSN HAS ''a plot var with all NSN + total + recruits
A FILLRATE ''1 - BO/DEMAND * 100, used in plotting meters
DEFINE FILLRATE AS REAL VARIABLE
EVERY COVAR.INFO HAS
A COVAR.DATA, ''used in calc. mean & variance of COVAR sample
OWNS A COVAR.SET ''contains last k previous samples
DEFINE COVAR.SET AS A FIFO SET
DEFINE COVAR.DATA AS A REAL VARIABLE

TEMPORARY ENTITIES
EVERY REQ.BO.MEMBER
HAS A BO.SIZE, ''unit backorders for a requisition
A BO.TYPE, ''either recruit or total NSN
BELONGS TO THE REQ.BO.QUEUE
DEFINE BO.TYPE AS INTEGER VARIABLES
DEFINE BO.SIZE AS REAL VARIABLES
EVERY COVAR.MEMBER ''sample BO data points w/ k points in set
HAS A DATA.POINT
BELONGS TO THE COVAR.SET
DEFINE DATA.POINT AS REAL VARIABLES

'' timing characteristics of simulation
DEFINE DAYS TO MEAN UNITS
DEFINE END.OF.SIMULATION,
LENGTH.OF.SIMULATION AS A REAL VARIABLE

'' PGC characteristics
DEFINE MAX.MONTH, ''number of months IN POI CTREQ forecasts
MAX.NSN AS INTEGER VARIABLE '' number of NSNs in PGC
DEFINE COST AS REAL VARIABLES
DEFINE PGC.NAME AS TEXT VARIABLE
DEFINE ICC AS TEXT VARIABLE ''type of requirements calculation
DEFINE FSC AS INTEGER VARIABLE ''federal supply code
DEFINE PGC.NO AS INTEGER VARIABLE ''PGC code number
DEFINE MIN.PC AS A REAL VARIABLE ''min. procurement cycle(MPT 11)
DEFINE SIM.PLT.DAY, ALT.DAY AS A INTEGER VARIABLES
simulated PLT used in order delay, PLT.DAY used in levels
DEFINE PGC.NET.STOCK AS REAL VARIABLE ''for plot & histogram
DEFINE PGC.SL.STOCK AS REAL VARIABLE ''PGC safety level stock
DEFINE RUN.ID AS REAL VARIABLE ''ID when run PGC more than once

THE SYSTEM HAS
A DEMAND.MAPE.F RANDOM LINEAR VARIABLE, ''mean & error in demand
or the ratio of forecast to actual demand
A REQUISITION.RATIO.F RANDOM LINEAR VARIABLE, ''ratio of size/ARS
distribution from USIMs
A PLT.DAY.DELAY.F RANDOM LINEAR VARIABLE'' CLIN PLT distribution
DEFINE DEMAND.MAPE.F AS A REAL, STREAM 10 VARIABLE
DEFINE REQUISITION.RATIO.F AS REAL, STREAM 9 VARIABLE
DEFINE PLT.DAY.DELAY.F AS REAL, STREAM 7 VARIABLE
DEFINE CTREQ.MAT AS A REAL, 3-DIMENSIONAL ARRAY
NSN specific means and stand. deviation of requirement matrix
DEFINE MEAN.CTREQ AND STD.CTREQ AS A REAL, 1-DIMENSIONAL ARRAYS
DEFINE TARGET.PGC AS INTEGER VARIABLE ''PGC looking for to get data
matrices & VAR for COVAR.SAMPLING & CONFIDENCE INTERVAL routines
DEFINE PRODUCT.MAT AS A REAL, 2-DIMENSIONAL ARRAY ''covar. product
holds sum for first and last k items
DEFINE SUM.K.ENDS AS A REAL, 2-DIMENSIONAL ARRAY

2-10
DEFINE PGC.NUM, K.LAG, M.COVAR, N.BLOCKS AS INTEGER VARIABLES
DEFINE CONF.INTV AS A REAL VARIABLE "'confidence interval derived

DEFINE XYZ.SUM AS A REAL, 1-DIMENSIONAL ARRAY 'sum of all X NSNs % of PCP demand, (same for Y, & Z items in matrix delivery scheme.
DEFINE MAX.DELIVERIES,"' no. of months of deliveries for the PGC (MPT011)
FIRST.DELIVERY, ''days of PLT before a NSN is delivered
Z.PERCENT,'''% item <= z% of PC*DEMAND for matrix deliveries
X.PERCENT '''% item >= x% of PC*DEMAND, Y item remainder

AS REAL VARIABLES
DEFINE M1, M2, T AS REAL VARIABLE 'used in procurement cycle PCP
DEFINE DELIVERY.PERCENT AS A REAL, 1-DIMENSIONAL ARRAY 'percnet
PGC order delivered each month in matrix delivery
DEFINE XYZ.MATRIX AS REAL, 2-DIMENSIONAL ARRAY 'XYZ matrix deliv. %
DEFINE MONTHLY.MAPE AS REAL VARIABLE 'MAPE for month

DEFINE ORDER.NUMBER,CTREQ.PERIOD, MAX.CTREQ.DIM AS INTEGER VARIABLE
DEFINE AT.MONTH,MONTH.I,NSN.I AS INTEGER VARIABLES 'array indices
simulation options & traces below, see SET.OPTIONS for definitions
DEFINE PLT.OPT, ''PLT Knob: 0 no variance, 1= CLIN, >1 a & of CLIN
DMDMAD.OPT ''Demand Knob 0 no MAD variance, 1 uses MAD,
>1 then ratio * forecast (eg .95 demand mean 95% of forecast
AS REAL VARIABLE
DEFINE NORMAL.OPT, DOREQ.OPT, POISSON.OPT, MAPE.OPT, VSL.OPT,
NEWASSUMP.OPT, COVARNIS.OPT, SHORT.OPT, DELIVERY.OPT, BATCH.OPT,
MODIFIEDATA.OPT, MODMPT011.OPT, ADDPGC.OPT AS INTEGER VARIABLES
DEFINE TRACE1, TRACE2, TRACE3, TRACE4, TRACE5, TRACE6, TRACE7,
TRACE8, TRACE9, TRACE10, TRACE11, TRACE12, TRACE13, TRACE14,
TRACE15, TRACE16, TRACE17, TRACE18, TRACE19, TRACE20, TRACE21,
TRACE22, TRACE23, TRACE24, TRACE.INTERVAL,
PLOT.INTERVAL, DEMAND.INTERVAL, REVIEW.INTERVAL, COVAR.INTERVAL,
QUICK.INTERVAL, WARMUP.PERIOD AS INTEGER VARIABLES

 constants
DEFINE .TOTAL TO MEAN 1 'next 3 constants are the columns
DEFINE .RECRUIT TO MEAN 2 ' of the CTREQ.MAT array
DEFINE .OTHER TO MEAN 3
DEFINE .TRUE TO MEAN 1
DEFINE .FALSE TO MEAN 0
DEFINE .DPM TO MEAN 30 'DAYS PER MONTH
DEFINE .MINVAL TO MEAN 0.000000000001
DEFINE HIS.BO.1, HIS.BO.2, HIS.BO.3 AS INTEGER VARIABLES

DISPLAY VARIABLES INCLUDE NET.STOCK, PGC.NET.STOCK, FILLRATE,
EDO.PIE, DEMAND.PIE 'backorders & demand values for pie chart
DEFINE EDO.PIE, DEMAND.PIE AS A REAL, 1-DIMENSIONAL ARRAY
DEFINE PGC.PLOT AS REAL VARIABLE
DEFINE PLOT.YSCALE AS REAL VARIABLE 'scales PGC net stock, Y axis
most of next tally are variables in routine PRINT.DEMANDS
TALLY AVE.MAPE AS THE MEAN OF MONTHLY.MAPE
TALLY SUM.FORECAST AS THE SUM OF FORECAST.MTH
TALLY AVE.REQ.SIZE AS THE MEAN, SUM.REQ.SIZE AS THE SUM, NO.REQ.SIZE
AS THE NUMBER OF REQ.SIZE 'sum & num is for unit & reqt. demands

2-11
TALLY AVE.REQ.INTERVAL AS THE MEAN OF REQ.INTERVAL
TALLY AVE.SIM.PLT AS THE MEAN OF SIM.PLT.DAY
TALLY AVE.COVAR.DATA AS THE MEAN, VAR.COVAR.DATA AS THE VARIANCE
OF COVAR.DATA
TALLY HIST.PGC.STOCK(HIS.BO.1 TO HIS.BO.2 BY HIS.BO.3) AS THE
HISTOGRAM, AVE.PGC.NET.STOCK AS THE MEAN OF PGC.NET.STOCK
TALLY HIST.PGC.PLOT(HIS.BO.1 TO HIS.BO.2 BY HIS.BO.3) AS THE
HISTOGRAM OF PGC.PLOT
ACCUMULATE AVE.REQ.EBO AS THE MEAN
OF REQ.BO 'time weighted BOS or EBOs
ACCUMULATE AVE.UNIT.EBO AS THE MEAN OF UNIT.BO 'time weighted EBOs
ACCUMULATE AVE.STOCK AS THE MEAN OF STOCK 'time weighted NSN stock
ACCUMULATE AVE.ONORDER AS THE MEAN OF ONORDER 'by NSN

MAIN
'This routine has the basic initialization steps and data input before
the actual simulation starts stepping through time
CALL SET.OPTIONS
CALL INPUT.SSCF.DATA
CALL INPUT.NPT11.DAY
CALL PGC.INITIALIZE
CALL PRINT.SSCF.DATA
IF VSL.OPT=.TRUE
    CALL INPUT.VSL.DATA
ALWAYS
CALL XYZ.PLOTS
CALL MATRIX.DELIVERY.SCHEDULE
CALL DISTRIBUTION.DATA
CALL RTC.REQUISIT.CUTOFF
CALL GRAPH.INITIALIZE
ACTIVATE A SIMULATION.RUN NOW
START SIMULATION

ROUTINE ADD.ALL.PGCS GIVEN NEWPGC
'This routine reads previous PGC tallied results and adds current
PGC results to it. If results in the file are from different PGCs
it also sums all PGCs and prints grand total to a file
DEFINE ROW, COL, MAX.COL, MAX.PGC AS INTEGER VARIABLES
DEFINE SUM.PGC, NEWPGC AS REAL, 1-DIMENSIONAL ARRAY
DEFINE PGC.MAT AS REAL, 2-DIMENSIONAL ARRAY
MAX.COL=11
MAX.PGC=1
IF (ADDPGC.OPT=0)
    'THEN don't add this PGC to accumulated PGC info from previous runs
    RETURN
ALWAYS
IF (ADDPGC.OPT>10)
    'THEN 1st run no reads
RESERVE PGC.MAT(*,*) AS MAX.PGC BY MAX.COL
ALWAYS
RESERVE NEWPGC(*), SUM.PGC(*) AS MAX.COL
IF (ADDPGC.OPT=1) OR (ADDPGC.OPT=10)
  "" THEN enter the run ID number instead of the PGC number
  NEWPGC(1)=RUN.ID
ALWAYS
OPEN UNIT 17 FOR INPUT, FILE NAME IS "C:\SIM\DLA\ALLPGCS.DAT"
USE UNIT 17 FOR INPUT
IF ADDPGC.OPT < 10
  "" THEN not first PGC so read existing information and store
  SKIP 2 INPUT RECORDS
  READ MAX.PGC
  MAX.PGC=MAX.PGC + 1
  RESERVE PGC.MAT(*,*) AS MAX.PGC BY MAX.COL
  SKIP 5 INPUT RECORDS
  FOR ROW = 1 TO (MAX.PGC-1) DO
    FOR COL = 1 TO MAX.COL
      READ PGC.MAT(ROW,COL)
  LOOP
ALWAYS
  "" stores current PGC in last row of summary statistics
  FOR COL = 1 TO MAX.COL
    PGC.MAT(MAX.PGC,COL) = NEWPGC(COL)
  CLOSE UNIT 17
OPEN UNIT 18 FOR OUTPUT, FILE NAME IS "C:\SIM\DLA\ALLPGCS.DAT"
USE UNIT 18 FOR OUTPUT
PRINT 6 LINE WITH MAX.PGC THUS
------------------------------- AGGREGATE PGC REPORT ------------------------
** PGC RESULTS IN SUMMARY (FILE ALLPGCS.DAT)

===AVERAGE== ===REQT== ===STOCK LEVELS==== ===DEMAND=====
PGC  BOB  SUP  AVAIL  ( $100,000 )  UNIT  REQT  RTC  REQT
/ID  UNIT  REQT  ALL  RTC  ONHAND  ONORDER  SAFETY  AMD/100  AMD  AMD

FOR RCW = 1 TO MAX.PGC DO
BEGIN REPORT PRINTING
  FOR COL=1 TO MAX.COL IN GROUPS OF 11
  PRINT 1 LINES WITH A GROUP OF PGC.MAT(ROW,COL) FIELDS
  THUS
  **  **  **  **  **  **  **  **  **  **  **
  LOOP
END "REPORT
IF ADDPGC.OPT =2
  "" THEN have a set of different PGCs so add to get system results
  FOR ROW =1 TO MAX.PGC DO
    FOR COL=1 TO MAX.COL DO
      IF (COL=4) OR (COL=5)
      "" THEN fill rates so weight by demand
        SUM.PGC(COL)=SUM.PGC(COL) +
        (PGC.MAT(ROW,COL) * PGC.MAT(ROW,COL+6))
      ELSE "' just sum values

SUM.PGC(COL) = SUM.PGC(COL) + PGC.MAT(ROW,COL)

ALWAYS
LOOP
LOOP
FOR COL=4 TO 5 DO
SUM.PGC(COL) = SUM.PGC(COL)/SUM.PGC(COL+6)
LOOP
BEGIN REPORT PRINTING
FOR COL=2 TO MAX.COL IN GROUPS OF 10
PRINT 2 LINES WITH A GROUP OF SUM.PGC(COL) FIELDS
THUS

TOTAL ** ** ** ** ** ** ** ** ** **
END "REPORT
ALWAYS
PRINT 7 LINES THUS

KEY: AMD = AVERAGE MONTHLY DEMAND
REQT = REQUISITIONS
ALL = ALL CUSTOMERS (PICS)
RTC = RECRUIT TRAINING CENTERS
BOH = BACKORDERS ON HAND
SUP AVAIL = SUPPLY AVAILABILITY

CLOSE UNIT 18

USE UNIT 1 FOR OUTPUT "print this PGC results to trace information
PRINT 4 LINES THUS

====AVERAGE== ==$REQT== ====STOCK LEVELS===== =====DEMAND=========
PGC BOH SUP AVAIL ONHAND ONORDER SAFETY UNIT REQT RTC
REQT /ID UNIT REQT ALL RTC ($ 100,000) AMD/100 AMD
AMD
BEGIN REPORT PRINTING
FOR COL=1 TO MAX.COL IN GROUPS OF 11
PRINT 1 LINES WITH A GROUP OF PGC.MAT(MAX.PGC,COL) FIELDS
THUS
** ** ** ** ** ** ** ** ** **
END "REPORT
CLOSE UNIT 1

END "routine ADD.ALL.PGCS

ROUTINE CALC.ORDER.QTY GIVEN NSN YIELDING ORDER.QTY
"This routine calculates the order quantity for a NSN in a PGC. The
"quantity equals the difference between the current inventory
"position (onhand + onorder - backorders) and the PTAO
"(peace time acquisition objective)
DEFINE DMD.YEAR, PERIOD, PTAO AS REAL VARIABLES
DEFINE NSN, ORDER.QTY AS INTEGER VARIABLES
**calculate requirements during the next buy period**

\[
\text{sum CT REQ from time of breach out a (ALT + PLT + PCP) period}
\]

\[
\text{PERIOD} = \left(\frac{\text{ALT.DAY} + \text{PLT.DAY(\text{NSN})}}{\text{DPM}}\right) + \text{PCP.MONTH(\text{NSN})}
\]

\[
\text{CALL SUM.FORECAST.OVER.TIME GIVEN \text{NSN AND PERIOD YIELDING PTAO}
}\]

**calculate the safety level in units**

\[
\text{Safety level} = \text{ave. monthly demand} \times \text{Safety level (months)}
\]

\[
\text{Sum next 12 months of forecast demand from time of breach}
\]

\[
\text{PERIOD-12} = \left(\frac{\text{ALT.DAY} + \text{PLT.DAY(\text{NSN})}}{\text{DPM}}\right) + \text{PCP.MONTH(\text{NSN})}
\]

\[
\text{CALL SUM.FORECAST.OVER.TIME GIVEN \text{NSN AND PERIOD YIELDING DMD.YEAR}
}\]

**calculate the order quantity for the \text{NSN**}

\[
\text{IF TRACE18=.TRUE}
\]

\[
\text{PRINT 1 LINE WITH \text{NSN, PTAO, DMD.YEAR, AT.MONTH THUS}
}\]

\[
++++++ + \text{NSN} ** \text{PLT+PCP} ** \text{DMD.YR} **.* \text{AT.MONTH} **
\]

\[
\text{ALWAYS}
\]

\[
\text{PTAO+PTAO + OWRM(\text{NSN}) + ((DMD.YEAR/12)\times\text{SAFETY.MONTH(\text{NSN})})}
\]

\[
\text{order} = \text{PTAO} - \text{inventory position (at time of breach)}
\]

\[
\text{ORDER.QTY=PTAO -(STOCK(\text{NSN}) + ONORDER(\text{NSN}) - UNIT.BO(\text{NSN, TOTAL}))}
\]

\[
\text{IF TRACE18=.TRUE}
\]

\[
\text{PRINT 1 LINE WITH \text{NSN, ORDER.QTY, PTAO,STOCK(\text{NSN}),}
\]

\[
\text{UNIT.BO(\text{NSN, TOTAL}) THUS}
\]

\[
++++ + \text{NSN} ** \text{ORDER} ** \text{PTAO} ** \text{STOCK} ** \text{BO} **
\]

\[
\text{ALWAYS}
\]

\[
\text{IF ORDER.QTY<0}
\]

\[
\text{THEN ROP has changed since beginning of month & do not order}
\]

\[
\text{ORDER.QTY=0}
\]

\[
\text{ALWAYS}
\]

\[
\text{END 'routine CALC.ORDER.QTY'
}\]

**PROCESS COMPUTE.ROP.PCP**

\[
\text{This routine computes PCP in months, ROP for all \text{NSNs every 30 days}
}\]

\[
\text{DEFINE ROP.REVIEW, VIP, \text{NSN AS INTEGER VARIABLES''array indices}
}\]

\[
\text{DEFINE FORECAST.YEAR AS REAL VARIABLE ''POI annual demand over 12 mths}
\]

\[
\text{DEFINE TOT.QFD AS REAL VARIABLE ''replen QFD and (POI+replen) QFD}
\]

\[
\text{DEFINE DVQD AS REAL, 1-DIMENSIONAL ARRAY ''$ value quarterly demand}
\]

\[
T = 2 \times \text{SQRT}(2 \times \text{PROCURE COST} / \text{HOLDING COST})
\]

\[
\text{DEFINE ROP.MONTH AS REAL VARIABLE ''the no. of months the ROP covers}
\]

\[
\text{RESERVE DVQD(*) AS MAX.NSN}
\]

\[
???? \text{NOTE: FOR QFD MIGHT HAVE A PGC MIXTURE OF VIP & NON VIP ITEMS}
\]

\[
???????????????? for each \text{NSN if mixture how to do ?????????????????}
\]

\[
\text{FOR \text{NSN=1 TO MAX.NSN}
\]

\[
\text{ADD VIP.ITEM(\text{NSN}) TO VIP ''no. of VIP items in PGC}
\]

\[
\text{IF VIP = .FALSE ''i.e. no VIP items in PGC}
\]

\[
\text{ROP.REVIEW = .DPM \times 3 '' adjust ROP each quarter}
\]

\[
\text{ELSE}
\]

\[
\text{ROP.REVIEW = .DPM ''adjust ROP every month}
\]

\[
\text{ALWAYS}
\]

\[
\text{UNTIL TIME.V} \geq \text{END.OP.SIMULATION}
\]

\[
\text{DO FOR \text{NSN=1 TO MAX.NSN DO}
\]

\[
2-15
\]
CALL SUM.FORECAST.OVER.TIME (NSN,12.0) YIELDING FORECAST.YEAR
TOT.QFD=TRUNC.F(FORECAST.YEAR/4)
DVQD(0SN)=COST*TOT.QFD
IF DVQD(NSN) <= M1
  'THEN DVQD set for a 36 month procurement cycle
  PCP.MONTH(NSN)=36
ELSE
  IF (DVQD(NSN) > M1) AND (DVQD(NSN) <= M2)
    'THEN between M1 & M2 so use Wilson Lot Size equation
    PROCURE CYCLE (MONTHS)= EOQ / MONTHLY DEMAND
    PCP.MONTH(NSN)=TRUNC.F((3*T)*(DVQD(NSN)**(-0.5)))
  ELSE 'greater than M2 or use 6 month PCP
    PCP.MONTH(NSN)=6
END 'ALWAYS

ALWAYS

Calculate Reorder point quantity by converting time to units
ROP= sum CTREQ over PLT+ALT+(safety level * AVE monthly demand)
ROP.MONTH = (ALT.DAY + PLT.DAY(NSN))/.DPM
CALL SUM.FORECAST.OVER.TIME (NSN,ROP.MONTH) YIELDING
ROP.QTY(NSN)

ROP.QTY(NSN)=ROP.QTY(NSN) + (SAFETY.MONTH(NSN)* (TOT.QFD/3))
+ OWRM(NSN)

LOOP ' 'for NSNs
WAIT ROP.REVIEW DAYS
LOOP'' until end of simulation
END 'process COMPUTE.ROP.PCP

ROUTINE CONFIDENCE INTERVAL
'This routine calculates the covariances for all lags and the
' confidence intervals.
' General covariance formula for the kth lag (Ck):
' \( C_k = \frac{1}{N-k} \sum (X_i - \text{ave} \cdot X)(X_{i+k} - \text{ave} \cdot X) \) : sum is for 1 to N-k
'The one pass formula
'\( C_k = \frac{1}{N-k} \left[ X_i X_{i+k} - (k+N) \text{ave} \cdot X^2 + \text{ave} \cdot X \left( \sum X_{1..k} + \sum X_{n-k+1..N} \right) \right] \)
'where \( X \)
\begin{array}{l}
  \text{PRODUCT.MAT} (NSN, X) \\
  \text{SUM.K.ENS} (NSN, K)
\end{array}

DEFINE NSN, LAG AS INT,VR VARIABLES
DEFINE COVAR, MEAN.VAR AS REAL VARIABLES
DEFINE COVAR.SUM AS REAL, 1-DIMENSIONAL ARRAY
RESERVE COVAR.SUM(*) AS PGC.NUM

'********** COVARIANCE CALCULATIONS AFTER SAMPLING COMPLETED ******
FOR NSN=1 TO PGC.NUM DO
  FOR LAG=1 TO K.LAG DO
    \( C_k = \frac{1}{N-k} \left[ X_i X_{i+k} - (k+N) \text{ave} \cdot X^2 + \text{ave} \cdot X \left( \sum X_{1..k} + \sum X_{n-k+1..N} \right) \right] \)
    COVAR=(1/(N.BLOCKS-LAG)) * (PRODUCT.MAT(NSN, LAG)
    -((LAG+N.BLOCKS)* (AVE.COVAR.DATA(NSN)**2))
    +(SUM.K.ENS (NSN, LAG)*AVE.COVAR.DATA(NSN)))
    IF LAG <= M.COVAR
      THEN add to total covariance for first M lags
      COVAR.SUM(NSN)=COVAR.SUM(NSN)
\[(2 \times ((N.BLOCKS-LAG)/N.BLOCKS) \times COVAR)\]

ALWAYS
IF VAR.COVAR.DATA(NSN) NE 0
    PRINT 1 LINE WITH NSN, LAG, COVAR, COVAR.SUM(NSN),
    COVAR/VAR.COVAR.DATA(NSN) THUS
NSN  ** LAG  ** COVAR  **.*  COVAR.SUM  **.*  CORR  .****
    ALWAYS
LOOP 'for Lags
    PRINT 1 LINE WITH NSN, COVAR.SUM(NSN), VAR.COVAR.DATA(NSN),
    AVE.COVAR.DATA(NSN) THUS
SUMMARY NSN  ** COVAR/N2  **.*  VAR  **.*  MEAN  **.*
    LOOP 'for NSNs
PRINT 5 LINES THUS

====================================================================
====================================================================

'********** CONFIDENCE INTERVAL  **********
PRINT 2 LINES WITH M.COVAR, COVAR.INTERVAL, N.BLOCKS,
(TIME.V-WARMUP.PERIOD)/360 THUS
*** STATS FOR RUN: M.LAGS ** INTVL ** BLOCKS ** YRs  **.*
NSN MEAN VAR 2COVAR/N MEAN.VAR  C.I.95% %C.I./MEAN
FOR NSN=1 TO PGC.NUM DO
    MEAN.VAR= (VAR.COVAR.DATA(NSN) + COVAR.SUM(NSN))/N.BLOCKS
    IF MEAN.VAR<0
        RETURN
    ALWAYS
    ' confid. interval of mean = mean +/- z.05 * stand. dev. of mean
    CONF.INTV=1.96 * SQRT.F(MEAN.VAR) '95% confidence interval
    PRINT 1 LINE WITH NSN, AVE.COVAR.DATA(NSN), VAR.COVAR.DATA(NSN),
    COVAR.SUM(NSN), MEAN.VAR, CONF.INTV,
    100*CONF.INTV/AVE.COVAR.DATA(NSN) THUS
    **  **  **.*  **.*  **.*  **  **.*
    LOOP
END 'ROUTINE CONFIDENCE INTERVAL

PROCESS COVAR.SAMPLING
' This process samples and updates required variables to estimate
' the covariances for each NSN and the PGC. (see Gross p418.)
DEFINE BLOCK, LAG, NSN, ITEM, HOLD.X1, POINT.X, I, NUM, REQBO.OPT
AS INTEGER VARIABLES
DEFINE SUM.PGC AS REAL VARIABLES
' ***** SET TRUE FOR REQUISITION, FALSE FOR UNIT BO COVARIANCE ***
REQBO.OPT= .FALSE

WAIT WARMUP.PERIOD DAYS
' Insert first K data points into set & sum values

2-17
FOR ITEM = 1 TO K.LAG DO
WAIT COVAR INTERVAL DAYS
FOR NSN = 1 TO PGC NUM DO
CREATE A COVAR MEMBER
IF NSN = PGC NUM
' THEN do PGC
SUM PGC = 0
FOR NUM = 1 TO MAX NSN DO
IF REQBO OPT = .TRUE 'do requisition BOs
ADD REQ BO (NUM,.TOTAL) TO SUM PGC
ELSE 'do unit BO for covar and C.I.
ADD UNIT BO (NUM,.TOTAL) TO SUM PGC
ALWAYS
LOOP
COVAR DATA (NSN) = SUM PGC
ELSE 'do NSN
COVAR DATA (NSN) = REQ BO (NSN,.TOTAL)
PRINT 1 LINE WITH NSN, TIME V, REQ BO (NSN,.TOTAL) THUS
NSN ** TIME V ** REQ BO **
COVAR DATA (NSN) = CREQ MAT (NSN, ITEM,.TOTAL)
ALWAYS
DATA POINT = COVAR DATA (NSN)
FILE COVAR MEMBER IN COVAR SET (NSN)
Add k values to each lag to handle 1st k items not in sum
FOR LAG = ITEM TO K.LAG
SUM K ENDS (NSN, LAG) = SUM K ENDS (NSN, LAG) + COVAR DATA (NSN)
LOOP
LOOP
'********** end: INITIAL SET UP *************
IF TRACE15 = .TRUE
FOR NSN = 1 TO PGC NUM
FOR LAG = 1 TO K LAG
PRINT 1 LINE WITH NSN, LAG, SUM K ENDS (NSN, LAG) THUS
AFTER INITIAL NSN ** LAG ** SUM OF 1ST K VALUES **
ALWAYS
'********** start: MIDDLE running phase of program
BLOCK = K LAG + 1
UNTIL TIME V = END OF SIMULATION DO
'** UNTIL (BLOCK = W BLOCKS + 1) DO
WAIT COVAR INTERVAL DAYS
FOR NSN = 1 TO PGC NUM DO
REMOVE FIRST POINT X FROM THE COVAR SET (NSN)
HOLD X1 = DATA POINT (POINT X)
IF NSN = PGC NUM
' THEN do PGC
SUM PGC = 0
FOR NUM = 1 TO MAX NSN DO
IF REQBO OPT = .TRUE 'do requisition BOs
ADD REQ BO (NUM,.TOTAL) TO SUM PGC
ELSE 'do unit BO for covar and C.I.
ADD UNIT BO (NUM,.TOTAL) TO SUM PGC
ALWAYS
LOOP
COVAR DATA (NSN) = SUM PGC

2-18
ELSE 'do NSN
  COVAR.DATA(NSN) = REQ.BO(NSN,.TOTAL)
  PRINT 1 LINE WITH NSN,TIME,V,REQ.BO(NSN,.TOTAL),BLOCK
THUS
  NSN ** TIME.V ** REQ BO ** BLOCK **
ALWAYS
  DATA.POINT(POINT.X) = COVAR.DATA(NSN)
  FILE POINT.X IN COVAR.SET(NSN)
  LAG=1
  FOR EACH ITEM IN THE COVAR.SET(NSN) DO
    ADD (HOLD.X1 * DATA.POINT(ITEM)) TO PRODUCT.MAT(NSN,LAG)
    ADD 1 TO LAG
  LOOP
  IF TRACE15=.TRUE
    LAG=1
    FOR EACH ITEM IN COVAR.SET(NSN) DO
      PRINT 1 LINE WITH NSN, LAG, HOLD.X1, DATA.POINT(ITEM),
          PRODUCT.MAT(NSN,LAG) THUS
      NSN ** LAG ** X1 ** X.LAG ** CUM PROD **
        ADD 1 TO LAG
    LOOP
    ALWAYS
    LOOP 'next NSN
    BLOCK=BLOCK+1
  LOOP 'until
' ********** end: MIDDLE running phase **********
'********* Add i=k+1 to n values
  FOR NSN=1 TO PGC.NUM DO
    I=1
    FOR EACH ITEM IN THE COVAR.SET(NSN) IN REVERSE ORDER DO
      FOR LAG=1 TO K.LAG DO
        ADD DATA.POINT(ITEM) TO SUM.K.ENDS(NSN,LAG)
      LOOP
      ADD 1 TO I
    LOOP
  LOOP
  IF TRACE15=.TRUE
    FOR NSN=1 TO PGC.NUM
      FOR LAG=1 to K.LAG
        PRINT 1 LINE WITH NSN, LAG, SUM.K.ENDS(NSN,LAG) THUS
        AFTER FINAL NSN ** LAG ** SUM OF LAST K VALUES **
        ALWAYS
'************ start: FINAL (Xl,Xi+k) product for remaining K.lag items
  FOR NSN=1 TO PGC.NUM DO
    FOR I=1 TO K.LAG-1 DO
      REMOVE FIRST POINT.X FROM THE COVAR.SET(NSN)
      HOLD.X1=DATA.POINT(POINT.X)
      LAG=1
      FOR EACH ITEM IN THE COVAR.SET(NSN) DO
        ADD (HOLD.X1 * DATA.POINT(ITEM)) TO PRODUCT.MAT(NSN,LAG)
        ADD 1 TO LAG

2-19
LOOP
IF TRACE15=.TRUE
LAG=1
FOR EACH ITEM IN COVAR.SET(NSN) DO
PRINT 1 LINE WITH NSN, LAG, HOLD.X1, DATA.POINT(ITEM),
PRODUCT.MAT(NSN,LAG) THUS
END NSN ** LAG ** X1 ** X.LAG ** CUM PROD **.
ADD 1 TO LAG
LOOP
ALWAYS
LOOP
LOOP 'do next NSN
**********
IF (AVE.COVAR.DATA(PGC.NUM) <> 0) AND (K.LAG<N.BLOCKS)
'THEN BO condition occurred & have enough samples to calculate C.I.
CALL CONFIDENCE.INTERVAL
ALWAYS
END 'Process Covar.Sampling

PROCESS DEMAND.GENERATOR
'This routine generates demands and requisitions for a given NSN
DEFINE REQ.SIZE.NOW AS INTEGER VARIABLES
DEFINE DEMAND.COUNT, TIME.OF.REQ AS REAL VARIABLES.

' USE UNIT 6 FOR OUTPUT

REQ.SIZE.NOW=INT.F(REQUISITION.RATIO.F*ARS(NSN.D))
WAIT DEMAND.INTERVAL DAYS
UNTIL TIME.V > END.OF.SIMULATION DO
' PRINT 1 LINE WITH NSN.D,TIME.V,DDR(NSN.D),REQ.SIZE(NSN.D,.TOTAL) THUS
' NSN.D * TIME.V ** DDR **.******* REQ SIZE **
IF (POISSON.OPT=.TRUE)
'THEN only simulate if both monthly and daily demand needed
SIM.DDR(NSN.D)=POISSON.P(DDR(NSN.D),1)
ELSE
SIM.DDR(NSN.D)= DDR(NSN.D) 'either CTREQ or MAPE adjusted
ALWAYS
DEMAND.COUNT=DEMAND.COUNT + SIM.DDR(NSN.D)
WHILE ((DEMAND.COUNT >= REQ.SIZE.NOW) AND (DOREQ.OPT=.TRUE)) DO
'loop for requisitions and recruit center info.
REQ.SIZE(NSN.D,.TOTAL)=REQ.SIZE.NOW -
REQ.INTERVAL(NSN.D,.TOTAL)=TIME.V - TIME.OF.REQ
TIME.OF.REQ=TIME.V
IF (REQ.SIZE.NOW >= RECRUIT.SIZE.CUTOFF(NSN.D))
'THEN update requisition and unit demands for recruit centers
REQ.INTERVAL(NSN.D,.RECRUIT)=REQ.INTERVAL(NSN.D,.TOTAL)
REQ.SIZE(NSN.D,.RECRUIT)=REQ.SIZE(NSN.D,.TOTAL)
ALWAYS
IF TRACE1=.TRUE
IF (REQ.SIZE.NOW >= RECRUIT.SIZE.CUTOFF(NSN.D))
PRINT 1 LINE THUS
==== A RECRUIT REQUISITION ABOVE CUTOFF ====
ALWAYS
PRINT 1 LINE WITH NSN.D, TIME.V, DEMAND.COUNT,
REQ.SIZE(NSN.D,.TOTAL), REQ.INTERVAL(NSN.D,.TOTAL) THUS
NSN * TIME ** DEM CNT **.* REQ.SIZE ** REQ.INTRVL **.*
ALWAYS
CALL REQ.TO.INVENTORY GIVEN NSN.D
DEMAND.COUNT=DEMAND.COUNT-REQ.SIZE(NSN.D,.TOTAL)
REQ.SIZE.NOW=TRUNC.P(REQUISITION.RATIO.P*ARS(NSN.D) + .9999)
LOOP 'while
IF (DOREQ.OPT=.FALSE)
'THEN each requisition equals daily demand
REQ.SIZE(NSN.D,.TOTAL)=SIM.DDR(NSN.D)
IF TRACE1=.TRUE
PRINT 1 LINE WITH NSN.D, TIME.V, REQ.SIZE(NSN.D,.TOTAL),
REQ.INTERVAL(NSN.D,.TOTAL) THUS
NOREQ NSN * DAY ** DDR/SIZE **.* REQ.INT **.*
ALWAYS
CALL REQ.TO.INVENTORY GIVEN NSN.D
ALWAYS
WAIT DEMAND.INTERVAL DAYS
LOOP 'until
END 'process DEMAND.GENERATOR

ROUTINE DISTRIBUTION.DATA
'This routine initializes random variables distributions for the
' PLT.DAY.DELAY (CLIN report), REQUISITION.RATIO.P(USIMS <5
' requisition distribution), and DEMAND.MAPE.P from Orchowsky's POI
' report pg. 6
' USE THE BUFFER FOR OUTPUT

' the PLT distribution gives the number of days early or late of a
' order. Format is probability then value (i.e. P(x), x)

'************ PLT CLIN DISTRIBUTION ***********************
WRITE AS */" 0.0 -20 0.10 0 0.7352 30 "
WRITE AS " 0.8116 90 0.8757 180 1.0000 360 * "

' NOT USED NOW

'************ PLT ANALYSIS REPORT DISTRIBUTION ***************
'WRITE AS */" 0.0 -360 0.02 -330 0.02 -300 0.02 -270 "
'WRITE AS " 0.04 -240 0.05 -210 0.07 -180 0.09 -150 "
'WRITE AS " 0.11 -120 0.15 -90 0.21 -60 0.31 -30 "
'WRITE AS " 0.49 0 0.62 30 0.73 60 0.79 90 "
'WRITE AS " 0.86 120 0.90 150 0.91 180 0.92 210 "
'WRITE AS " 0.94 240 0.96 270 0.96 300 0.97 330 "
'WRITE AS " 1.00 360 * "

READ PLT.DAY.DELAY.F USING THE BUFFER

' 1 +/- MAPE CUM probability density function F(x), x
WRITE AS */" 0.0 0.0 .00000001 .01 0.0433 0.26 "
WRITE AS " 0.1371 0.51 .2673 0.76 0.2970 1.00 "

2-21
WRITE AS " 0.3443 1.24 0.5511 1.49 0.7030 1.74 */
WRITE AS " 0.7690 1.39 0.9590 11.0 1.0000 13.0 */

READ DEMAND.MAPE.F USING THE BUFFER

' Cumulative probability function F(X), X
' random variable REQUISITION.RATIO.F from USIMS DPSC w/ ARS > 5 pg. W-8
WRITE AS /* " 0.0 0.0 0.169 0.1 0.307 0.2 0.482 0.4 0.612 0.6 0.688 0.8 */
WRITE AS " 0.753 1.0 0.805 1.25 */
WRITE AS " 0.844 1.5 0.872 1.75 0.893 2.0 0.922 2.5 0.941 3.0 0.963 4.0 0.974 5.0 */
WRITE AS " 0.98 6.0 0.988 8.0 0.992 10.0 33.9125 */
READ REQUISITION.RATIO.F USING THE BUFFER

USE UNIT 1 FOR OUTPUT 'switch back to output file

IF TRACE14=.TRUE
' THEN (can't set trace so must disable directly
DEFINE I AS INTEGER VARIABLE
' send to printer
' USE UNIT 2 FOR OUTPUT
DEFINE HOLD.PLT AS REAL VARIABLES
FIRST.DELIVERY=100
FOR I= 1 TO 1000 DO
  HOLD.PLT= FIRST.DELIVERY + PLT.OPT * PLT.DAY.DELAY.F
PRINT 1 LINE WITH I, HOLD.PLT
NUM ** PLT DELAY **
IF HOLD.PLT<10
' THEN order will arrive before placed so set to 10 days
SIM.PLT.DAY=10
ELSE
SIM.PLT.DAY = HOLD.PLT
ALWAYS
LOOP
PRINT 1 LINE WITH AVE.SIM.PLT
AVERAGE SIM PLT **,**
PRINT 2 LINES
CUMMULATIVE DISTRIBUTION FOR THE MAPE DEMAND FUNCTION
INDEX VALUE CUM PROB &
FOR EACH RANDOM.E IN PLT.DAY.DELAY.F,
PRINT 1 LINE WITH RVALUE.A(RANDOM.E), 100*PROB.A(RANDOM.E) THUS **,** **,** */
STOP
ALWAYS
END 'routine DISTRIBUTION.DATA

PROCESS GET.PLOT.DATA
'this process gets or calculates several plotted variables for
' dynamic net stock plot & static plotted histogram graphics
DEFINE NSN, SUM.HOLD AS INTEGER VARIABLES

2-22
WAIT WARMUP.PERIOD DAYS
UNTIL TIME.V >= END.OF.SIMULATION DO
  WAIT PLOT.INTERVAL DAYS
  ***** calculating NET.STOCK ********
  IF TRACE10 = .TRUE
    THEN do the 1st 3 NSNs plot of their net stock
      FOR NSN=1 TO 3
        NET.STOCK(NSN)=(STOCK(NSN) - UNIT.BO(NSN,.TOTAL))/1000
        ALWAYS
  Do PGC.NET.STOCK always for histogram
  SUM.HOLD=0
  FOR NSN=1 TO MAX.NSN
    SUM.HOLD= SUM.HOLD +
    (STOCK(NSN) - UNIT.BO(NSN,.TOTAL))
    PGC.NET.STOCK = SUM.HOLD/PLOT.YSCALE
  LOOP
END 'process GET.PLOT.DATA

ROUTINE GRAPH.INITIALIZE
' 'This routine initializes graphics at the start of program: shows
' ' the pie charts, determine histogram intervals (.5 of safety level)
' ' and displays the dynamic traces
DEFINE DEVICE.ID AS POINTER VARIABLE
DEFINE NSN AS INTEGER VARIABLE

' ***** DYNAMIC GRAPHICS INITIALIZATION ************

' ***** PIE CHARTS ***********************
  IF TRACE13=.TRUE
    SHOW EBO.PIE WITH "EBOPIE.GRF"
    SHOW DEMAND.PIE WITH "DEMPIE.GRF"
    RESERVE EBO.PIE(*), DEMAND.PIE(*) AS 3
  ALWAYS

' ********** HISTOGRAMS **********
  FOR NSN = 1 TO MAX.NSN DO
    PGC.SL.STOCK = PGC.SL.STOCK +
    (AVE.FORECAST(NSN)*SAFETY.MONTH(NSN)) + OWRM(NSN)
  LOOP

  IF TRACE9=.TRUE
    'then turn on histogram prints at end
    'histogram is 3 SL Intervals long, 1 negative, 2 positive
    HIS.BO.3=(PGC.SL.STOCK/2)/PLOT.YSCALE 'intvl=.5 SL, scale stock
    HIS.BO.1=-(2 * HIS.BO.3) 'i.e. -PGC.SL.STOCK
    HIS.BO.2= 6 * HIS.BO.3 'i.e. + 2 * PGC.SL.STOCK
    SHOW HISTOGRAM HIST.PGC.PLOT WITH "HISTPGC.GRF"
    SHOW HISTOGRAM HIST.EBO.PLOT (1),HIST.EBO.PLOT(2),
                  HIST.EBO.PLOT(3)
    WITH "EBOHIST.GRF"

2-23
ALWAYS

''******** DYNAMIC TRACE OF NET STOCK LEVELS FOR 3 NSNs OR PGC ********
IF (TRACE10=.TRUE) OR (TRACE20=.TRUE)
' THEN
' set virtual terminal
CALL DEVINIT.R("VT,GRAPHIC") YIELDING DEVICE.ID
OPEN 7 FOR INPUT, DEVICE=DEVICE.ID
OPEN 8 FOR OUTPUT, DEVICE=DEVICE.ID
USE 8 FOR GRAPHIC OUTPUT
IF TRACE10=.TRUE
' THEN display 1st 3 NSNs
DISPLAY NET.STOCK(1), NET.STOCK(2), NET.STOCK(3)
WITH "NETSTOCK.GRF"
ELSE 'display PGC net stock
DISPLAY PGC.NET.STOCK WITH "PGCSTOCK.GRF"
LET VXFORM.V = 5
CALL SETWORLD.R (0,79,0, 23)
CALL MXRESET.R (0)
CALL MXLATE.R (40,0) 'X,Y coordinates position
CALL TEXTANGLE.R (0) 'angle of the text from 0 to 3600
WRITE PLOT.YSCALE AS "STOCK SCALING FACTOR =", D(9,2), / USING 8
CALL GUPDATE.R
ALWAYS
ALWAYS

END ''GRAPH.INITIALIZE

ROUTINE INPUT.MPT011.DATA
''This routine Reads management policy table 11 and gets the minimum
' procurement cycle, PGC delivery percents for all delivery
' increments, 1 of 4 methods of delivery, PGC first delivery in days.

DEFINE TEST.TEXT, TEST2 AS TEXT VARIABLE
DEFINE I, PGC.NUM, MONTH AS INTEGER VARIABLE
DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE PGC.PERCENT AS REAL VARIABLE

USE UNIT 11 FOR INPUT
'' USE 6 FOR OUTPUT
EOF.V=1

'' **** PHASED DELIVERY SET UP ********
MAX.DELIVERIES=12
RESERVE DELIVERY.PERCENT(*) AS MAX.DELIVERIES

UNTIL PGC.NUM = TARGET.PGC DO ''loop to find PGC target number
TEST.TEXT="NEW PGC"
UNTIL TEST.TEXT="ROUP" DO '' loop to find GROUP label
START NEW INPUT RECORD
READ TEST.EOF ''
IF ((TEST.EOF<>26) AND (EOF.V<>2))
' THEN look for GROUP in file to find PGC NUM
READ TEST.TEXT
ELSE ' at end of file without finding PGC's MPT 011 file
WRITE AS "### ERROR: TARGET PGC MPT011 FILE NOT FOUND ",
/ USING 6
STOP
REGARDLESS
LOOP
 ' have found the GROUP label now read PGC.NUM
START NEW INPUT RECORD
READ PGC.NUM, I, MIN.PC, TEST.TEXT, TEST2
LOOP
FOR MONTH = 1 TO MAX.DELIVERIES
READ DELIVERY.PERCENT(MONTH)
MONTH=1
WHILE (((MONTH<= MAX.DELIVERIES) AND (DELIVERY.PERCENT(MONTH) > 0))
DO ' no. incremental deliveries
   DELIVERY.PERCENT(MONTH) = DELIVERY.PERCENT(MONTH)/10 ' make a %
   PGC.PERCENT = PGC.PERCENT + DELIVERY.PERCENT(MONTH)
   MONTH=MONTH + 1
LOOP
MAX.DELIVERIES=MONTH - 1
IF (PGC.PERCENT < 99.99) OR (PGC.PERCENT > 100.01)
 ' THEN
WRITE AS "### ERROR: PGC DELIVERY PERCENT NOT EQUAL TO 100",
/ USING 6
STOP
REGARDLESS
START NEW INPUT RECORD
FOR I=1 TO 3
READ TEST2
READ DELIVERY.OPT
FOR I=1 TO 4
READ TEST2
READ FIRST.DELIVERY
FOR I=1 TO 3
READ TEST2
READ X.PERCENT, Z.PERCENT
LINES.V=0
PRINT 10 LINES WITH RUN.ID THUS

"THE DETAIL TRACE OUTPUT REPORT "
(FILE: DLAOUT.DAT)" (ID NUMBER OF RUN **)

INPUT DATA

PRINT 4 LINES WITH PGC.NUM, DELIVERY.OPT, FIRST.DELIVERY, X.PERCENT,
Z.PERCENT, MIN.PC THUS

2-25
ROUTINE INPUT.SSCP.DATA

This routine reads the required input data to run the simulation
original captured from the Special Supply Control File Report via
a SIMSCRIPT program in directory DLA DATA. The routine finds the
desired PGC number, and reads in the data into the appropriate
variables. If the PGC number is not found the program prints
error message and stops

DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE COL, NSN AS INTEGER VARIABLE
DEFINE TEST.TEXT AS TEXT VARIABLE
USE UNIT 4 FOR INPUT 'C:\SIM\DLA\SSCP SIM.DAT/MOD
EOF.V=1

******** Find target PGC's beginning of data input ****

UNTIL PGC.NO = TARGET.PGC DO ''loop to find PGC target number
TEST.TEXT="NEW PGC"
UNTIL TEST.TEXT="ROC.GR.CD" DO '' loop to find PROC.GR.CD label
START NEW INPUT RECORD
READ TEST.EOF
IF ((TEST.EOF<>26) AND (EOF.V<>2))
  THEN look for GROUP in file to find PGC NUM
  READ TEST.TEXT
ELSE '' at end of file without finding PGC in MPT 011 file
  WRITE AS "### ERROR: TARGET PGC NOT IN SSCF REPORT FILE",
  / USING 6
  STOP

REGARDLESS
LOOP
' have found the GROUP label now read PGC.NO
READ PGC.NO, TEST.TEXT, MAX.NSN
' PRINT 1 LINE WITH PGC.NO, TEST.TEXT, MAX.NSN THUS
' PGC NO. ** TEXT ******** MAX.NSN **
LOOP

******** Start reading PGC related data ***************

CREATE EVERY NSN.ATTRIBUTES(MAX.NSN)
READ PGC.NAME AS /,/,B 1,T 20
READ FSC, ICC, ALT.DAY, COST, MAX.MONTH
"AS /,/,B 1,T 17, B 22,1 7, B 29,T 3, B 34,1 6, B 41, D(10,2), B 57, I 5

SKIP 2 RECORDS

" ******* Read NSN specific data ***********************
FOR NSN = 1 TO MAX.NSN
  READ NSN, NSN.NO(NSN), PLT.DAY(NSN), VIP.ITEM(NSN),
  SAFETY.MONTH(NSN), QFD(NSN)
SKIP 3 RECORDS

FOR NSN = 1 TO MAX.NSN
  READ NSN, MAD(NSN), OWRM(NSN), ALPHA(NSN), ARS(NSN),
  PER_RTC.DEMAND(NSN)
IF ICC="P"
  'THEN
  "***** Read C&T requirements matrix ***************
  'for calculating order.QTY, the CTREQ.MAT has to have enough
  'future months of data for the maximun of PLT, ALT, & PCP
  CTREQ.PERIOD=12 'no. of months before CTREQ mat is shifted & updated
  MAX.CTREQ.DIM=CTREQ.PERIOD + 24 + 6 + 36 'PLT=24, ALT=6, PCP=36
  RESERVE CTREQ.MAT(*,*,*)) AS MAX.NSN BY (MAX.CTREQ.DIM) BY 1 'or 3

FOR NSN = 1 TO MAX.NSN DO
  SKIP 3 RECORDS
  FOR COL= 1 TO MAX.MONTH
    READ CTREQ.MAT(NSN,COL,.TOTAL)
  LOOP
ALWAYS
CLOSE UNIT 4
END "routine INPUT.SSCF.DATA

ROUTINE INPUT.VSL.DATA
"This routine is called if user specifies the VSL option as true
"question (4). It searches the "VSL.DAT" file for the PGC number
"and overrides the fix safety level values from the SSCF with the
"VSL values in months in the file

DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE TEST.TEXT AS TEXT VARIABLE
DEFINE NSN, I, PGC.NUM AS INTEGER VARIABLE

OPEN UNIT 12 FOR INPUT, FILE NAME IS "C:\SIM\DLA\VSL.DAT"
USE UNIT 12 FOR INPUT
EOF.V=1

UNTIL PGC.NUM = TARGET.PGC DO 'loop to find PGC target number
  TEST.TEXT="NEW PGC"
UNTIL TEST.TEXT="PGC" DO 'loop to find GROUP label

2-27
START NEW INPUT RECORD
READ TEST.EOF ''
IF ((TEST.EOF<>26) AND (EOF.V<>2))
  'THEN look for PGC in file to find PGC NUM
  READ TEST.TEXT
ELSE '' at end of file without finding PGC's MPT 011 file
  WRITE AS "### ERROR:TARGET PGC IN VSL.DAT FILE NOT FOUND ", / USING 6
STOP
REGARDLESS
LOOP
  have found the PGC label now read PGC.NUM
  READ PGC.NUM
LOOP
SKIP 2 INPUT RECORDS
FOR NSN = 1 TO MAX.NSN
  READ I, SAFETY.MONTH(NSN), TEST.TEXT
  I=0
CLOSE UNIT 12
END ''routine INPUT.VSL.DATA

ROUTINE LAYINTO.MATRIX GIVEN ITEM
  This routine develops the X, Y, Z amounts of delivery for all delivery
  months. Example: It takes the Z item target of 100% in 6th month
  and the total percent SUM of Z items for the PGC (e.g., 20%). It
  then makes sure those targets times the SUM will not exceed the
  PGC DELIVERY.PERCENT (e.g., 15%) by month. Since it does exceed the
  PGC.DELIVERY.PERCENT (20>15), it takes the overflow (5%) and moves
  it into the previous month (the 5th month). The 5% now becomes the
  target for the 5th month and the cycle repeats.

DEFINE ITEM AS INTEGER VARIABLE ''whether an X, Y, or Z items vector
DEFINE DELIVER AS REAL VARIABLE '' the percent delivered this month
DEFINE OVERFLOW AS REAL VARIABLE ''percent that overflow to next month
DEFINE MONTH AS INTEGER VARIABLE
DEFINE ITEM.SUM AS REAL VARIABLE '' the % of the X,Y, or Z in PGC

ITEM.SUM=XYZ.SUM(ITEM)
MONTH=MAX.DELIVERIES
While (ITEM.SUM > 0) AND (MONTH > 0) Do 
  DELIVER= XYZ.SUM(ITEM) * (XYZ.MATRIX(ITEM,MONTH)/100)
  DELIVERY.PERCENT(MONTH)=DELIVERY.PERCENT(MONTH) - DELIVER
  IF (DELIVERY.PERCENT(MONTH) >= 0) OR (MONTH=1)
    'THEN this months delivered can fit & no overflow to next month
    XYZ.MATRIX(ITEM,MONTH)= DELIVER
    ITEM.SUM=ITEM.SUM - DELIVER
  ELSE 'can fit all in this month so overflow to next month
   .enter deliver - overflow into XYZ matrix
    OVERFLOW = ABS.F(DELIVERY.PERCENT(MONTH))
    XYZ.MATRIX(ITEM,MONTH)= DELIVER - OVERFLOW

2-28
ITEM.SUM = ITEM.SUM - (DELIVER - OVERFLOW)
XYZ.MATRIX(ITEM, MONTH-1) = XYZ.MATRIX(ITEM, MONTH-1) + 
(100*OVERFLOW/XYZ.SUM(ITEM))
DELIVERY.PERCENT(MONTH) = 0
ALWAYS
IF TRACE22 = .TRUE
PRINT 1 LINE WITH ITEM, MONTH, DELIVER, ITEM.SUM, OVERFLOW
THUS
LAY IN ITEM ** MONTH ** DELIVER **.* ITEM SUM **.* OVL **
ALWAYS
MONTH = MONTH - 1
LOOP

END ' 'LAYINTO.MATRIX

ROUTINE MATRIX.DELIVERY.SCHEDULE
' 'This routine lays in the different X, Y, & Z percent deliveries per
' 'month vector into the XYZ.MATRIX for each of the 3 delivery
' 'methods. Method 1 does not use the X, Y, Z percents but a sort
' 'in routine METHOD1.SCHEDULE
DEFINE ROW, X, Y, Z, MONTH AS INTEGER VARIABLES
RESERVE XYZ.MATRIX(*,*) AS 3 BY MAX.DELIVERIES
X=1
Y=2
Z=3
XYZ.MATRIX(Z, MAX.DELIVERIES) = 100
SELECT CASE DELIVERY.OPT
CASE 1 ' ** DELIVERY METHOD 1 **************
CALL METHOD1.SCHEDULE

CASE 2 ' ** DELIVERY METHOD 2 **************
CALL LAYINTO.MATRIX(Z)
' 'fill in Y percents over last 1/2 of months if odd no. round up
' 'i.e., put Y's in partial month
FOR MONTH BACK FROM MAX.DELIVERIES TO
TRUNC.F(MAX.DELIVERIES/2) + 1
XYZ.MATRIX(Y, MONTH) = (100/
(MAX.DELIVERIES-TRUNC.F(MAX.DELIVERIES/2)))
CALL LAYINTO.MATRIX(Y)
' 'make X item vector equal to remaining PGC delivery percents
FOR MONTH = 1 TO MAX.DELIVERIES UNLESS XYZ.SUM(X)=0 DO
XYZ.MATRIX(X, MONTH) = DELIVERY.PERCENT(MONTH)
LOOP

CASE 3 ' ** DELIVERY METHOD 3 **************
' 'lay in Z in equal percents over only the last 2/3s of schedule
' 'for odd delivery months round up
FOR MONTH BACK FROM MAX.DELIVERIES TO
TRUNC.F(MAX.DELIVERIES/3 + 1)
XYZ.MATRIX(Z, MONTH) = (100/

2-29
(MAX.DELIVERIES - TRUNC.F(MAX.DELIVERIES/3)))
CALL LAYINTO.MATRIX(Z)
''
make X & Y item vector equal to remaining PGC delivery percents
FOR ROW = X TO Y
FOR MONTH = 1 TO MAX.DELIVERIES
XYZ.MATRIX(ROW,MONTH) = XYZ.SUM(ROW) * DELIVERY.PERCENT(MONTH)
/ (100 - XYZ.SUM(Z)) ''% of order remaining
ENDSELECT
'' *** calculate item type percents (X,Y, or Z) after lay in rebalancing
IF TRACE22 = .TRUE
PRINT 1 LINE
====== TRACE WITH PERCENT OF PGC DELIVERED EACH MONTH ======
CALL PRINT.DELIVERY.MATRIX
ALWAYS
FOR ROW = 1 TO 3 DO
IF XYZ.SUM(ROW) = 0
'' THEN fix so no divide by zero errors
XYZ.SUM(ROW) = .MINVAL
ALWAYS
LOOP
FOR ROW = 1 TO 3
FOR MONTH = 1 TO MAX.DELIVERIES
XYZ.MATRIX(ROW,MONTH) = XYZ.MATRIX(ROW,MONTH) / XYZ.SUM(ROW)
CALL PRINT.DELIVERY.MATRIX
END ''routine MATRIX.DELIVERY.SCHEDULE

ROUTINE METHOD1.SCHEDULE
'' For method 1, the delivery is made in a clump, not spread over several
'' months like other methods. So take first months delivery percent and
'' bring in as many NSNs as month can handle (ex. 10%). Set the
'' XYZ.MONTH value at 1 to mean bring entire order in first month. If
'' an NSN can have 50% of its order brought in in the current month do
'' so, if not have NSN be brought in next month.

DEFINE SORTED AS A INTEGER, 1-DIMENSIONAL ARRAY
RESERVE SORTED(*) AS MAX.NSN

DEFINE DELIVERY.MONTH, MONTH.PERCENT, MAX.PERCENT, AT.NSN,
SUM.PERCENT AS REAL VARIABLES
DEFINE ROW, NSN, DONE AS INTEGER VARIABLES
DONE=1
DELIVERY.MONTH = 1
MONTH.PERCENT = DELIVERY.PERCENT(DELIVERY.MONTH)
FOR ROW=1 TO MAX.NSN DO '"for each NSN determine PLT
MAX.PERCENT=0
FOR NSN=1 TO MAX.NSN DO '"find NSN w/ biggest PCP*AMP percent
IF (((XYZ.MONTH(NSN) > MAX.PERCENT) AND (SORTED(NSN)<>'DONE'))
"'then this NSN has largest PCP.PERCENT so switch
MAX.PERCENT=XYZ.MONTH(NSN)
AT.NSN=NSN
ALWAYS
LOOP
"'have just found next NSN w/ biggest PCP*AMP not already done
SORTED(AT.NSN)=DONE
"'now determine when NSN will be delivered in clump i.e. its PLT
IF (SUM.PERCENT + (MAX.PERCENT * 0.5)) <= MONTH.PERCENT
"'then bring in this month
XYZ.MONTH(AT.NSN) = DELIVERY.MONTH
ELSE 'bring in next month
XYZ.MONTH(AT.NSN) = DELIVERY.MONTH + 1
ALWAYS
SUM.PERCENT=SUM.PERCENT + MAX.PERCENT
WHILE (((SUM.PERCENT >= MONTH.PERCENT) AND
(DELIVERY.MONTH < MAX.DELIVERIES)) DO 'updates for next month
DELIVERY.MONTH=DELIVERY.MONTH + 1
MONTH.PERCENT = MONTH.PERCENT + DELIVERY.PERCENT(DELIVERY.MONTH)
LOOP
IF TRACE22=.TRUE
PRINT 1 LINE WITH ROW, AT.NSN, MAX.PERCENT, XYZ.MONTH(AT.NSN),
DELIVERY.MONTH, SUM.PERCENT, MONTH.PERCENT
THUS
PASS ** AT ** MAX% **.* XYZ.MONTH ** DEL.MTH ** SUM% **.* MTH% **
ALWAYS
LOOP '"do next NSN & find delivery month

END '"routine METHOD1.SCHEDULE

ROUTINE OPTIONAL.ASSUMPTIONS
"'This routine lets the user override the standard assumptions, options
"'or traces settings found in PGC.INITIALIZE & SET.OPTIONS and lets
"'the user specify there own by editing the file ASSUMP.MOD & entering
"'1 in the user query (10), select alternate Assumption file.

DEFINE TEST.TEXT AS TEXT VARIABLE

USE UNIT 3 FOR INPUT '"ASSUMP.DAT
UNTIL TEST.TEXT="T" DO
READ TEST.TEXT
START NEW INPUT RECORD
LOOP
READ T, M1, M2
UNTIL TEST.TEXT="TRACES" DO
START NEW INPUT RECORD
READ TEST.TEXT

2-31
LOOP
START NEW INPUT RECORD
READ TRACE17
SKIP 3 INPUT RECORD
READ DOREQ.OPT
CLOSE UNIT 3

PRINT 1 LINE WITH TRACE17 AND DOREQ.OPT THUS
TRACE 17 IS ** DO REQ OPTIONS **
END ''routine OPTIONAL.ASSUMPTIONS

ROUTINE PGC.INITIALIZE
'' This routine intializes some of the basic PGC variables such as
'' time intervals between processes, PC variables, mean FORECASTs,
'' and covariance information.

DEFINE MONTH, TYPE, NSN AS INTEGER VARIABLE

DEMAND.INTERVAL=1 ''days between generated demands, for DDR=1 day
REVIEW.INTERVAL=2 ''days between review of inventory for breaches
LET BUFFER.V=1000

IF NEWASSUMP.OPT=.FALSE
''THEN use standard assumptions
''
***** PROCUREMENT CYCLE VALUES *****
T=365 ''ordering and holding cost constant
M1=925 ''dollar value quarterly demand floor, < M1 PCP=36 mth
M2=9999 ''dollar value quarterly demand ceiling, >M2 PCP=6 mth
ELSE ''read file with optional assumptions
CALL OPTIONAL.ASSUMPTIONS
ALWAYS

'' *************** COVAR & CONFIDENCE INTVL. variables ***************
COVAR.INTERVAL=180 ''Interv. betw. sample points for covariance calc.
QUICK.INTERVAL=2*360 ''time betw. cont. quick covar. calc.
K.LAG=6 ''no. of lags that separate covariances are calculated
M.COVAR=4 ''number of lag terms in full covariance & confid. Intvl
N.BLOCKS=LENGTH.OF.SIMULATION/COVAR.INTERVAL
IF COVARNSN.OPT=.TRUE
'' THEN do covariances & confidence interv. for all NSNs and total PGC
PGC.NUM=MAX.NSN+1
ELSE
PGC.NUM=1
ALWAYS
CREATE EVERY COVAR.INFO(PGC.NUM)
RESERVE PRODUCT.MAT(*,*) AND SUM.K.ENDS(*,*) AS (PGC.NUM) BY K.LAG
''************
CREATE EVERY DEMAND.BO(MAX.NSN)
''** if no recruit info make NSN detail dimension = 1
CREATE EVERY NSN.DETAIL(2)
'' prepare for divide by 0 error
FOR NSN=1 TO MAX.NSN
   FOR TYPE=1 TO 2 DO
      SUM.REQ.SIZE(NSN,TYPE) = .MINVAL
      NO.REQ.SIZE(NSN,TYPE) = .MINVAL
      SUM.REQ.BO(NSN,TYPE) = .MINVAL
      SUM.UNIT.BO(NSN,TYPE) = .MINVAL
   LOOP

   IF ICC="P"
      THEN do CTRerequirements matrix statistics
         RESERVE MEAN.CTREQ(*) AND STD.CTREQ(*) AS MAX.NSN
      FOR NSN=1 TO MAX.NSN DO
         FOR MONTH=1 TO MAX.MONTH DO
            COMPUTE
               MEAN.CTREQ(NSN) AS THE MEAN AND
               STD.CTREQ(NSN) AS THE STD.DEV OF
               CTREQ.MAT(NSN,MONTH,.TOTAL)
            LOOP
         AVE.FORECAST(NSN)=MEAN.CTREQ(NSN) + (QFD(NSN)/3)
      LOOP
      ELSE 'do QFD item only
      FOR NSN=1 TO MAX.NSN
         AVE.FORECAST(NSN) = (QFD(NSN)/3)
   ALWAYS

END'"PCG.INITIALIZE

PROCESS PLACE.PGC.ORDER
   'This process checks the inventory position of all NSNs in the PGC at
   'the time of breach. The process determines whether any of the other
   'NSNs will breach their ROP within the next minimum procurement cycle.
   'It then calls the CALC.ORDER.QTY to determine the specific NSN
   'order quantity of the NSNs that will be ordered. The process then
   'waits an ALT + 1ST delivery days + a PLT delay before calling
   'RECEIVE.PGC.ORDER. It then waits 30 days for each additional phased
   'order (again calling RECEIVE.PGC.ORDER) until the entire order is
   'received.

   DEFINE ORDER.QTY.MAT AS A INTEGER, 1-DIMENSIONAL ARRAY
   DEFINE HOLD.PLT, PC.DMD AS REAL VARIABLES
   DEFINE SUM.ORDERS, ORDER.QTY, NSN, ASSET.POSITION, SCH.MONTH,
      ORDER.NUM AS INTEGER VARIABLES
   RESERVE ORDER.QTY.MAT(*) AS MAX.NSN
   '****** determine which NSNs and how much to order
   FOR NSN=1 TO MAX.NSN DO
      ASSET.POSITION=STOCK(NSN) + ONORDER(NSN) - UNIT.BO(NSN,.TOTAL)
      CALL SUM.FORECAST.OVER.TIME(NSN,MIN.PC) YIELDING PC.DMD
      IF (ASSET.POSITION-PC.DMD) <= ROP.QTY(NSN)
         THEN this NSN will breach soon so order more
            NOW CALC.ORDER.QTY GIVEN NSN YIELDING ORDER.QTY
            ORDER.QTY.MAT(NSN)=ORDER.QTY
            ONORDER(NSN)=ONORDER(NSN) + ORDER.QTY
   END"
SUM.ORDERS=SUM.ORDERS + ORDER.QTY

ALWAYS
LOOP
IF SUM.ORDERS=0
   'THEN ROP has changed and is false order so stop process
   RETURN
ALWAYS
ORDER.NUMBER=ORDER.NUMBER + 1
ORDER.NUM=ORDER.NUMBER

'*** NOTE: PLT below is from any NSN since all the same, change soon
IF PLT.OPT=.FALSE
   'THEN input PLT = simulated PLT, or hold PLT constant, no variability
   SIM.PLT.DAY=FIRST.DELIVERY
ELSE 'draw from production leadtime delay distribution
   'hold for ave. stat, use PLT shape nob
   HOLD.PLT=FIRST.DELIVERY + (PLT.DAY.DELAY.F * PLT.OPT)
   IF HOLD.PLT<10
      ' THEN order will arrive before placed so set to 10 days
      SIM.PLT.DAY=10
   ELSE
      SIM.PLT.DAY = HOLD.PLT
ALWAYS
ALWAYS
CALL PRINT.ORDER (ORDER.QTY.MAT(*),.TRUE,ORDER.NUM)
WAIT (ALT.DAY + SIM.PLT.DAY) DAYS 'first incremental phased delivery
FOR SCH.MONTH = 1 TO MAX.DELIVERIES DO
   CALL RECEIVE.PGC.ORDER GIVEN
      ORDER.QTY.MAT(*), SCH.MONTH, ORDER.NUM
   IF SCH.MONTH < MAX.DELIVERIES
      WAIT .DPM DAYS
ALWAYS
LOOP
   CALL PRINT.ORDER (ORDER.QTY.MAT(*),.FALSE,ORDER.NUM)
END 'process PLACE.PGC.ORDER

ROUTINE PLOT.ATEND
   'This routine plots the histogram, BO & demand pie charts, fill rate
   ' meters at the end of program run.
   DEFINE ANS, COL, PROB, NSN AS INTEGER VARIABLE

   IF (WARMUP.PERIOD=0) AND (TARGET.PGC <> '1505)
      ' THEN no graph to hold on screen & print histogram to show done
   READ ANS
   ALWAYS

   IF TRACE29=.TRUE
      'THEN print histograms
      '******** DYNAMIC GRAPHICS INITIALIZATION **********
      'set virtual terminal
      DEFINE DEVICE2.ID AS POINTER VARIABLE
      CALL DEVINIT.R("VT,GRAPHIC") YIELDING DEVICE2.ID
      OPEN 9 FOR INPUT, DEVICE=DEVICE2.ID
OPEN 10 FOR OUTPUT, DEVICE=DEVICE2.ID
USE 10 FOR GRAPHIC OUTPUT

'******** calculating the % of time w/ EDO value distribution ********
'******** if prob. = 20 % goes through loop 20 times ********
FOR COL=1 TO (((HIS.BO.2-HIS.BO.1)/HIS.BO.3)+1)
   FOR PROB=1 TO TRUNC.F(((HIST.PGC.STOCK(COL)
      /(LENGTH.OF.SIMULATION/PLOT.INTERVAL))*100)+0.5)
      PGC.PLOT=(HIS.BO.1 + (COL*HIS.BO.3))-0.5 'histog. point
      DISPLAY HISTOGRAM HIST.PGC.PLOT
      IF TRACE23=.TRUE
      PRINT 2 LINES WITH PLOT.YSCALE THUS
      & PROBABILITY SIZE DISTRIBUTIONS PGC NET STOCK (YScale= **.**)
      COL   SL INTERVAL   SL VALUE   PLOT % Prob. NO.OF_SAMPLES
      FOR COL=1 TO (((HIS.BO.2-HIS.BO.1)/HIS.BO.3)+1) DO
         PRINT 1 LINE WITH COL,((COL*0.5)-1.5), ((COL*0.5)-1),
            (HIS.BO.1 + COL*HIS.BO.3), HIST.PGC.PLOT(COL),
            HIST.PGC.STOCK(COL) THUS
            ** **.* < SL < **.**    < **    **    **    **
         LOOP
      ALWAYS

'Used to hold graph on screen & not switched by next graph
READ ANS

ALWAYS 'end of histogram plot

IF TRACE23=.TRUE
' THEN graph the pie chart for UNIT EBOs and UNIT demands
   DEFINE DEVICE4.ID AS POINTER VARIABLE
   DEFINE TOT.EDO AS REAL VARIABLES
   DEFINE TOT.UNIT.DEM AS INTEGER VARIABLES
   CALL DEVINIT.R("VT,GRAPHIC") YIELDING DEVICE4.ID 'set Virt. term.
   OPEN 15 FOR INPUT, DEVICE=DEVICE4.ID
   OPEN 16 FOR OUTPUT, DEVICE=DEVICE4.ID
   USE 16 FOR GRAPHIC OUTPUT
   FOR NSN=1 TO 3 DO 'total EBOs and unit Demands for all NSN
      TOT.EDO=TOT.EDO + AVE.UNIT.EDO(NSN,.TOTAL)
      TOT.UNIT.DEM=TOT.UNIT.DEM + SUM.REQ.SIZE(NSN,.TOTAL)
   LOOP
   IF TOT.EDO<>0 ' THEN can print EBO pie since EBOs do not equal zero
      FOR NSN=1 TO 3 DO 'calculate % for 1st 3 NSNs in pie chart
         EBO.PIE(NSN)=AVE.UNIT.EDO(NSN,.TOTAL)/TOT.EDO
         DEMAND.PIE(NSN)=SUM.REQ.SIZE(NSN,.TOTAL)/TOT.UNIT.DEM
      LOOP
      'average annual unit demand at end of simulation for PGC
      TOT.UNIT.DEM=TOT.UNIT.DEM/(LENGTH.OF.SIMULATION/(12*.DPM))
      DISPLAY EBO.PIE
      DISPLAY DEMAND.PIE
   ALWAYS
LET VXFORM.V = 5
CALL SETWORLD.R (0, 79, 0, 23)
CALL MXRESET.R (0)
CALL MXLATE.R (5, 2) 'X,Y coordinates position
CALL TEXTANGLE.R (0) 'angle of the text from 0 to 3600
WRITE TOT.EBO, TOT.UNIT.DEM AS "PGC TIME WEIGHTED BACKORDERS",
D(8, 1), " PGC ANNUAL DEMANDS", I 7, / USING 16
CALL MXLATE.R (20, 17) 'X,Y coordinates position
WRITE AS "UNIT BACKORDERS AND DEMANDS", / USING 16
CALL GUPDATE.R
READ ANS

ALWAYS

IF TRACE11 = .TRUE. 'display fill rate meters

' ************** FILL RATE GRAPHICS **************
DEFINE SUM.DEM, SUM.BO AS INTEGER, 1-DIMENSIONAL ARRAY
RESERVE SUM.DEM(*), SUM.BO(*) AS 2
DEFINE DEVICE3.ID AS POINTER VARIABLE
DEFINE TYPE AS INTEGER VARIABLE
CREATE EVERY PLOTNSN(3+2)

CALL DEVINIT.R("VT, GRAPHIC") YIELDING DEVICE3.ID 'set Virt. term.
OPEN 13 FOR INPUT, DEVICE=DEVICE3.ID
OPEN 14 FOR OUTPUT, DEVICE=DEVICE3.ID
USE 14 FOR GRAPHIC OUTPUT
DISPLAY FILLRATE(3+.TOTAL) WITH "FILRTPGC.GRF"
DISPLAY FILLRATE(1) WITH "FILRT1.GRF"
DISPLAY FILLRATE(2) WITH "FILRT2.GRF"
DISPLAY FILLRATE(3) WITH "FILRT3.GRF"

' *** calculating total & recruit PGC REQUISITIONS fills for all NSNs
' *** do NSNs fill rates
TYPE=.RECRUIT
FOR NSN=1 TO 3 DO
  FILLRATE(NSN)=100 * (1-
    (SUM.REQ.BO(NSN,TYPE)/NO.REQ.SIZE(NSN,TYPE)))
LOOP

' *** do total and recruit fill rates
FOR TYPE = 1 TO 2 DO '' if want dynamic fill rates, initial sums
  SUM.BO(TYPE)=0
  SUM.DEM(TYPE)=0
LOOP
FOR TYPE=1 TO 2
  FOR NSN=1 TO 3 DO
    SUM.BO(TYPE)=SUM.BO(TYPE) + SUM.REQ.BO(NSN,TYPE)
    SUM.DEM(TYPE)=SUM.DEM(TYPE) + NO.REQ.SIZE(NSN,TYPE)
  LOOP
  FILLRATE(3+.TOTAL)=(1-(SUM.BO(.TOTAL)/SUM.DEM(.TOTAL)))*100
  IF DOREQ.OPT = .TRUE.
  THEN RTC fill rates, else no requisitions so no RTC fill rates
  DISPLAY FILLRATE(3+.RECRUIT) WITH "FILRTRTC.GRF"
  FILLRATE(3+.RECRUIT)=
    (1-(SUM.BO(.RECRUIT)/SUM.DEM(.RECRUIT)))*100
LOOP

ALWAYS
LET VXFORM.V = 5 'mapping from real world to normalized
CALL SETWORLD.R (0,79,0, 23)
CALL MXLATE.R (20,3) 'X,Y coordinates position
CALL TEXTANGLE.R (0) 'angle of the text from 0 to 3600
WRITE AS " \-------- RECRUIT SUPPLY AVAILABILITY --------/",
/ USING 14
CALL MXRESET.R (0) 'resets pointer to given object, 0=null
CALL MXLATE.R (22,0) 'X,Y coordinates position
WRITE FILLRATE (3 + TOTAL) AS " PGC SUPPLY AVAILABILITY",
D(5,0),"%",/USING 14
CALL MXLATE.R (3,21)
WRITE AS "REQUISITION SUPPLY AVAILABILITY", /USING 14
CALL GUPDATE.R
CALL mxreset.r(0)
READ ANS
ALWAYS 'end of fill rate plot

END 'PLOT.ATEND

ROUTINE PRINT.ASSUMPTIONS
'Prints all pertinent assumptions and variables for the run including
'options, query answers, safety level, OWRM, PLT, ALT, M1, M2, T, COST
' ARS, RTC CUTOFF, VIP, XYZ. MONTH ETC.
DEFINE NSN AS INTEGER VARIABLES
DEFINE PER.SD AS REAL VARIABLE
LINES .V-0
PRINT 2 LINES THUS

=================================
PRINT LINES WITH TARGET.PGC, DMDMAD.OPT, PLT.OPT, SHORT.OPT,
(LENGTH.OF.SIMULATION/360), (END.OF.SIMULATION/360), ADDPGC.OPT,
VSL.OPT, MODIFYDATA.OPT, MDMPT011.OPT, NEWASSUMP.OPT,
POISSON.OPT, DOREQ.OPT, MAPE.OPT, NORMAL.OPT, COVARNSN.OPT THUS
======== MODEL OPTION ASSUMPTIONS (true=1 and false=0) ========
1) PGC NUMBER **
2) SIMULATED DEMAND KNOB *** (0:FALSE = DEMAND IS FORECAST, else MAD)
3) PLT DAYS DELAYED KNOB *** (0:FALSE= Constant PLT, else variance)
4) SHORT RUN WITH PLOT ** (0:FALSE=longer run for definitive results)
5) LENGTH OF SIMULATION ** TOTAL LENGTH OF RUN WITH WARMUP **
6) **: 0 DO NOT ADD; 1=runs for same PGC(10 = 1st PGC in group);
2= add different PGC info (20 = 1st PGC in group)
8) VARIABLE SAFETY LEVEL OPTION ** (0:FALSE= FIXED SAFETY LEVEL)
9) EDITED THE SSCF DATA ** (0:FALSE= use standard data with no change)
10) EDITED MPT011 TABLE ** (0:FALSE= use standard data with no change)
11) EDITED ASSUMPTIONS ** (0:FALSE = standard assumptions, no change)
o DAILY DEMAND RATE FROM POISSON DIST. ** (0:FALSE=MONTHLY DEMAND/30)
o REQUISITION GROUPINGS FOR DEMANDS ** (0:FALSE=REQ.SIZE=DDR each day)
o SIMULATED DEMAND via MAPE ** (0:FALSE =NO adjustments used)
o NORMAL CTRREQ DISTRIBUTION ** (0:FALSE= 1st 3yrs. are actual CTRREQ)
o COVARIANCE FOR ALL NS~s ** (0:FALSE= only PGC covar calculated)

PRINT 4 LINE WITH ALT.DAY, FIRST.DELIVERY, COST, M1, M2, T THUS
------------------------ KEY VARIABLES USED IN RUN ------------------------
ALT ** PLT OF FIRST DELIVERY ** COST $ **,** M1 **,* M2 **,* T **

PRINT 1 LINE THUS
NSN RTC CUT ARS RTN STOCK OWRM PLT SAFETY MTHS $SD/AMF VIP XYZ
FOR NSN=1 TO MAX.NSN DO
'callculate % of monthly stand. deviat. of MAD divide by forecasts
PER SD = (100 * MAD(NSN) * 1.25) / AVE.FORECAST(NSN)
IF VIP.ITEM(NSN) = .FALSE
PER SD = PER SD / SQRT.F(3) 'adjust from quarterly to monthly
ALWAYS
PRINT 1 LINE WITH NSN, RECRUIT.SIZE.CUTOFF(NSN), ARS(NSN),
PER.RTC.DEMAND(NSN)*100, STOCK(NSN), OWRM(NSN), PLT.DAY(NSN),
SAFETY.MONTH(NSN), PER SD, VIP.ITEM(NSN), XYZ.MTHM(NSN) THUS
** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
LOOP
END ' 'routine PRINT.ASSUMPTIONS

ROUTINE PRINT.ATEND
'This routine prints the table of summary statistics during and at the
'end of simulation: requisition vs unit, total vs RTC, for EBO,
'AVBOD, fill rates, and demands/yr
DEFINE NSN, TYPE AS INTEGER VARIABLES
DEFINE FOR.TIME AS REAL VARIABLE

IF TIME.V <= WARMUP.PERIOD
FOR.TIME=time.V + (.MINVAL*100000)
ELSE
FOR.TIME=TIME.V - WARMUP.PERIOD
ALWAYS
PRINT 3 LINES THUS
----------REQUISITIONS--------- ----------UNIT DEMANDS----------
=AVBOD== =DEM/YR== =AVBOD== =DEM/YR==
NSN TOT RTC TOT RTC TOT RTC TOT RTC
FOR NSN=1 TO MAX.NSN DO
BEGIN REPORT PRINTING
FOR TYPE=1 TO 2 IN GROUPS OF 2
PRINT 1 LINE WITH NSN,
A GROUP OF ((AVE.REQ.EBO(NSN,TYPE)*FOR.TIME)
/SUM.REQ.BO(NSN,TYPE)) FIELDS,
A GROUP OF (360*NO.REQ.SIZE(NSN,TYPE)/FOR.TIME) FIELDS,
A GROUP OF ((AVE.UNIT.EBO(NSN,TYPE)*FOR.TIME)
/SUM.UNIT.BO(NSN,TYPE)) FIELDS,
A GROUP OF (360*SUM.REQ.SIZE(NSN,TYPE)/FOR.TIME)
FIELDS THUS
** **,** **,** **,** **,** **,** **,** **,** **,** **,** **,** **,** **,**
END ' 'REPORT

2-38
**LOOP**

```plaintext
PRINT 3 LINES THUS
=========REQUISITIONS=========
=========UNIT DEMANDS=========
=========EBOs========
--FILL RATES--
=========EBOs========
--FILL RATES--

NSN TOT   RTC TOT   RTC   TOT   RTC   TOT   RTC
```

FOR NSN=1 TO MAX.NSN DO
BEGIN REPORT PRINTING
FOR TYPE=1 TO 2 IN GROUPS OF 2
PRINT 1 LINE WITH NSN,
   A GROUP OF AVE.REQ.EBO(NSN,TYPE) FIELDS,
   A GROUP OF (100*{(SUM.REQ.BO(NSN,TYPE)
       /NO.REQ.SIZE(NSN,TYPE))}) FIELDS,
   A GROUP OF AVE.UNIT.EBO(NSN,TYPE) FIELDS,
   A GROUP OF (100*{(SUM.UNIT.BO(NSN,TYPE)/
       SUM.REQ.SIZE(NSN,TYPE))}) FIELDS
   THUS
   ** ** ** ** ** ** ** ** ** **
   END '' REPORT
LOOP
END ''PRINT.ATEND

**ROUTINE PRINT.DELIVERY.MATRIX**

This routine prints the delivery matrix: the 3 rows are X, Y, Z;
'columns are for the number of deliveries. Values are the fraction
'of the total item (X,Y,Z) order delivered that month (i.e. each row
'sums to 1.

DEFINE ROW, MONTH AS INTEGER VARIABLES

PRINT 3 LINE THUS

```
DELIVERY MATRIX FOR X, Y, AND Z ITEMS

XYZ 1 2 3 4 5 6 ! SUM%
```

FOR ROW = 1 TO 3 DO
BEGIN REPORT PRINTING
FOR MONTH = 1 TO MAX.DELIVERIES IN GROUPS OF 6
PRINT 1 LINE WITH ROW, A GROUP OF XYZ.MATRIX(ROW,MOMTH)
   FIELDS, XYZ.SUM(ROW) THUS
   ** ** ** ** ** ** ** ** ** **
   END '' REPORT
LOOP

IF TRACE22=.TRUE
BEGIN REPORT PRINTING
FOR MONTH = 1 TO MAX.DELIVERIES IN GROUPS OF 6
PRINT 2 LINES WITH A GROUP OF DELIVERY.PERCENT(MONTH)
   FIELDS THUS
   PGC DELIVERY PERCENTS
   ** ** ** ** ** ** ** ** ** **
   END '' REPORT
ALWAYS

---

2-39
END "routine PRINT.DELIVERY.MATRIX"

PROCESS PRINT.DEMANDS
'This process gives requisition size and interval, AMF, AMF/AND ratio, 'onhand, onorder information at specified intervals, calls 'PRINT.ATEND, and gives cumulative BOs and demands.

DEFINE NSN, TYPE AS INTEGER VARIABLES
DEFINE YEARS AS REAL VARIABLE

WAIT TRACE. INTERVAL DAYS
UNTIL TIME.V > END.OF.SIMULATION DO
  IF TIME.V <= WARMUP.PERIOD
    YEARS=TIME.V/360
  ELSE
    YEARS= (TIME.V - WARMUP.PERIOD)/360
  ALWAYS

  IF TRACE2=.TRUE
    THEN print the following (NOTE: ARS.SIM is based on all requisitions 'except the current REQ.size.now that has not hit the inventory, 'SIM DD is the total monthly demand felt by the inventory
    PRINT 4 LINES WITH AT.MONTH, YEARS, TIME.V THUS

    END OF MONTH DATA: MONTH ** YEAR ***(time.v **)
    =CUMULATIVE= =AMF== RATIO ====AVE UNITS======= ==AVE MONTHS==
    NSN ARS.SM INTRVL FORCST FOR/DD ONORDER ONHAND %0/0 OR/F OH/F WAR

FOR NSN= 1 TO MAX.NSN DO
  PRINT 1 LINE WITH NSN,
  AVE.REQ.SIZE(NSN,.TOTAL),AVE.REQ.INTERVAL(NSN,.TOTAL),
  SUM.FORECAST(NSN)/(YEARS *12),
  (100*SUM.FORECAST(NSN)/SUM.REQ.SIZE(NSN,.TOTAL)),
  AVE.ONORDER(NSN), AVE.STOCK(NSN),
  (100*AVE.ONORDER(NSN)/(AVE.STOCK(NSN)+AVE.ONORDER(NSN))),
  (AVE.ONORDER(NSN)/((SUM.FORECAST(NSN)/YEARS)/12)),
  (AVE.STOCK(NSN)/((SUM.FORECAST(NSN)/YEARS)/12)),
  (OWRM(NSN)/((SUM.FORECAST(NSN)/YEARS)/12))
  THUS
  * **.* ** ** ** ** ** ** ***
  LOOP
  ALWAYS

  IF TRACE3=.TRUE
    CALL PRINT.ATEND
  ALWAYS

  IF TRACE3=.TRUE
    THEN print BOs & Demands
    PRINT 4 LINES WITH AT.MONTH THUS

    MONTH ** ===REQUISITIONS BACKORDERS==== ========UNIT BACKORDERS=====
    === CUM === ==CURRENT======= === CUM === ==CURRENT=====
    NSN TOT RTC TOT RTC TOT RTC TOT RTC TOT RTC

2.40
FOR NSN=1 TO MAX.NSN DO
BEGIN REPORT PRINTING
FOR TYPE=1 TO 2 IN GROUPS OF 2
PRINT 1 LINE WITH NSN,
A GROUP OF SUM.REQ.BO(NSN,TYPE) FIELDS,
A GROUP OF REQ.BO(NSN,TYPE) FIELDS,
A GROUP OF SUM.UNIT.BO(NSN,TYPE) FIELDS,
A GROUP OF UNIT.BO(NSN,TYPE) FIELDS
** ** **
END" REPORT
LOOP
ALWAYS
WAIT TRACE.INTERVAL DAYS
LOOP "of Until
END "process PRINT.DEMANDS

PROCESS PRINT.LEVELS
"This process prints the requirements, PC, MIN.PC, stock, backorders
' at specified intervals

DEFINE NSN.I AS INTEGER VARIABLES "array indices

UNTIL TIME.V > END.OF.SIMULATION DO
PRINT 4 LINES WITH AT.MONTH, TIME.V THUS
BEGINNING OF MONTH ** C & T LEVELS & DEMANDS BY NSN (time.v **)
NSN 30xDDR FORCTS PCP.MTH MIN.PC ROP QTY STOCK ORDER UBO RBO
FOR NSN.I=1 TO MAX.NSN
PRINT 1 LINE WITH NSN.I, (DDR(NSN.I)*(.DPM/DEMAND.INTERVAL)),
FORECAST.MTH(NSN.I),PCP.MONTH(NSN.I),MIN.PC, ROP.QTY(NSN.I),
STOCK(NSN.I),ONORDER(NSN.I),UNIT.BO(NSN.I,1), REQ.BO(NSN.I,1) THUS
** ** ** ** ** ** **
WAIT TRACE.INTERVAL DAYS
LOOP
END".routine PRINT.LEVELS

ROUTINE PRINT.ORDER GIVEN ORDER.MAT, SENDOUT, AND ORDER.NUM
"Prints the order quantity placed for all NSNs, PLT delays, when
' will come in, and inventory positions all at time of breach and
' when order completely delivered.

IF TRACE5=.TRUE
DEFINE BO.MEMBER, NSN, ORDER.NUM, SENDOUT AS INTEGER VARIABLES
DEFINE ORDER.MAT AS INTEGER, 1-DIMENSIONAL ARRAY
RESERVE ORDER.MAT(*) AS MAX.NSN

IF SENDOUT=.TRUE
' THEN just placed the following order
PRINT 2 LINES WITH ORDER.NUM, FIRST.DELIVERY,
SIM.PLT.DAY-FIRST.DELIVERY, (TIME.V+SIM.PLT.DAY+ALT.DAY) THUS
ORDER NO. ** 1ST PLT DELIVERY ** PLT DELAY ** COME IN(LT) **
NSN ORDER ROP STOCK ONORDER U.BO U.BO_RTC REQ.BO REQ.B_RTC

' TRACE1=.TRUE
ELSE
' IF TRACE12=.TRUE
' THEN **** Print backorder requisition queue
FOR NSN=1 TO MAX.NSN
   FOR EACH BO.MEMBER IN REQ.BO.QUEUE(NSN) DO
      PRINT 1 LINE WITH NSN, BO.TYPE(BO.MEMBER),
               BO.SIZE(BO.MEMBER) THUS
      NSN ** TYPE OF BO ** SIZE OF BO **
   LOOP
ALWAYS

PRINT 3 LINES WITH ORDER.NUM, TIME.V THUS

xxxxxxxx RECIEVED ALL OF ORDER ** AT TIME **.* xxxxxxxxxxxxxx
NSN ORDER ROP STOCK ONORDER U.BO U.BO_RTC REQ.BO REQ.B_RTC

' TRACE1=.FALSE
ALWAYS

FOR NSN = 1 TO MAX.NSN DO
   PRINT 1 LINE WITH NSN, ORDER.MAT(NSN), ROP.QTY(NSN),STOCK(NSN),
               ONORDER(NSN), UNIT.BO(NSN,.TOTAL), UNIT.BO(NSN,.RECRUIT),
               REQ.BO(NSN,.TOTAL), REQ.BO(NSN,.RECRUIT) THUS
   ** ** ** ** ** ** **
LOOP
ALWAYS
END ''routine PRINT.ORDER

ROUTINE PRINT.PGCSTATS
' 'This routine prints the summary statistics at the end of the
 ' simulation at a PGC level. Specifically, backorders,
 ' availabilities, and annual demands (requisition & unit levels);
 ' onhand, onorder, orders/yr values; and calibration information
 ' confidence intervals, % onorder to total stock, etc.

DEFINE I, NSN, TYPE AS INTEGER VARIABLES
DEFINE PER.CI, PGC.STOCK, PGC.ONORDER, PGC.FORECAST, FOR.TIME
   AS REAL VARIABLE
DEFINE PGC.REQDEM, PGC.UNITDEM, PGC.REQBO, PGC.UNITBO, PGC.REBO,
   PGC.UBO AS REAL, 1-DIMENSIONAL ARRAY
RESERVE PGC.REQDEM(*), PGC.UNITDEM(*), PGC.REQBO(*), PGC.UNITBO(*),
   PGC.REBO(*), PGC.UBO(*) AS 2
DEFINE NEWPGC AS REAL, 1-DIMENSIONAL ARRAY
RESERVE NEWPGC(*) AS 11
LINES.V=0

IF TIME.V <= WARNUP.PERIOD
   FOR.TIME=TIME.V + .MINVAL
ELSE
   FOR.TIME=TIME.V - WARNUP.PERIOD
ALWAYS

' ' sum from NSN to PGC level
FOR TYPE=1 TO 2
   FOR NSN=1 TO MAX.NSN DO
      PGC.REBO(TYPE)= PGC.REBO(TYPE) + AVE.REQ.EBO(NSN,TYPE)
      PGC.UEBO(TYPE)= PGC.UEBO(TYPE) + AVEUNIT.EBO(NSN,TYPE)
      total number of backorders used in fillrate/availability
      PGC.REQBO(TYPE)=PGC.REQBO(TYPE) + SUM.REQ.BO(NSN,TYPE)
      PGC.UNITBO(TYPE)=PGC.UNITBO(TYPE) + SUM.UNIT.BO(NSN,TYPE)
      total demands over simulation
      PGC.REQDEM(TYPE)=PGC.REQDEM(TYPE) + NO.REQ.SIZE(NSN,TYPE)
      PGC.UNITDEM(TYPE)=PGC.UNITDEM(TYPE) + SUM.REQ.SIZE(NSN,TYPE)
   LOOP
   FOR TYPE=1 TO 2 DO ' ' check for divide by zero
      IF PGC.REQDEM(TYPE)=0
         PGC.REQDEM(TYPE)= .MINVAL
      ALWAYS
      IF PGC.UNITDEM(TYPE)=0
         PGC.UNITDEM(TYPE)=0
      ALWAYS
   LOOP

FOR NSN=1 TO MAX.NSN DO
   ADD AVE.STOCK(NSN) TO PGC.STOCK
   ADD AVE.ONORDER(NSN) TO PGC.ONORDER
   ADD SUM.FORECAST(NSN) TO PGC.FORECAST
   LOOP

FOR I = 1 TO 2 DO ' '2 prints of below 1st to file, 2nd to screen
   PRINT 6 LINE WITH PGC.NO, RUN.ID, PGC.NAME, MAX.NSN, COST, 
   (ORDER.NUMBER/(LENGTH.OF.SIMULATION/360)) THUS
====================================================================
====================================================================
---------------------------------------------------------------------
PGC NAME ********** NSNs ** COST ** Orders/YR **
---------------------------------------------------------------------
TOT RTC TOT RTC TOT RTC

BEGIN REPORT PRINTING
FOR TYPE=1 TO 2 IN GROUPS OF 2
PRINT 2 LINES WITH
' ' for requisitions EBOs, availability, demands/yr
   A GROUP OF PGC.REBO(TYPE) FIELDS,
   A GROUP OF (100*(1-(PGC.REQBO(TYPE)
   /PGC.REQDEM(TYPE)))) FIELDS,
   A GROUP OF (360*PGC.REQDEM(TYPE)/FOR.TIME) FIELDS,
for units EBOs, availability, demands/yr
A GROUP OF PGC.UENO(TYPE) FIELDS,
A GROUP OF (100*(1-((PGC.UNITBO(TYPE)/
PGC.UNITDEM(TYPE)))) FIELDS,
A GROUP OF (360*PGC.UNITDEM(TYPE)/FOR.TIME) FIELDS

REQUISIT. **** *** *** *** *** *** ***
UNITS **** *** *** *** *** *** ***

PRINT 4 LINES WITH PGC.STOCK, PGC.ONORDER, PGC.SL.STOCK,
COST*PGC.STOCK, COST*PGC.ONORDER, COST*PGC.SL.STOCK

AVERAGE: ====== STOCK ====== ONORDER ====== SAFETY LEVEL ======

PRINT 4 LINES WITH
TIME.V/360, LENGTH.OF.SIMULATION/360, WARMUP.PERIOD/360,
REVIEW.INTERVAL, DEMAND.INTERVAL, AVE.COVAR.DATA(PGC.NUM),
PER.CI, PLOT.YSCALE*AVE.PGC.NET.STOCK,
(100*PGC.ONORDER/(PGC.STOCK+PGC.ONORDER)),
(100*PGC.FORECAST/PGC.UNITDEM(1)), (360*PGC.FORECAST/FOR.TIME)

CALIBRATION/VALIDATION INFORMATION

TIME.V(YR) *** SIM (YRS) *** WARMUP ** (REVIEW ** DEMAND ** DAYS)
PCC.BO %CI/MEAN AVE NET STOCK %OR/OR+OR % FORE/DEND YR FORCST

IF I=1 'end of first pass to file, switch output to screen
USE UNIT 6 FOR OUTPUT

LOOP

'******** Prepare information to go into table w/ many PGCs
NEWPGC(1)=PGC.NO
NEWPGC(2)=PGC.UENO(1)
NEWPGC(3)=PGC.REBO(1)
NEWPGC(4)=100*(1-(PGC.REQBO(1)/PGC.REQDEM(1)))
NEWPGC(5)=100*(1-(PGC.REQBO(2)/PGC.REQDEM(2)))
NEWPGC(6)=COST*PGC.STOCK/100000
NEWPGC(7)=COST*PGC.ONORDER/100000
NEWPGC(8)=COST*PGC.SL.STOCK/100000
NEWPGC(9)=(30*PGC.UNITDEM(1)/FOR.TIME)/100
NEWPGC(10)=(30*PGC.REQDEM(1)/FOR.TIME)
NEWPGC(11)=(30*PGC.REQDEM(2)/FOR.TIME)

CALL ADD.ALL.PGCS GIVEN NEWPGC(*)
ROUTINE PRINT.QUERIES
'Prints the answers entered by user during interactive session.

PRINT 2 LINES THUS

PRINT 13 LINES WITH
TARGET.PGC, DMDMAD.OPT, PLT.OPT, SHORT.OPT,
.LENGTH.OF.SIMULATION/360,(END.OF.SIMULATION/360), ADDPGC.OPT,
VSL.OPT, MODIFYDATA.OPT, MODMPT011.OPT, NEWASSUMP.OPT

====== MODEL OPTION ASSUMPTIONS (true=1 and false=0) ========
1) PGC NUMBER **
2) SIMULATED DEMAND KNOB. ** (0:FALSE = DEMAND IS FORECAST, else MAD)
3) PLT DAYS DELAYED KNOB ** (0:FALSE= Constant PLT, else variance)
4) SHORT RUN WITH PLOT ** (0:FALSE=longer run for definitive results)
5) LENGTH OF SIMULATION ** TOTAL LENGTH OF RUN WITH WARMUP **
6) : 0 DO NOT ADD; 1=runs for same PGC(10 = 1ST PGC in group);
    2=add different PGC info (20 = 1ST PGC in group)
7) VARIABLE SAFETY LEVEL OPTION ** (0:FALSE= FIXED SAFETY LEVEL)
8) EDITED THE SSCP DATA ** (0:FALSE= use standard data with no change)
9) EDITED MPT011 TABLE ** (0:FALSE= use standard data with no change)
10) EDITED ASSUMPTIONS ** (0:FALSE = standard assumptions, no change)

END ''routine PRINT.QUERIES

PROCESS PRINT.QUICK.COVAR
' This process uses an approximation formula to estimate the
' covariance continuously at intervals throughout the simulation.
' Used primarily to determine end of warmup period and length of
' run as well as the rate of confidence interval change. Uses
' info. collected by COVAR.SAMPLING process and automatically printed
' for long runs

DEFINE NSN, LAG, BLOCK AS AN INTEGER VARIABLE
DEFINE COVAR.SUM, C.I. AS A REAL VARIABLES

****** NOTE: FUNCTION IS WRONG & ONLY APPROXIMATION. ******
' DOES NOT CONSIDER K.LAG ITEMS IN SET FOR Xi*Xi+k PRODUCT
' UNTIL THE LAST INTERVAL ONCE COVAR.SAMPLING HAS STORED THEM

WAIT WARMUP.PERIOD DAYS
UNTIL TIME.V >= END.OF.SIMULATION DO
    WAIT QUICK.INTERVAL DAYS
    BLOCK=TRUNC.F((TIME.V-WARMUP.PERIOD)/COVAR.INTERVAL)
    FOR NSN=1 TO PGC.NUM DO
        COVAR.SUM=0
        FOR LAG=1 TO M.COVAR DO
            COVAR.SUM=COVAR.SUM + ((2/BLOCK)*(PRODUCT.MAT(NSN,LAG) +
                             ((LAG-BLOCK)*(AVE.COVAR.DATA(NSN)**2))))
        LOOP
IF ((AVE.COVAR.DATA(NSN)<>0) 
    AND (VAR.COVAR.DATA(NSN)+COVAR.SUM=0)) 
C.I. = 1.96 * SQRT.P((VAR.COVAR.DATA(NSN)+COVAR.SUM)/BLOCK) 
PRINT 1 LINE WITH NSN, ((TIME.V-WARMUP.PERIOD)/360), 
AVE.COVAR.DATA(NSN),PGC.NET.STOCK*PLOT.YSCALE, C.I., 
(100*C.I./AVE.COVAR.DATA(NSN)) THUS 
QUICK NSN ** YR ** MEAN ** NETSTOCK ** CI ** %CI/AVE ** 
ELSE 
' ' PRINT 1 LINE WITH NSN, BLOCK, AVE.COVAR.DATA(NSN),COVAR.SUM, 
' ' VAR.COVAR.DATA(NSN) THUS 
' ' QUICK NSN** BLOCK ** MEAN ** 2COV/N ** VAR ** 
ALWAYS 
LOOP 
LOOP 'until 
END ''PRINT.QUICK.COVAR 

ROUTINE PRINT.SSCF.DATA 
' 'This routine prints the SSCF data read in by routine INPUT.SSCF.DATA 
DEFINE NSN, COL AS INTEGER VARIABLE 

IF TRACE17=.TRUE 
PRINT 5 LINE WITH PGC.NO, MAX.NSN THUS 
=============================================== 
=============================================== PGC SPECIAL SUPPLY CONTROL FILE INPUT DATA ========= 
PROC.GR.CD ** NUMBER OF NSN ** 
PRINT 2 LINES WITH PGC.NAME, FSC, ICC, ALT.DAY, COST, MAX.MONTH 
THUS 
ITEM NAME FSC ICC ADM.LT STANDARD.PRICE MAX.MONTH 
*************** ** * ** **,** ** 
PRINT 1 LINE THUS 
NSN NIIN PRO.LT VIP.LT VIP.ITEM(NSN), SAFETY.MONTH(NSN), QFD(NSN) THUS 
FOR NSN = 1 TO MAX.NSN 
PRINT 1 LINE WITH NSN, NSN.NO(NSN), PLT.DAY(NSN), VIP.ITEM(NSN), 
SAFETY.MONTH(NSN), QFD(NSN) THUS 
** ******** ** * ** **,** ** 
PRINT 2 LINES THUS 
NSN MAD OWMRP ALPHA ARS PER.RTC.Demand 
FOR NSN =1 TO MAX.NSN 
PRINT 1 LINE WITH 
NSN, MAD(NSN), OWMRP(NSN), ALPHA(NSN), ARS(NSN), 
PER.RTC.Demand(NSN) THUS 
** ******** ** * ** **,** ** *.**** 
IF (ICC="P") AND (TRACE24=.TRUE) 
' 'THEN POI item and print CTREQ matrix 

2-46
FOR NSN = 1 TO MAX.NSN DO

PRINT 2 LINE WITH NSN, NSN.NO(NSN), MAX.MONTH THUS
NSN ** NIIN ******** CT REQUIREMENT MATRIX FOR ** MONTHS =====
MONTHS: 1 ------- 2 ------- 3 ------- 4 ------- 5 ------- 6
BEGIN REPORT PRINTING
FOR COL= 1 TO MAX.MONTH IN GROUPS OF 6
PRINT 1 LINE WITH A GROUP OF CTREQ.MAT(NSN,COL,.TOTAL) FIELDS
THUS
** ** ** ** ** **
END 'REPORT
LOOP
ALWAYS 'CTREQ print

ALWAYS 'trace block
END 'routine PRINT.SSCF.DATA

ROUTINE RECEIVE.PGC.ORDER GIVEN ORDER.QTY.MAT, SCH.MONTH, ORDER.NUM
'This routine adjusts stock and backorders when a stock shipment is
'received from suppliers. It is called by PLACE.PGC.ORDER. It
'uses the XYZ.MATRIX for methods 2, 3, 4 and XYZ.MONTH for method 1.

DEFINE ORDER.QTY.MAT AS A INTEGER, 1-DIMENSIONAL ARRAY
DEFINE I, ORDER.QTY, REQ.BO.MEMBER, NSN,
ORDER.NUM, SCH.MONTH, SUM.ORDER AS INTEGER VARIABLES
RESERVE ORDER.QTY.MAT(*) AS MAX.NSN

*** Determine delivery for this month based on delivery method
FOR NSN=1 TO MAX.NSN DO
ORDER.QTY=0
IF DELIVERY.OPT=1
'THEN Method for clumping is used
IF SCH.MONTH=XYZ.MONTH(NSN)
'THEN NSNs entire order is delivered in this month
ORDER.QTY=ORDER.QTY.MAT(NSN)
ALWAYS
ELSE 'methods 2 to 4 with incremental deliveries so use vector
IF (SCH.MONTH < MAX.DELIVERIES)
'THEN not last month so take percent using item's vector
ORDER.QTY= TRUNC.0F(ORDER.QTY.MAT(NSN)
* XYZ.MATRIX(XYZ.MONTH(NSN),SCH.MONTH))
ELSE 'last month, deliver remaining order in case of rounding
FOR I=1 TO MAX.DELIVERIES-1
SUM.ORDER = SUM.ORDER + TRUNC.0F(ORDER.QTY.MAT(NSN)
* XYZ.MATRIX(XYZ.MONTH(NSN),I))
ORDER.QTY=ORDER.QTY.MAT(NSN) - SUM.ORDER
SUM.ORDER=0

ALWAYS
ALWAYS
IF TRACE19=.TRUE
PRINT 1 LINE WITH ORDER.NUM, NSN, TIME.V, SCH.MONTH,
ORDER.QTY THUS
DELIVER ORDER ** NSN ** TIME.V ** SCH MTH ** QUANTITY **
ALWAYS
**Process delivers adjusting stock, backorders, & onorder levels**

ONORDER(NSN) = ONORDER(NSN) - ORDER.QTY

WHILE (ORDER.QTY > 0) AND (REQ.BO.QUEUE(NSN) NOT EMPTY) DO
  REMOVE THE FIRST REQ.BO.MEMBER FROM THE REQ.BO.QUEUE(NSN)
  ORDER.QTY = ORDER.QTY - BO.SIZE

IF ORDER.QTY >= 0
  THEN can fill this back order totally
  FOR I=1 TO BO.TYPE DO
    "update Backorder statistics"
    REQ.BO(NSN,I) = REQ.BO(NSN,I) - 1
    UNIT.BO(NSN,I) = UNIT.BO(NSN,I) - BO.SIZE
  LOOP

  DESTROY THIS REQ.BO.MEMBER
ELSE
  "have partial requisition fill so update only Unit BOs"
  FOR I=1 TO BO.TYPE DO
    "remaining order quantity"
    UNIT.BO(NSN,I) = UNIT.BO(NSN,I) - (BO.SIZE + ORDER.QTY)
  BO.SIZE = ABS. F(ORDER.QTY) "put partial fill back in Q"
  FILE THIS REQ.BO.MEMBER FIRST IN THE REQ.BO.QUEUE(NSN)

ALWAYS
  LOOP "while"
  IF ORDER.QTY > 0
  THEN have stock remaining after Backorder fill so add
  STOCK(NSN) = STOCK(NSN) + ORDER.QTY

ALWAYS
  LOOP

END "routine RECEIVED.PGC.ORDER"

**Routine REQ.TO.INVENTORY GIVEN NSN**

"This routine updates on hand STOCK and if necessary updates NSN
" and recruit backorders (BO) when ever a requisition/customer
" demand is felt.

DEFINE I, TYPE, SHORTAGE, NSN AS INTEGER VARIABLES

IF REQ.SIZE(NSN,.TOTAL) <= STOCK(NSN)
  THEN reduce stock value
  STOCK(NSN) = STOCK(NSN) - REQ.SIZE(NSN,.TOTAL)
ELSE
  "have a Back order condition"
  SHORTAGE = REQ.SIZE(NSN,.TOTAL) - STOCK(NSN)
  STOCK(NSN) = 0
  IF ((REQ.SIZE(NSN,.TOTAL) >= RECRUIT.SIZE.CUTOFF(NSN))
      AND (DORQ.OPT=TRUE))
    THEN set index to update recruit center Backorder info.
      TYPE = RECRUIT
  ELSE
    TYPE = TOTAL

ALWAYS
  FOR I=1 TO TYPE DO
    REQ.BO(NSN,I) = REQ.BO(NSN,I) + 1
    UNIT.BO(NSN,I) = UNIT.BO(NSN,I) + SHORTAGE
    SUM.REQ.BO(NSN,I) = SUM.REQ.BO(NSN,I) + 1
    SUM.UNIT.BO(NSN,I) = SUM.UNIT.BO(NSN,I) + SHORTAGE
  LOOP
IF TRACE4=.TRUE
  PRINT 1 LINE WITH NSN, SHORTAGE, SUM.UNIT.BO(NSN,1), SUM.UNIT.BO(NSN,2) THUS
  BACKORDER NSN ** SHORTAGE ** TOT UNIT BO ** RTC UNIT BO ** ALWAYS
  CREATE A REQ.BO.MEMBER
  BO.TYPE=TYPE
  BO.SIZE=SHORTAGE
  FILE REQ.BO.MEMBER IN REQ.BO.QUEUE(NSN)
  ALWAYS
END "'REQ.TO.INVENTORY

PROCESS REVIEW.INVENTORY
' Reviews the inventory every REVIEW_INTERVAL days to see if
' inventory position IP (onorder + stock + BO) < ROP. If so will
' activate PLACE.PGC.ORDER to determine which NSNs and how much to buy
DEFINE NSN, BREACH AS INTEGER VARIABLES
WAIT REVIEW INTERVAL DAYS
UNTIL TIME.V > END.OF.SIMULATION DO
  BREACH=.FALSE
  FOR NSN=1 TO MAX.NSN,
    UNTIL BREACH=.TRUE DO
      IF (STOCK(NSN)+ONORDER(NSN)-UNIT.BO(NSN,.TOTAL)) <= ROP.QTY(NSN)
        THEN inventory position < ROP so have breach
    IF TRACES=.TRUE
      PRINT 3 LINE WITH TIME.V, NSN, ROP.QTY(NSN), STOCK(NSN), ONORDER(NSN), UNIT.BO(NSN,.TOTAL) THUS
      BREACH TIME NSN ROP STOCK ONORDER BO
    **,** ** ** ** **
    ALWAYS
    BREACH=.TRUE
    ACTIVATE A PLACE.PGC.ORDER NOW
  ALWAYS
LOOP
  WAIT REVIEW INTERVAL DAYS
LOOP
END "'Process Review.Inventory

ROUTINE RTC.REQUISIT.CUTOFF
'This routine automatically determines the requisition size cutoff.
' All requisition sizes above cutoff will be assumed to come from the
' Recruit Training Centers and if summed their percent demand would
' equal the PER_RTC.DEMAND. This routine finds the point in the
' Requisition distribution where those conditions are meet.
DEFINE RVALL1, RVALL2, PROB1, PROB2, RTC.DEM1, RTC.DEM2 AS REAL VARIABLES
DEFINE NSN AS INTEGER VARIABLES
FOR NSN=1 TO MAX.NSN DO
  RVAL1=0
  RVAL2=0
  RTC.DEM1=0
  RTC.DEM2=0
  FOR EACH RANDOM.E IN REQUISITION.RATIO.F,
    WHILE (1 - PER.RTC.DEMAND(NSN)) > RTC.DEM2 DO
      RVAL1=RVAL2
      PROB1=PROB2
      RVAL2=RVALUE.A(RANDOM.E)
      PROB2=PROB.A(RANDOM.E)
      RTC.DEM1=RTC.DEM2
      calculate the % of demand at this point in the requisit. dist.
      CUM % DEMAND = PRE CUM + (MIDPOINT IN REQ. INTERVAL * PDF )
      RTC.DEM2 = RTC.DEM1 + (((RVAL1+RVAL2)/2) * (PROB2-PROB1))
      LOOP
    ' ' found proper interval, now do interpolation
    ' ' cutoff = (% prob. of interval * interval val. + bot. intvl val)*ARS
    ' ' THEN no divide error for cutoff of zero
    RECRUIT.SIZE.CUTOFF(NSN)=(((1-PER.RTC.DEMAND(NSN))-RTC.DEM1)
      /(RTC.DEM2-RTC.DEM1)) * (RVAL2-RVAL1) + RVAL1) * ARS(NSN)
    IF PER.RTC.DEMAND(NSN)=0
    ' ' THEN correct for rounding error
    RECRUIT.SIZE.CUTOFF(NSN)=RVAL2*ARS(NSN)+1
    ALWAYS
    ALWAYS
    IF TRACE21=.TRUE
      PRINT 9 LINE THUS
      $RTC CUTOFF $DEMD ARS RVAL1 RVAL2 PROB1 PROB2 $DEM1 $DEM2
      ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** **
      ALWAYS
    LOOP
  END ' 'RTC.REQUISIT.CUTOFF

ROUTINE SET.OPTIONS
  ' 'This key routine is where all options are set, queries are asked,
  ' 'traces are defined and set, and I/O units are defined
  DEFINE TIME.VAL, YEAR AS REAL VARIABLE
  DEFINE DETAIL.OPT, GRAPH.ANS, ANS AS INTEGER VARIABLE
  ' ' OPEN UNIT 2 FOR OUTPUT, FILE NAME IS "LPT1:"
  ' ' USE UNIT 2 FOR OUTPUT
  USE UNIT 5 FOR INPUT
  ' ' ********************** OPTIONS SET BELOW **********************
  PRINT 9 LINE THUS
  1)ENTER #### NUMBER 0 TO 5 FOR THE PGC SELECTED TO RUN ****************
NAME | SERVICE | MAX | NSN | PGC NUMBER
-----|---------|-----|-----|-----------
0  | DEMO PGC (MAN'S SHIRT) | ARMY | 3  | 1672
1  | MAN'S COAT | ARMY | 65 | 1765
2  | WOMAN'S SHIRT | AIR FORCE | 21 | 1671
3  | WOMAN'S SKIRT | ARMY | 80 | 1748
4  | MEN'S SHOE | ALL | 113 | 1505
5  | MEN & WOMEN GLOVES | ALL | 17 | 1834
6  | WANT TO ENTER AN ALTERNATE PGC NUMBER

READ ANS
PLOT.YSCALE=1000 '' scale factor for PGC net stock dynamic plot
'TIME.VAL is real time minutes to run ALL NSNs for a simulation year
SELECT CASE ANS
CASE 0
  TARGET.PGC=1672
  TIME.VAL=0.15 * 3 ''MINS/NSN/YR SIM * MAX.NSN
CASE 1
  TARGET.PGC=1765
  TIME.VAL=0.055 * 65
CASE 2
  PLOT.YSCALE=100
  TARGET.PGC=1671
  TIME.VAL=0.055 * 21
CASE 3
  PLOT.YSCALE=100
  TARGET.PGC=1748
  TIME.VAL=0.055 * 80
CASE 4
  TARGET.PGC=1505
  TIME.VAL=0.083 * 113
  TRACE20=.FALSE
  PRINT 1 LINE THUS
  NO DYNAMIC PLOT IS USED FOR SHOES SINCE SLOWS SIMULATION
CASE 5
  TIME.VAL=0.089 * 17
  TARGET.PGC=1834
DEFAULT
  PRINT 2 LINE THUS
1a) ENTER THE PGC NUMBER (NOTE: BOTH THE SSCFSIM.DAT/MOD AND THE
MPT011.DAT/MOD FILES MUST ALREADY HAVE THIS PGC'S DATA WITHIN
READ TARGET.PGC
ENDSELECT

PRINT 4 LINES THUS
2) ENTER 0 FOR DEMAND (CUSTOMER BEHAVIOR) EQUAL TO MONTHLY FORECAST
  1 FOR VARIANCE IN DEMAND BASED ON MAD OF FORECAST
  >0 FOR DEMAND KNOB (e.g., 0.95 DECREASES MEAN DEMAND BY 5%,
   1.05 INCREASES THE MEAN DEMAND 5% IN RELATION TO FORECAST)
READ DMDMAD.OPT ''1 means MAD used, 0 means no MAD adjusted demand

PRINT 5 LINES THUS
3) ENTER 0 FOR CONSTANT PLT (SUPPLIERS BEHAVIOR) EQUALING THE SSCF PLT
  1 FOR VARIANCE IN PLT WITH AVERAGE BEING 2 MONTHS LATE
  >0 FOR PLT SHAPE KNOB(e.g., .5 DECREASES VARIANCE SO AVERAGE

2-51
IS 1 MONTH LATE: 2 INCREASES VARIANCE SO AVERAGE IS
APPROXIMATELY 4 MONTHS LATE)
READ PLT.OPT '>::0 Then a draw from the PLT delay distrib * PLT.OPT
'' knob used, else simulated PLT =input PLT value

''1 if you want to have requisition and recruit info.
'' generated, if 0:False will treat daily demand as the requisition
'' size
DOREQ.OPT=.TRUE

POISSON.OPT=.FALSE ''when true will use Poisson distribution for DDR
'' else will keep DDR constant for each day of month
NORMAL.OPT=.TRUE ''when true will generate normally distributed
random
''CTREQ/month based on actual forecasted, else 1st max months
''will be actual forecasts, rest will be from random normal draws

COVARNSN.OPT=.FALSE '' when what covariances & Confid. interv. for
''NSN and for total PGC, else only for total PGC

'' TO GET C&T REQUIREMENTS AS SIMULATED DEMANDS ENTER 0 FOR NEXT TWO
'' ENTER 1 FOR MAPE ADJUSTMENT or 0 NO MAPE VARIANCE IN DEMANDS
MAPE.OPT=.FALSE ''1 means MAPE used, 0 means no mape adjusted demand

BATCH.OPT=.TRUE ''if true, runs batch mode for several PGCs

'' ****************** TRACZ OPTIONS SET BELOW **********************
'' in routine PRINT.DEMANDS for trace 2 & 3
TRACE17=.TRUE ''prints the values read in from the SSCF file
TRACE6=.TRUE ''prints EBOs, fill rates, AVBOD, DEM/YR for Total & RTCs
TRACE2=.TRUE ''prints annual forecasts vs demands & onhand & onorder
TRACE24=.FALSE ''prints the 3 years of CTREQ from the SSCF
TRACE7=.TRUE ''prints the 1st 5 yr.CTREQ.MAT matrix used in simulation
TRACE9=.TRUE ''prints PGC net stock histogram

PRINT 2 LINE THUS
4) ENTER 0 FOR FINAL RESULTS
   1 FOR SHORT RUN WITH PLOT & DETAIL TRACES
READ SHORT.OPT

PRINT 1 LINE THUS
5) ENTER SIMULATION LENGTH IN YEARS
READ YEAR

IF SHORT.OPT=.TRUE
''THEN ********** SHORT RUN: set detail traces & graphics **********
TRACZ5=.TRUE ''prints at time of breach infor & order value
TRACZ19=.TRUE ''prints each delivery months order received
TRACZ8=.TRUE ''prints demand & forecast for the month to come
TRACZ20=.TRUE ''prints PGC NET STOCK dynamic plot
TRACE.INTERVAL=1*12*30 ''prints the summary end of month stats
WARMUP.PERIOD=0
PLOT.INTERVAL=10 ''accumulates plot data for net stock overtime
ELSE '### LONG RUN: looking for final results ###'

TRACE16=.TRUE 'prints the quick covar & C.I. over time
TRACE7=.FALSE 'prints the first CTREQ.MAT matrix
TRACE.INTERVAL=(YEAR*12*30)/2 'prints the summary end of month stats

WARMUP.PERIOD=5 * 12 * .DPM
PLOT.INTERVAL=90 'accumulates plot data for net stock over time

ALWAYS

' ****************** MISC TRACES ******************
TRACE18=.FALSE 'prints at breach the PTAO, PLT+PCP, DMD/YR in units
TRACE1=.FALSE 'prints the requisit NSN,time, size, & time interval
TRACE4=.FALSE 'prints when BO occurs with NSN & totals for BO
TRACE12=.FALSE 'prints requisition BO queue when get order
TRACE14=.FALSE 'prints PLT stored values, runs PLT 1000 times
TRACE21=.FALSE 'prints RTC cutoff detail info on prob. & intervals
TRACE15=.FALSE 'COVAR sampling information
TRACE22=.FALSE 'Matrix delivery PLTs, %PCP, XYZ vectors & NSNs,
TRACE23=.FALSE 'prints histogram ranges, values, no. in sample

LENGTH.OF.SIMULATION=YEAR*12*30
END.OF.SIMULATION=WARMUP.PERIOD + LENGTH.OF.SIMULATION 'in days

PRINT 5 LINE THUS
6) ENTER 0 NOT TO ACCUMULATE RESULTS ACROSS PGCs
   1 TO DISPLAY RESULTS OF SEVERAL MODEL RUNS WITH THE SAME PGC
   10 TO DESTROY EXISTING RUNS, & START RUNS WITH THE SAME PGC
   2 TO ADD RESULTS OF RUNS OF DIFFERENT PGCs TOGETHER
   20 TO DESTROY EXISTING RUNS, & START RUNS WITH DIFFERENT PGCs
READ ADDPGC.OPT
IF ((ADDPGC.OPT=1) OR (ADDPGC.OPT=10))
   PRINT 1 LINE THUS
6a) ENTER 5 DIGIT RUN ID NUMBER
   READ RUN.ID
   ALWAYS

' ****************** DETAIL QUERIES SET BELOW ******************
PRINT 2 LINE THUS
7) ENTER 0 FOR NO FURTHER CHANGE AND RUN
   1 FOR OPTIONAL INPUT DATA FILES (QUERIES 8 TO 12)
   READ DETAIL.OPT
   GRAPH.ANS=-1
   IF DETAIL.OPT=.TRUE
      THEN ******** do DETAIL QUERY for graphs, files, phasing
   PRINT 2 LINE THUS
8) ENTER 1 FOR VARIABLE SAFETY LEVEL
   0 FOR FIXED SAFETY LEVEL [D]
   READ VSL.OPT

PRINT 2 LINE THUS
9) ENTER 1 FOR OPTIONAL SCF INPUT DATA
   0 FOR STANDARD SCF INPUT DATA [D]
   READ MODIFYDATA.OPT

2-53
PRINT 2 LINES THUS
10) ENTER 1 FOR OPTIONAL MANAGEMENT POLICY TABLE INPUT DATA (MPT011)
   0 FOR STANDARD MANAGEMENT POLICY TABLE INPUT DATA [D]
   READ MODMPT011.OPT

PRINT 2 LINE THUS
11) ENTER 1 FOR OPTIONAL ASSUMPTION FILE: M1,M2,T, OPTIONS, TRACES
    0 FOR STANDARD ASSUMPTIONS [D]
    READ NEWASSUMP.OPT

IF SHORT.OPT=.TRUE
   ' ' THEN **** GRAPHIC TRACE SET OPTIONS ****
   PRINT 4 LINE THUS
12) ENTER 0 FOR NO GRAPHICS
   1 FOR PGC NET STOCK PLOT AND HISTOGRAM [D]
   2 FOR FIRST 3 NSNs NET STOCK PLOT [D - DEMO]
   3 FOR FIRST 3 NSNs NET STOCK PLOT, BO & AVAILABILITY GRAPHS
   READ GRAPH.ANS
   IF (GRAPH.ANS=0) OR (GRAPH.ANS>1)
      Trace20=.FALSE ' 'prints PGC NET STOCK dynamic plot
   ALWAYS
   IF GRAPH.ANS=2
      ' ' THEN below traces assumes first 3 NSNs graphed
      Trace10=.TRUE ' 'print NET.STOCK dynamic plot 1ST 3 NSNs
   ALWAYS
   IF GRAPH.ANS=3
      ' ' THEN below traces assumes first 3 NSNs graphed
      Trace11=.TRUE ' 'prints the FILLRATE meters graphics
      Trace13=.TRUE ' 'prints the EBO pie chart graphics
      Trace12=.TRUE ' 'print NET.STOCK dynamic plot 1ST 3 NSNs
   ALWAYS
   ALWAYS

ALWAYS
CALL PRINT.QUERIES
PRINT 6 LINES WITH (3 * END.OF.SIMULATION * TIME.VAL/(360*60)) THUS

*** THIS MODEL RUN WILL TAKE **.* HOURS REAL TIME ON ZENITH***
---------------------- MODEL RUN SUBMITTED, TO ABORT HIT CTRL-C ---------------------

' ' ********** INPUT/ OUTPUT SPECIFICATIONS **********

IF BATCH.OPT=.TRUE
   ' ' THEN batch mode: runs several PGCs (see ans1, ans2, batchrun files)
   OPEN UNIT 1 FOR OUTPUT
   ELSE ' ' standard run with query's interactive
   OPEN UNIT 1 FOR OUTPUT, FILE NAME IS "C:\SIM\DLA\DLAOUT.DAT"
   ALWAYS
   USE UNIT 1 FOR OUTPUT

IF TARGET.PGC=1672
   ' ' THEN use sample input file
IF MODIFYDATA.OPT=.TRUE
   OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\DEMOPGC.MOD"
ELSE
   OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\DEMOPGC.DAT"
ENDALWAYS
IF (GRAPH.ANS<0) AND (SHORT.OPT=.TRUE)
   'THEN no detail selection and using DEMOPGC so set NSN plot
   TRACE20=.FALSE 'prints PGC NET STOCK dynamic plot
   TRACE10=.TRUE 'print NET.STOCK dynamic plot 1ST 3 NSNs
ENDALWAYS
ELSE
   IF MODIFYDATA.OPT=.TRUE
      OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\SSCFSIM.MOD"
   ELSE
      OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\SSCFSIM.DAT"
   ENDALWAYS
ENDIF NEWASSUMP.OPT=.TRUE
   OPEN UNIT 3 FOR INPUT, FILE NAME IS "C:\SIM\DLA\ASSUMP.MOD"
ENDALWAYS
' matrix delivery schedule info. and first delivery PLT
IF MODMPT011.OPT=.TRUE
   OPEN UNIT 11 FOR INPUT, FILE NAME IS "C:\SIM\DLA\MPT011.MOD"
ELSE
   OPEN UNIT 11 FOR INPUT, FILE NAME IS "C:\SIM\DLA\MPT011.DAT"
ENDALWAYS
' NOTE: UNIT 12 will be OPENed in the "INPUT.VSL" routine.
END ''ROUTINE OPTIONS

PROCESS SET.SIMULATED.DDR
' 'This process updates monthly DDR for the simulation. First converts
'the forecast value to simulated monthly demand via MAPE, MAD, and
''demand KNOB factors if activated and then divides the monthly value
''(30 days/demand interval) to get a daily demand rate (DDR). Note
'' if demand.interval > 1 could be demand for 2, 10, 15 days, whatever.
DEFINE MONTH,NSN.I AS INTEGER VARIABLE
DEFINE DDR.TEMP AS REAL VARIABLE
MONTHLY.MAPE=1 'set to 1 in case no MAPE option
UNTIL TIME.V >= END.OF.SIMULATION
   DO
      AT MONTH=AT MONTH + 1
      IF ICC="P"
         ' THEN calculate month index for CTREQ.MAT
         MONTH=TRUNC.F(TIME.V/.DPM) + 1 'local independent variable
         MONTH= MOD.F(MONTH,CTREQ.PERIOD)
         IF MONTH = 0
            MONTH=CTREQ.PERIOD
         ENDIF
   ENDDO
2-55
ALWAYS FOR NSN.I=1 TO MAX.NSN DO
IF ICC = "P"
   FORECAST.MTH(NSN.I) = CTREQ.MAT(NSN.I,MONTH,.TOTAL) + (QFD(NSN.I)/3)
ELSE
   FORECAST.MTH(NSN.I) = (QFD(NSN.I)/3)
ALWAYS
IF (MAPE.OPT=.TRUE) AND (NSN.I=1)
   "'THEN draw 1 MAPE for all NSNs in the PGC
   MONTHLY.MAPE=DEMAND.MAPE.F
ALWAYS
IF DMDMAD.OPT <> .FALSE
   "'THEN draw a NSN specific variance in demand
   IF VIP.ITEM(NSN.I)=.TRUE
      "'THEN MAD already monthly value & multiply by 1.25 to
      ' convert MAD to stand. deviation
      DDR.TEMP=NORMAL.F((FORECAST.MTH(NSN.I)*DMDMAD.OPT),
      (1.25*MAD(NSN.I)),6) 'stand.dev.
   ELSE "' MAD quarterly value = sum deltas/4 so to get
   ' monthly value divide by SQRT (3).
   ' then multiply by 1.25 to get S.D. or 1.25/sqrt3=.7217
   DDR.TEMP=NORMAL.F((FORECAST.MTH(NSN.I)*DMDMAD.OPT),
      (0.7217 * MAD(NSN.I)),6) 'stand. dev.
ALWAYS
IF DDR.TEMP <1 "'make sure no negative demands
   DDR.TEMP=1
ALWAYS
ELSE
   DDR.TEMP=FORECAST.MTH(NSN.I)
ALWAYS
   ' adjust for mape \ monthly to daily DDR/
   DDR(NSN.I)=(DDR.TEMP/MONTHLY.MAPE)/ (.DPM/DEMAND INTERVAL)
IF TRACES=.TRUE
   PRINT 1 LINE WITH NSN.I, AT MONTH, TIME.V,
   DDR(NSN.I)*(.DPM/DEMAND INTERVAL), FORECAST.MTH(NSN.I) THUS
NSN ** MONTH ** TIME.V ** DEMAND-MTH ** FORCPTS-MTH **
ALWAYS
LOOP
WAIT .DPM DAYS
LOOP
END "SET.SIMULATED.DDR

PROCESS SIMULATION.RUN
   'The general structure of the simulation and starting point for all
   ' processes.
DEFINE NSN, ANS AS INTEGER VARIABLES
   ACTIVATE A WARMUP.RESET IN 0 DAYS
IF ICC = "P"
   "'THEN do CT requirements matrix for POI item
ACTIVATE A UPDATE.CTRZQ.MAT IN 0 DAYS
ALWAYS
ACTIVATE A COMPUTE.ROP.PCP IN 0 DAYS
WAIT 0 DAYS ' ' lets COMPUTE.ROP.PCP be activated & computes ROP

' ' Set initial stock levels
FOR NSN = 1 TO MAX.NSN
STOCK(NSN)=ROP.QTY(NSN)
CALL PRINT.ASSUMPTIONS

ACTIVATE A SET.SIMULATED.DDR IN 0 DAYS
ACTIVATE A REVIEW.INVENTORY IN 0 DAYS

' ' next 2 processes all wait a warm up period before starting
ACTIVATE A COVAR.SAMPLING IN 0 DAYS
IF TRACE16=.TRUE
    ACTIVATE A PRINT.QUICK.COVAR IN 0 DAYS
ALWAYS
FOR NSN=1 TO MAX.NSN DO
    ACTIVATE A DEMAND.GENERATOR GIVEN NSN IN 0 DAYS
LOOP
PRINT 5 LINE THUS

''___________________________________________________________
'' THE SIMULATION OVER TIME _________________________________
''___________________________________________________________

ACTIVATE A PRINT.LEVELS IN WARMUP.PERIOD DAYS
ACTIVATE A PRINT.DEMANDS IN WARMUP.PERIOD DAYS
ACTIVATE A GET.PLOT.DATA IN 0 DAYS ' ' wait a warmup period first
WAIT END.OF.SIMULATION DAYS
CALL PRINT.PGCSTATS
CALL PLOT.ATEND
ANS=ANS ' 'removes warning
PRINT 1 LINE THUS
EXIT MODEL RUN, ENTER INTEGER [RETURN]
READ ANS
STOP
END ' 'process SIMULATION.RUN

ROUTINE SUM.FORECAST.OVER.TIME GIVEN NSN AND PERIOD YIELDING FORECAST
' ' This routine sums the CT POI and QFD requirements over the given period
' ' ( TIME.V to TIME.V + PERIOD) to get a total FORECAST. The PERIOD is
' ' in months or month fractions, a real number. and is used to sum PCP,
' ' Safety level, ROP, MIN.PC, values. With POI items, this routine
' ' can some 8 years of monthly data, however for non POI items the
' ' monthly demand does not chang over time but is QFD/3.

DEFINE SUM.QFD, PERIOD, SUM.CTREQ, FORECAST AS REAL VARIABLES
DEFINE AT.MONTH, MONTH, NSN AS INTEGER VARIABLES

' ' Sum QFD over time PERIOD in months
SUM.QFD = PERIOD * (QFD(NSN)/3)

'** DO POI requirement sum

IF ICC = "P"
  THEN POI item so do CTREQ.MAT forecasts
  Since routine may start in middle of month and period might also
  have a fraction of month add two together.

  AT.MONTH=TRUNC.F(TIME.V/.DPM) + 1
    'local independent variable
    every CTREQ.period the data w/i CTREQ.mat will be shifted
    forward, discarding used data and entering new data now required
    for future time, the following corrects AT.MONTH for this shift
    AT.MONTH= MOD.F(AT.MONTH,CTREQ.PERIOD)
    IF AT.MONTH = 0
      AT.MONTH=CTREQ.PERIOD
    ALWAYS
     PERIOD=PERIOD + FRAC.F(TIME.V/.DPM)
    IF PERIOD < 1.0
      THEN handle the exception case since only will be in 1 month
      SUM.CTREQ=CTREQ.MAT(NSN,AT.MONTH,.TOTAL) *(PERIOD - FRAC.F(TIME.V/.DPM))
    ELSE 'handle standard summing case
      CT REQ fraction for the remaining part of current month
      SUM.CTREQ=CTREQ.MAT(NSN,AT.MONTH,.TOTAL) *(1- FRAC.F(TIME.V/.DPM))
      adding middle months to CTREQ
      FOR MONTH= (AT.MONTH+1) TO (AT.MONTH + TRUNC.F(PERIOD) - 1)
      SUM.CTREQ=SUM.CTREQ + CTREQ.MAT(NSN,MONTH,.TOTAL)
      adding CT REQ fraction from last month of period
      MONTH=AT.MONTH + TRUNC.F(PERIOD)
      SUM.CTREQ=SUM.CTREQ +
        (CTREQ.MAT(NSN,MONTH,.TOTAL)* FRAC.F(PERIOD))
    ALWAYS
    ALWAYS

  ** total forecast of POI and QFD summed over time period**
  FORECAST = SUM.CTREQ + SUM.QFD

END 'routine SUM.FORECAST.OVER.TIME

PROCESS UPDATE.CTREQ.MAT
  This process makes sure there are enough future months of POI
  forecasts so that all levels (ROP and PTAO) can be calculated.
  This process determines the mean and standard deviation for normal
  distribution from the input CTREQ.MAT. Also, Every CTREQ.period
  this process shifts the CTREQ values a period up in the matrix so
  that old values are discarded. It then fills
  in the empty last period spots in the matrix with newly generated
  CTREQ from the normal distribution.
  DEFINE NSN, MONTH, MONTH1 AS INTEGER VARIABLES

PRINT 3 LINE THUS
SUMMARY OF MONTHLY TOTAL FORECAST AND C&T 36 MONTH POI FORECASTS

<table>
<thead>
<tr>
<th>NSN</th>
<th>TOTAL AMP</th>
<th>POI AMP</th>
<th>POI STD</th>
<th>% POI STD/POI AMP</th>
</tr>
</thead>
</table>

FOR NSN=1 TO MAX.NSN DO

PRINT 1 LINE WITH NSN, AVE.FORECAST(NSN),
MEAN.CTREQ(NSN), STD.CTREQ(NSN),
(100*STD.CTREQ(NSN)/MEAN.CTREQ(NSN)) TUS

LOOP

IF NORMAL.OPT=.TRUE
'' THEN entire CTREQ.mat with normally distributed random values
MONTH1=1
ELSE ''keep actual CTREQ data for 1st MAX.MONTHS & rest random values
MONTH1=MAX.MONTH + 1

ALWAYS

''Initialize CTREQ.MAT
FOR NSN=1 TO MAX.NSN
FOR MONTH=MONTH1 TO MAX.CTREQ.DIM DO

IF STD.CTREQ(NSN) > 0
'' THEN draw next random CTREQ from normal distribution
CTREQ.MAT(NSN,MONT, .TOTAL)=
NORMAL.F(MEAN.CTREQ(NSN),STD.CTREQ(NSN),8)
ELSE '' STD = 0 so no variance and use the mean
CTREQ.MAT(NSN,MONT, .TOTAL) = MEAN.CTREQ(NSN)

ALWAYS
IF CTREQ.MAT(NSN,MONT, .TOTAL) < 1
'' THEN to avoid div errors & have forecast not = actual, set
CTREQ.MAT(NSN,MONT, .TOTAL) = 1 ''*** ASSUMPTION ***

ALWAYS
LOOP

IF TRACE7=.TRUE
FOR NSN=1 TO MAX.NSN DO

PRINT 4 LINES WITH NSN, TIME.V, AT.MONT,
MOD.F((TRUNC.F(TME.V/DPM)+1),CTREQ.PERIOD) TUS

BEGIN REPORT PRINTING
FOR MONTH = 1 TO MAX.CTREQ.DIM IN GROUPS OF 12
PRINT 1 LINE WITH A GROUP OF CTREQ.MAT(NSN,MONT, .TOTAL)
FIELDS THUS

END ''REPORT
LOOP
ALWAYS
UNTIL TIME.V > END.OF.SIMULATION DO
WAIT (CTREQ.PERIOD *.DPM) DAYS
FOR NSN = 1 TO MAX.NSN DO

throw 1st months away and move up last in months to beginning
FOR MONTH = 1 TO (MAX.CTREQ.DIM - CTREQ.PERIOD)
CTREQ.MAT(NSN,MONT, .TOTAL)=

2-59
CTREQ.MAT(NSN,(MONTH+CTREQ.PERIOD),.TOTAL)  
'' generate new CTREQs for months at end of matrix  
FOR MONTH = (MAX.CTREQ.DIM-CTREQ.PERIOD+1) TO MAX.CTREQ.DIM DO  
IF STD.CTREQ(NSN) > 0  
'' THEN draw next random CTREQ from normal distribution  
CTREQ.MAT(NSN,MONTH,.TOTAL) = NORMAL.F(MEAN.CTREQ(NSN),STD.CTREQ(NSN),8)  
ELSE '' STD = 0 so no variance and use the mean  
CTREQ.MAT(NSN,MONTH,.TOTAL) = MEAN.CTREQ(NSN)  
ALWAYS  
IF CTREQ.MAT(NSN,MONTH,.TOTAL) < 1  
'' THEN avoid divid errors & have forecast not = actual, set  
CTREQ.MAT(NSN,MONTH,.TOTAL) = 1 ''*** ASSUMPTION ***  
ALWAYS  
LOOP  
LOOP  
LOOP ''of ur.til  
END ''UPDATE.CTREQ.MAT  

PROCESS WARMUP.RESET  
'' This process resets all appropriate statistics back to zero  
'' once the initial warmup period is over and the transient effects  
'' have apparently been washed out of simulation. This is so the  
'' final statistics at end of simulation are not effected by warmup  
'' period.  

WAIT WARMUP.PERIOD DAYS  
'' CALL PRINT.ATEND  
PRINT 5 LINES WITH TIME.V, AT.MONTH THUS  

$\text{END OF WARMUP PERIOD: RESET VARIABLES: TIME.V ** AT.MONTH **}$  

FOR EACH DEMAND.BO DO  
FOR EACH NSN.DETAIL DO  
RESET TOTALS OF REQ.SIZE(DEMAND.BO,NSN.DETAIL)  
RESET TOTALS OF REQ.BO(DEMAND.BO,NSN.DETAIL)  
RESET TOTALS OF UNIT.BO(DEMAND.BO,NSN.DETAIL)  
RESET TOTALS OF REQ.INTERVAL(DEMAND.BO,NSN.DETAIL)  
LOOP  
LOOP  

'' prepare for divide by 0 error after reset  
FOR EACH DEMAND.BO DO  
FOR EACH NSN.DETAIL DO  
SUM.REQ.BO(DEMAND.BO,NSN.DETAIL) = .MINVAL  
SUM.UNIT.BO(DEMAND.BO,NSN.DETAIL) = .MINVAL  
SUM.REQ.SIZE(DEMAND.BO,NSN.DETAIL) = .MINVAL  
NO.REQ.SIZE(DEMAND.BO,NSN.DETAIL) = .MINVAL  
LOOP
LOOP

FOR EACH NSN.ATTRIBUTES DO
    RESET TOTALS OF FORECAST.MTH(NSN.ATTRIBUTES)
    RESET TOTALS OF STOCK(NSN.ATTRIBUTES)
    RESET TOTALS OF ONORDER(NSN.ATTRIBUTES)
END LOOP

RESET TOTALS OF MONTHLY.MAPE, SIM.PLT.DAY
ORDER. NUMBER=0
AT. MONTH=0

' Call PRINT.ATEND
END ''WARMUP.RESET

ROUTINE XYZ.PLTS
' This routine determines which NSN are X, Y, or Z items, and based
' on delivery method 1 to 4, the PLTs for each NSN.

DEFINE NSN AS INTEGER VARIABLE
DEFINE PERCENT.PCP, PGC.PCP, DVQD AS REAL VARIABLE
RESERVE XYZ.SUM(*) AS 3
PRINT 5 LINE

------------------------------ SIMULATION DATA DESCRIPTION ------------------------------
------------------------------ SIMULATION DATA DESCRIPTION ------------------------------
------------------------------ SIMULATION DATA DESCRIPTION ------------------------------

FOR NSN=1 TO MAX.NSN DO
    DVQD=TRUNC.F(AVE.FORECAST(NSN)*3)*COST
    IF DVQD <= M1
       ' THEN DVQD set for a 36 month procurement cycle
       PCP.MONTH(NSN)=36
    ELSE
       IF (DVQD > M1) AND (DVQD <= M2)
          ' THEN between M1 & M2 so use Wilson Lot Size equation
          PRODUCE CYCLE (MONTHS)= EOQ / MONTHLY DEMAND
          PCP.MONTH(NSN)=TRUNC.F((3*T)*(DVQD**(-0.5)))
       ELSE ' greater than M2 or use 6 month PCP
          PCP.MONTH(NSN)=6
    ALWAYS
END LOOP

FOR NSN=1 TO MAX.NSN ' sum to use as average order quantity
    PGC.PCP=PGC.PCP + (AVE.FORECAST(NSN)*PCP.MONTH(NSN))
END LOOP

PERCENT.PCP = (100 * (AVE.FORECAST(NSN)*PCP.MONTH(NSN))/PGC.PCP)
IF (PERCENT.PCP >= X.PERCENT)
   ' THEN X item
   XYZ.MONTH(NSN)=1
ELSE
  IF (PERCENT.PCP <= Z.PERCENT)
    'THEN Z item
    XYZ.MONTH(NSN)=3
  ELSE 'Y item
    XYZ.MONTH(NSN)=2
  ALWAYS
  ALWAYS
XYZ.SUM(XYZ.MONTH(NSN)) = XYZ.SUM(XYZ.MONTH(NSN)) + PERCENT.PCP

SELECT CASE DELIVERY.OPT
CASE 1 '********** METHOD 1 DELIVERY OPTION **********
  IF (PERCENT.PCP >= X.PERCENT)
    'THEN X item
    PLT.DAY(NSN)= FIRST.DELIVERY +
      ((1/3)*MAX.DELIVERIES * .DPM)
  ELSE
    IF (PERCENT.PCP <= 2.PERCENT)
      'THEN Z item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        ((5/6)*MAX.DELIVERIES * .DPM)
    ELSE 'Y item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        ((2/3)*MAX.DELIVERIES * .DPM)
  ALWAYS
  ALWAYS
' store for use in matrix scheduling since method 1 uses
' clumping not incremental (w/ 1 of 3 vector of percents)
XYZ.MONTH(NSN)=PERCENT.PCP

CASE 2 '********** METHOD 2 DELIVERY OPTION **********
  IF (PERCENT.PCP >= X.PERCENT)
    'THEN X item
    PLT.DAY(NSN)= FIRST.DELIVERY +
      ((1/2)*MAX.DELIVERIES * .DPM)
  ELSE
    IF (PERCENT.PCP <= 2.PERCENT)
      'THEN Z item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        ((5/6)*MAX.DELIVERIES * .DPM)
    ELSE 'Y item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        ((2/3)*MAX.DELIVERIES * .DPM)
  ALWAYS
  ALWAYS

CASE 3 '********** METHOD 3 DELIVERY OPTION **********
  IF (PERCENT.PCP >= X.PERCENT)
    'THEN X item
    PLT.DAY(NSN)= FIRST.DELIVERY +
      ((1/2)*MAX.DELIVERIES * .DPM)
  ELSE
    IF (PERCENT.PCP <= 2.PERCENT)
      'THEN Z item
      PLT.DAY(NSN)= FIRST.DELIVERY +

2-62
ELSE 'Y item
PLT.DAY(NSN) = FIRST.DELIVERY +
(2/3)*MAX.DELIVERIES * DPM)

ALWAYS
ALWAYS
CASE 4 "********** METHOD 4 DELIVERY OPTION **********
IF (PERCENT.PCP >= X.PERCENT)
' THEN X item
PLT.DAY(NSN) = FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * DPM)
ELSE
IF (PERCENT.PCP <= Z.PERCENT)
' THEN Z item
PLT.DAY(NSN) = FIRST.DELIVERY +
((5/6)*MAX.DELIVERIES * DPM)
ELSE 'Y item
PLT.DAY(NSN) = FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * DPM)

ALWAYS
ALWAYS
ENDSELECT
LOOP
' USE UNIT 6 FOR OUTPUT
IF TRACE22=.TRUE.
PRINT 2 LINES WITH PGC.PCP, XYZ.SUM(1), XYZ.SUM(2), XYZ.SUM(3),
FIRST.DELIVERY THUS
PGC TOTAL PCP ** X % **, Y % **, Z % ** 1ST DEL **
NSN PCP PLT PERCENT.PCP XYZ.MONTH AVE.FORECAST FOR DVQD

FOR NSN=1 TO MAX.NSN DO
PRINT 1 LINE WITH NSN, PCP.MONTH(NSN), PLT.DAY(NSN),
(100 * (AVE.FORECAST(NSN)*PCP.MONTH(NSN))/PGC.PCP),
XYZ.MONTH(NSN), AVE.FORECAST(NSN),
(TRUNC.F(AVE.FORECAST(NSN)*3)*COST) THUS
** ** ** ** ** ** ** **
LOOP
ALWAYS
END 'routine XYZ.PLTS
CHAPTER 3
THE CAPTURE PROGRAM

The DLA data capture program is displayed in this chapter. The capture program reads a downloaded Standard Automated Materiel Management System (SAMMS) file, extracts the information required by the simulation, and stores it on the PC hard disk. The program resides in the subdirectory C:\SIM\DLADATA on the PC's hard disk. The chapter contains four sections: the outline flow of the program, the list of all procedures, a short description of each procedure, and the program source code.

FLOW OUTLINE OF PROGRAM

PREAMBLE

MAIN

INITIAL.NEW.NSN
INITIAL.NEW.PGC
PRINT.DEMAND.INFO
PRINT.LINE.SEARCH
PRINT.OUTPUT.FILE
READ.HEADER.PAGE
STORE.NSN.DATA
TRLR.E.READ.DEMAND
TRLR.U.READ.RQMT

LISTING OF PROCEDURES

PREAMBLE

MAIN

INITIAL.NEW.NSN
INITIAL.NEW.PGC
PRINT.DEMAND.INFO
PRINT.LINE.SEARCH
PRINT.OUTPUT.FILE
READ.HEADER.PAGE
STORE.NSN.DATA
TRLR.E.READ.DEMAND
TRLR.U.READ.RQMT

DESCRIPTION OF PROCEDURES

PREAMBLE

'This program reads the Special Supply Control File raw data copied
' from tape and extracts the required information to run the C&T
'Simulation Model. The program then saves this data in the output file that will be directly read by the model.

INPUT FILE: DLATAPE.XX
OUTPUT FILE: DLAINPUT.DAT

MAIN
The main routine has the basic outline for entire model below are key assumptions: (see PRINT.OUTPUT.FILE for general description)

THIS PROGRAM CAPTURES THE DATA FROM THE SPECIAL SUPPLY CONTROL FILE (SSCF) REPORT. THE SSCF REPORT FILE MUST BE STORED IN:
C:\SIM\DLADATA\SSCFTAPE
FOR THE CAPTURE PROGRAM TO RUN PROPERLY
THE OUTPUT OF THIS PROGRAM GOES DIRECTLY TO THE SIMULATION MODEL DIRECTORY, TO BE INCORPORATED AUTOMATICALLY WHEN THE SIMULATION RUNS. THAT OUTPUT FILE IS:
C:\SIM\DLA\SSCFSIM.DAT

IMPORTANT NOTE: <<<
RUNNING THIS PROGRAM WILL OVERWRITE THE EXISTING DATA IN THE SSCFSIM.DAT FILE WITH NEW DATA. IF YOU HAVE NOT BACKED UP THE CURRENT CONTENTS OF SSCFSIM.DAT OR WANT TO READ CHAPTER 2 OF THE DOCUMENTATION FOR FURTHER EXPLANATION
PRESS CTRL-C (to stop run)
-----------------------------------------------

OPTIONS OF MODEL ***********************************
Make sure you choose either option before running model
TEST.OPT=.TRUE ' 'If true will use test input and output data files
' ' else will use actual, full blown data files
SIMOUTPUT.OPT=.TRUE ' 'If true produces output file for simulation;
' ' false produces data analysis output

INITIAL.NEW.NSN
This routine reinitializes certain cumulative counters after each NSN has been completely processed

INITIAL.NEW.PGC
This routine initializes all cumulative variables after before each NSN is read.

PRINT.DEMAND.INFO
This routine prints the demand info from trailer E and other comparison stats: PGM vs QFD, for the data analysis report. Also trace 5 and trace 7.

PRINT.LINE.SEARCH
This routine simply prints the char. string in the intermediate searches for the next specific line location for trace 4

PRINT.OUTPUT.FILE
This routine prints the actual data required by the C&T model
Below are the NSN required input for the C&T model. Most values come directly off the Special Supply Control File (SSCF) report and are given the identical labels as appears in the report.
1) Most variables come directly from the Header page and captured by READ.HEADER.PAGE routine.

2) ARS (average requisition size) is total demands/total frequency and calculated in routine TRLR.E.READ.DEMAND from the Trailer E of SSCF data.

3) For Program Oriented Items (POI) additional data are captured: PER.RTC.DEMAND (the percent of RTC PIC requirement demands to total NSN demand). MAX.MONTH is the number of actual months of Program Requirement forecast since forecast can start at any quarter in the current fiscal year & then go to additional years. CTREQ.MAT are the actual monthly C&T Requirements forecasts sum across all PICs for each NSN. ALL of this info is obtained in routine TRLR.U.READ.RQMT (from Trailer U of SSCF)

4) MAX.NSN is the total number of NSNs in the PGC calculated in Main

READ.HEADER.PAGE
This routine reads the header page on from the tape file of the special supply control file

ASSUMPTIONS:
1) COST & STAND PRICES, SYSTEM SS & DS, MAD, ASFE ARE < 10 MILLION
2) QFD, NEWQFD, (12 MTH, PAST MTH, PAST QTR) PGM RQMT ARE < 100 MILLION

STORE.NSN.DATA
Stores the NSN data for later final printing once Max.Nsn is known

TRLR.E.READ.DEMAND
This routine reads the trailer E that contains historic demands and their frequency for the last 4 quarters. The routine sums each quarters demands separately. It also takes the total demands and divides by the total frequency to get average requisition size. Returns are not part of the calculations and nonrecurring, high demand items have only applicable percent in the calculation of total demands and ARS.

TRLR.U.READ.RQMT
This routine reads the 3 years of monthly C&T Program Requirement data from trailer U of the Special SCF report for POI items. It calculates the number of months of requirements or MAX.MONTH. Finally, it calculates the percent of recruit training center demand (PER.RTC.DEMAND) by dividing the recruit PICs (last 2 letters of PIC = AA, AW, GB) requirement over the total NSNs requirement from all the PICS.

SOURCE CODE

PREAMBLE
''This program reads the Special Supply Control File raw data copied from tape and extracts the required information to run the C&T Simulation Model. The program then saves this data in the output file that will be directly read by the model.
'' INPUT FILE: DLATAPE.XX
'' OUTPUT FILE: DLAINPUT.DAT
NORMALLY MODE IS UNDEFINED

PERMANENT ENTITIES "stored variables for final print
EVERY NSN.ATTRIBUTES HAS
A NSN.NO, '' NIIN number
A PLT.DAY, '' production lead time in days
A VIP.ITEM, '' VIP items reviewed every month vs quarter
A SAFETY.MONTH, '' fixed safety level in months
A QFDP, '' quarterly forecasts
A NEW.QFD, '' New QFD
A MADP, 'MAD, mean absolute deviation in forecast demands
A OWMRPP, 'OWMRP war reserves
A ALPHAP, 'alpha factor
A ARSP, 'ARS or average requisition size for nsn
A PER_RTC.DEMANDP ' percent of RTC demand vs total demand

DEFINE VIP.ITEM, PLT.DAY, OWMRPP
AS INTEGER VARIABLES
DEFINE QFDP, NEW.QFD, MADP, ALPHAP, ARSP, PER_RTC.DEMANDP,
SAFETY.MONTH AS REAL VARIABLES
DEFINE NSN.NO AS TEXT VARIABLE

DEFINE NIIN.T, ITEM.NAME.T, PROC.CYC.T, DVC.T, ICC.T, VIP.IND.T,
OT.IND.T, TRLR.T, PIC.T,
NAME, ICC ''PGC stored variable for final print

AS TEXT VARIABLES

DEFINE FSC, ADM.LT, PRO.LT, TSCC, PROC.CYCLE,
SL.E.FACTOR, PROC.GR.CD, QFD, NEW.ITEM.QFD, PGM.RQMT.12.MTH,
PGM.RQMT.PAST.MTH, PGM.RQMT.PAST.QTR, OWMRP, TRACE1, TRACE2,
TRACE3, TRACE4, TRACE5, TRACE6, TRACE7, TRACE8, MONTH, YEAR,
MAX.MONTH, MAX.NSN, PGC.COUNT, OLD.PGC, START.MONTH,
FSCP, ALT.DAY, PGC.NO, MAX.MONTHP 'PGC stored var. for final print

AS INTEGER VARIABLES

DEFINE FIX.SAFE, ANRDP, ALPHA, RETURNS, SUM.FREQ, PER_RTC.DEMAND,
PGC.QD, PGC.QRQMT

AS REAL VARIABLES

DEFINE COST.PRICE, STANDARD.PRICE, SYSTEM_SS, SYSTEM_DS, MAD,
ALG.SUM.FE, ARS,
COST
AS DOUBLE VARIABLES

DEFINE SUM.QD AS AN REAL, 1-DIMENSIONAL ARRAY
DEFINE CTREQ.MAT AS AN INTEGER, 1-DIMENSIONAL ARRAY
DEFINE CTREQ.MAT.HOLD AS AN INTEGER, 2-DIMENSIONAL ARRAY 'holds PGC
DEFINE .TRUE TO MEAN 1
DEFINE .FALSE TO MEAN 0
DEFINE .MAX.DIM TO MEAN 200
DEFINE ANS, SIMOUTPUT.OPT, TEST.OPT AS INTEGER VARIABLE
ANS=1
PRINT 21 LINES THUS
======================================================================
THIS PROGRAM CAPTURES THE DATA FROM THE SPECIAL SUPPLY CONTROL
FILE (SSCF) REPORT. THE SSCF REPORT FILE MUST BE STORED IN:
C:\SIM\DLADATA\SSCFTAPE
FOR THE CAPTURE PROGRAM TO RUN PROPERLY
THE OUTPUT OF THIS PROGRAM GOES DIRECTLY TO THE SIMULATION
MODEL DIRECTORY, TO BE INCORPORATED AUTOMATICALLY WHEN THE
SIMULATION RUNS. THAT OUTPUT FILE IS:
C:\SIM\DLA\SSCFSIM.DAT

====> IMPORTANT NOTE: <=====

RUNNING THIS PROGRAM WILL OVERWRITE THE EXISTING DATA IN THE
SSCFSIM.DAT FILE WITH NEW DATA. IF YOU HAVE NOT BACKED UP THE
CURRENT CONTENTS OF SSCFSIM.DAT OR WANT TO READ CHAPTER 2 OF
THE DOCUMENTATION FOR FURTHER EXPLANATION

PRESS: CTRL-C
(TO STOP THIS CAPTURE PROGRAM )
======================================================================
PRINT 1 LINE THUS
ENTER ANY NUMBER TO CONTINUE RUN
READ ANS

'**************************************************************
'****************** OPTIONS OF MODEL ***********************
' Make sure you choose either option before running model
TEST.OPT=.TRUE '"If true will use test input and output data files
'' else will use actual, full blown data files
SIMOUTPUT.OPT=.TRUE '"If true produces output file for simulation;
'' false produces data analysis output

TRACE1=.FALSE '"Print header page output
TRACE2=.FALSE '"Print Trailer E, historic demand output
TRACE3=.FALSE '"Print Trailer E summary results
TRACE4=.FALSE '"Print Trailer U, Program Requirement Matrix
TRACE5=.FALSE '"Print Trailer U, summary results
TRACE6=.FALSE '"Print intermediate searching between lines
'below traces set later
TRACE7=.FALSE '"Prints the useful stats not needed for simulation
TRACE8=.FALSE '"Prints output file for simulation

3-5
'************* INPUT FILES *************

IF TEST.OPT=.TRUE
' THEN Sample set of NSNs with 3 PGCs
OPEN UNIT 1 FOR INPUT, FILE NAME IS "C:SIM\DLADATA\DLATAPE.QFD"
ELSE '' ** Full SSCF tape of 10 megs
' OPEN UNIT 1 FOR INPUT, FILE NAME IS "D:\DLADATA\SSCFTAPE"
OPEN UNIT 1 FOR INPUT, FILE NAME IS "C:\SIM\DLADATA\SSCFTAPE"

ALWAYS
USE UNIT 1 FOR INPUT

IF SIMOUTPUT.OPT = .TRUE
' THEN produces file for simulation model run (full or sample)
IF TEST.OPT = .TRUE
OPEN UNIT 2 FOR OUTPUT, FILE NAME IS "C:\SIM\DLADATA\IN3PGC.DAT"
ELSE
OPEN UNIT 2 FOR OUTPUT, FILE NAME IS "D:\DLADATA\SSCFSIM.DAT"
OPEN UNIT 2 FOR OUTPUT, FILE NAME IS "C:\SIM\DLA\SSCFSIM.DAT"

ALWAYS
TRACE7=.FALSE
TRACE8=.TRUE
ELSE '' produce data analysis file
OPEN UNIT 2 FOR OUTPUT, FILE NAME IS "D:\DLADATA\ANALYSIS.DAT"
TRACE7=.TRUE
TRACE8=.FALSE

ALWAYS

USE UNIT 2 FOR OUTPUT
RESERVE SUM.QD(*) AS 4
RESERVE CREQ.MAT (*,*) AS 36
RESERVE CREQ.MAT.HOLD(*,*) AS .MAX.DIM BY 36
CREATE EVERY NSN.ATTRIBUTES (.MAX.DIM)
OLD.PGC=99999

WHILE (EOF.V=0) DO
' WHILE (MAX.NSN<2) AND (EOF.V=0) DO
CALL READ.HEADER.PAGE
IF (OLD.PGC not equal to PROC.GR.CD)
' THEN have new PGC so
IF OLD.PGC NE 99999
' THEN not the first PGC
CALL PRINT.OUTPUT.FILE
OLD.PGC = PROC.GR.CD
CALL INITIAL.NEW.PGC
ELSE '' first PGC so set to another default
OLD.PGC = PROC.GR.CD

ALWAYS

ALWAYS
CALL INITIAL.NEW.NSN

WHILE TRLR.T = "0TYPE TRLR E" DO
CALL TRLR.E.READ.DEMAND
LOOP
MAX.NSN = MAX.NSN + 1
CALL PRINT.DEMAND.INFO '' calculates ARS when all QFD read
IF ICC.T = "P"
' THEN POI item so read requirements trailer
CALL TRLR.U.READ.RQMT
ALWAYS
CALL STORE.NSN.DATA
LOOP'' while loop for each NSN
CALL PRINT.OUTPUT.FILE ''for last PGC
END ''main

ROUTINE INITIAL.NEW.NSN
'' This routine reinitializes certain cumulative counters after
'' each NSN has been completely processed
DEFINE ROW AS AN INTEGER VARIABLE
FOR ROW =1 TO 4
SUM.QD(ROW)=0
SUM.FREQ=0
RETURNS=0
FOR ROW = 1 TO 36
CTREQ.MAT(ROW) =0
END ''routine INITIAL.NEW.NSN

ROUTINE INITIAL.NEW.PGC
'' This routine initializes all cumulative variables after before
'' each NSN is read.
MAX.NSN=0
PGC.QD=0
PGC.QRQMT=0
END ''routine INITIAL.NEW.PGC

ROUTINE PRINT.DEMAND.INFO
'' This routine prints the demand info from trailer E and other
'' comparison stats: PGM vs QFD,
DEFINE RQQD, SD.MAD, AVE.QD, SD.QD AS REAL VARIABLES
DEFINE ROW, CORREL AS INTEGER VARIABLES
FOR ROW = 1 TO 4 DO
COMPUTE
AVE.QD AS THE MEAN AND
SD.QD AS THE STD.DEV OF SUM.QD(ROW)
LOOP
ARS= (4 * AVE.QD)/SUM.FREQ
'' to convert MAD to quarterly value and into stand. deviation
IF VIP.IND.T ="Y"
'' THEN monthly value and actual std dev. for demand is quarterly
SD.MAD=MAD*1.25*(7/4) ''mad * 1.25 * sqrt(3)
ELSE ''already quarterly data just convert to std. dev.
SD.MAD = MAD * 1.25
ALWAYS
PGC.QD = PGC.QD + AVE.QD
PGC.QRQMT = PGC.QRQMT + (PGM.RQMT.12.MTH/4)

IF TRACE5 = .TRUE
PRINT 6 LINES WITH SUM.QD(1), SUM.QD(2), SUM.QD(3), SUM.QD(4), SUM.FREQ, ARS, MAD, RETURNS, PGC.QD, PGC.QRQMT, SD.MAD, SD.QD, PGM.RQMT.12.MTH/4, AVE.QD, (PGM.RQMT.12.MTH/4 - AVE.QD), ALG.SUM.FE
THUS
SUMS:

<table>
<thead>
<tr>
<th>QD1</th>
<th>QD2</th>
<th>QD3</th>
<th>QD4</th>
<th>FREQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
<td>**</td>
</tr>
</tbody>
</table>

ARS MAD RETURNS Q PGC QD PGC Q.RQMT
** ** ** ** **
SD.MAD SD.QD RQMT QD AVE QD F-A ASFE
** ** ** ** ** **
ALWAYS

IF TRACE7 = .TRUE
RQQD = (PGM.RQMT.12.MTH/4)
IF ((RQQD > AVE.QD) AND (ALG.SUM.FE > 0)) OR
((RQQD < AVE.QD) AND (ALG.SUM.FE < 0))
' THEN have a correlation between forecast error and past year
CORREL = 1
ELSE
CORREL = 0
ALWAYS

IF MAX.NSN = 1
PRINT 1 LINE
THUS
NSN %RETURN %SD/RQMT %ACT/RQM ASFE RQMT QD %SD QD/MAD CORREL
ALWAYS

PRINT 1 LINES WITH MAX.NSN, (100*RETURNS/AVE.QD), 100*SD.MAD/RQQD,
100*(AVE.QD/RQQD), ALG.SUM.FE, RQQD, 100*SD.QD/SD.MAD, CORREL
THUS
** ** ** ** ** ** ** **
ALWAYS

END ' 'routine PRINT.DEMAND.INFO

ROUTINE PRINT.LINE.SEARCH GIVEN INTER.STRING
'This routine simply prints the char. string in the intermediate
' searches for the next specific line location
DEFINE INTER.STRING AS TEXT VARIABLE

IF TRACE4 = .TRUE
PRINT 1 LINE WITH INTER.STRING

***************
ALWAYS
END ' 'routine PRINT.LINE.SEARCH

3-8
This routine prints the actual data required by the C&T model. Below are the NSN required input for the CST model. Most values come directly off the Special Supply Control File (SSCF) report and are given the identical labels as appears in the report.

1) Most variables come directly from the Header page and captured by READ.HEADER.PAGE routine.
2) ARS (average requisition size) is total demands/total frequency and calculated in routine TRLR.E.READ.DEMAND from the Trailer E of SSCF data.
3) For Program Oriented Items (POI) additional data are captured:
   PER.RTC.DEMAND (the percent of RTC PIC requirement demands to total NSN demand). MAX.MONTH is the number of actual months of Program Requirement forecast since forecast can start at any quarter in the current fiscal year & then go to additional years.
   CTREQ.MAT are the actual monthly C&T Requirements forecasts sum across all PICs for each NSN. ALL of this info is obtained in routine TRLR.U.READ.RQMT (from Trailer U of SSCF)
4) MAX.NSN is the total number of NSNs in the PGC calculated in Main

```
DEFINE NSN, COL AS INTEGER VARIABLE

IF TRACES=.TRUE
LINES.V=0
PRINT 4 LINE WITH PGC.NO, MAX.NSN THUS

-------------------NEW PROCUREMENT GROUPING CODE -------------------
PROC.GR.CD ** MAX.NSN **

PRINT 2 LINES WITH NAME, FSCP, ICC, ALT.DAY, COST, MAX.MONTHP
THUS
ITEM NAME FSCP ICC ADM.LT STANDARD.PRICE MAX.MONTHP
************** ** * ** ** ** **

PRINT 1 LINE THUS
NSN NIIN PRO.LT VIP(L=Y) FIX.SAFE QFD
FOR NSN = 1 TO MAX.NSN
PRINT 1 LINE WITH NSN, NSN.NO(NSN), PLT.DAY(NSN), VIP.ITEM(NSN), SAFETY.MONTH(NSN), QFDP(NSN) THUS
** ********** ** * ** ** ** **

PRINT 2 LINES THUS
NSN MAD OWRMRP ALPHA ARS PER.RTC.DEMAND
FOR NSN =1 TO MAX.NSN
PRINT 1 LINE WITH
   NSN, MADP(NSN), OWRMRPP(NSN), ALPHAP(NSN), ARSP(NSN),
   PER.RTC.DEMANDP(NSN) THUS
** ** ** ** ** ** ** ** ** **
```
IF ICC="P"
 'THEN POI item do requirements
 FOR NSN = 1 TO MAX.NSN DO

 PRINT 2 LINE WITH NSN, NSN.NO(NSN), MAX.MONTH THUS
 NSN ** NIIN ******* CT REQUIREMENT MATRIX FOR ** MONTHS ======
 MONTHS: 1 ------- 2 ------- 3 ------- 4 ------- 5 ------- 6
 BEGIN REPORT PRINTING
 FOR COL= START.MONTH TO 36 IN GROUPS OF 6
 PRINT 1 LINE WITH A GROUP OF CTREQ.MAT.HOLD(NSN,COL) FIELDS
 ** ** ** ** ** **
 END 'REPORT LOOP
 ELSE
 PGC.QRQMT=0.0000000001
 ALWAYS ALWAYS

 PRINT 3 LINE WITH OLD.PGC, PGC.QRQMT, PGC.QD, 100*PGC.QD/PGC.QRQMT
 THUS
 END.OF.PGC ** PGC QRQMT ** PGC QD ** %A/R **,**
==================================================================

END 'routine PRINT.OUTPUT.FILE

ROUTINE READ.HEADER.PAGE
 'This routine reads the header page on from the tape file of the
 ' special supply control file
 ' ASSUMPTIONS:
 '1)COST & STAND PRICES, SYSTEM SS & DS, MAD, ASFE ARE < 10 MILLION
 '2)QFD, NEWQFD, (12 MTH, PAST MTH, PAST QTR) PGM RQMT ARE < 100 MILLION

 DEFINE HOLD.TEXT, ASFE.SIGN AS TEXT VARIABLES

 UNTIL (TRLR.T="GTYP NS") DO
 READ TRLR.T AS //, B 1, T 12
 CALL PRINT.LINE.SEARCH GIVEN TRLR.T
 LOOP
 ' ==== find 1st line of input
 READ FSC, NIIN.T AS //, B 6, I 4, B 12, T 9, /

 ' ====== find 2nd line of input
 UNTIL HOLD.TEXT=" ITEM NAME" DO 'finds next read record
 READ HOLD.TEXT AS T 12,/
 CALL PRINT.LINE.SEARCH GIVEN HOLD.TEXT
 LOOP

 READ ITEM.NAME.T, PROC.CYC.T, DVC.T, ICC.T, ADM.LT, PRO.LT
 AS B 2,T 17, B 36,T 1, B 55,T 1, B 60,T 1, B 73,I 3, B 78,I 3
READ VIP.IND.T, OT.IND.T, TSCC, PROC.CYCLE
    AS B 100,T 1,   B 104,T 3,   B 110,I 3,   B 123,I 3 , /

'=== find 3rd line of input data
UNTIL HOLD.TEXT="SAFE" DO ''finds next read record
    READ HOLD.TEXT AS B 116, T 4, /
    CALL PRINT.LINE.SEARCH GIVEN HOLD.TEXT
LOOP
READ FIX.SAFE, SL.E.FACTOR
    AS B 116,D(4,1),   B 123,I 1, /

'=== Find 4th line of input data ===
READ ANRDP, PROC.GR.CD, COST.PRICE, STANDARD.PRICE, ALPHA
    AS //,   B 35,D(4,1),B 41,I 5,   B 88,D(10,2),B 100,D(10,2),B 113,D(4,1), /

'=== Find 5th line of input data ======
READ QFD, NEW.ITEM.QFD, SYSTEM.SS, SYSTEM.DS
    AS //,   B 23,I 8,   B 33,I 8,   B 73,D(9,1),   B 84,D(9,1)
READ MAD,   ALG.SUM.FE,   ASFE.SIGN
    AS   B 95,D(9,1),   B 106,D(9,1),   B 115, T 1, /

IF ASFE.SIGN ="-" 
' THEN ASFE is negative
   ALG.SUM.FE=--ALG.SUM.FE
ALWAYS

'=== Find 6th line of input data ========
READ PGM.RQMT.12.MTH, PGM.RQMT.PAST.MTH, PGM.RQMT.PAST.QTR, OWRMRP
    AS //,   B 53,I 8,   B 73, I 8,   B 83, I 8,       B 93, I 8,/

IF TRACE1=.TRUE
PRINT 14 LINES WITH FSC, NIIN.T,ITEM.NAME.T, PROC.CYC.T, DVC.T, ICC.T,
       ADM.LT, PRO.LT, VIP.IND.T, OT.IND.T, TSCC, PROC.CYCLE, FIX.SAFE,
       SL.E.FACTOR, ANRDP, PROC.GR.CD, COST.PRICE, STANDARD.PRICE, ALPHA,
       QFD, NEW.ITEM.QFD, SYSTEM.SS, SYSTEM.DS, MAD, ALG.SUM.FE,
       PGM.RQMT.12.MTH, PGM.RQMT.PAST.MTH, PGM.RQMT.PAST.QTR, OWRMRP Thus

FSC= ** NIIN *******
ITEM.NAME.T, PROC.CYC.T DVC ICC ADM.LT, PRO.LT
**************   *   *   *   **   **
VIP.IND.T, OT.IND.T, TSCC PROC.CYCLE FIX.SAFE SL.E
** *** ** ** *** **
ANRDP PROC.GR.CD COST.PRICE STANDARD.PRICE ALPHA
** ** ** ** ** ** **
QFD NEW.ITEM.QFD SYSTEM.SS, SYSTEM.DS
** ** ** ** ** **
MAD ALG.SUM.FE
** ** **
PGM.RQMT.12.MTH PGM.RQMT.PAST.MTH PGM.RQMT.PAST.QTR OWRMRP
** ** **
ALWAYS

'Set record pointer at trailer E
UNTIL (TRLR.T="O TYPE TRLR E") DO
READ TRLR.T AS /,T 12
CALL PRINT.LINE.SEARCH GIVEN TRLR.T
LOOP
END "routine READ.HEADER.PAGE"

ROUTINE STORE.NSN.DATA
" Stores the NSN data for later printing once Max.Nsn is known
DEFINE COL AS INTEGER VARIABLE

IF MAX.NSN = 1
" THEN store the NSN data that is constant for entire PGC
   NAME =ITEM.NAME.T
   FSCP =FSC
   ICC=ICC.T
   ALT.DAY =ADM.LT
   PGC.NO =PROC.GR.CD
   COST =STANDARD PRICE
   MAX.MONTHP =MAX.MONTH
ALWAYS
" NSN specific data (i.e. changes with each NSN)
NSN.NO(MAX.NSN)=NIIN.T
PLT.DAY(MAX.NSN)=PRO.LT
SAFETY.MONTH(MAX.NSN)=FIX.SAFE
IF QFD < NEW.ITEM.QFD
" THEN use the new item QFD since QFD not old enough to be correct
   QFDP(MAX.NSN)=NEW.ITEM.QFD
ELSE " QFD is OK
   QFDP(MAX.NSN)=QFD
ALWAYS
MADP(MAX.NSN)=MAD
OWRMRPP(MAX.NSN)=OWRMRP
ALPHAP(MAX.NSN)=ALPHA
ARSP(MAX.NSN)=ARS

IF VIP.IND.T = "Y"
" THEN VIP item and requirements done monthly set to integer true val.
   VIP.ITEM(MAX.NSN) = .TRUE
ELSE
   VIP.ITEM(MAX.NSN) = .FALSE
ALWAYS
IF ICC.T = "P"
" THEN POI item store requirement info
   PER.RTC.DEMANDP(MAX.NSN)=PER.RTC.DEMAND
   FOR COL= START.MONTH TO 36
     CREQ.MAT.HOLD(MAX.NSN,COL)=CTREQ.MAT(COL)
   ELSE
   PER.RTC.DEMANDP(MAX.NSN)=0
   MAX.MONTHP=0
ALWAYS
END "routine STORE.NSN.DATA"
ROUTINE TRLR.E.READ.DEMAND

"This routine reads the trailer E that contains historic demands and their frequency for the last 4 quarters. The routine sums each quarters demands separately. It also takes the total demands and divides by the total frequency to get average requisition size. Returns are not part of the calculations and nonrecurring, high demand items have only applicable percent in the calculation of total demands and ARS.

DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE ROW AS AN INTEGER VARIABLE
DEFINE DMCD.T, NSN.T AS TEXT VARIABLES
DEFINE FREQ, QD AS AN INTEGER, 1-DIMENSIONAL ARRAYS
RESERVE FREQ(*), QD(*) AS 4

"Assumes pointer at TYP TRLR E record left from Header or TRLR E
READ NSN.T AS //,/,/, B 1, T 9
IF NSN.T not equal to NIIN.T
THEN
PRINT 1 LINE WITH NIIN.T, NSN.T THUS
$$ ERROR $$$$ HEADER NSN ******** NOT EQUAL TO TRLR E ********
STOP ""processing
ALWAYS

IF TRACE2=.TRUE
PRINT 2 LINE WITH NSN.T THUS
NSN ******** DMCD QD 1 FREQ 1 QD 2 FREQ 2 QD 3 FREQ 3 QD 4 FREQ
ALWAYS

WHILE NSN.T = NIIN.T DO
READ DMCD.T, QD(1), FREQ(1), QD(2), FREQ(2)
AS B 14,T 1, B 73,I 8, B 81,I 5, B 88,I 8, B 96, I 5
READ QD(3), FREQ(3), QD(4), FREQ(4), NSN.T
AS B 103,I 8, B 111, I 5, B 118,I 8, B 126,I 5, /, B 1,T 9

IF (DMCD.T="N") AND (DVC.T = "H") 'nonrecurring, hi demand chap25
'THEN add applicable percent of nonrecurring demands to total
FOR ROW=1 TO 4 DO
SUM.QD(ROW)=SUM.QD(ROW) + (QD(ROW)*ANRDP)
SUM.FREQ=SUN.FREQ + FREQ(ROW)
LOOP
ELSE
IF (DMCD.T ="T")
'THEN add to return data
FOR ROW=1 TO 4
RETURNS=RETURNS + QD(ROW)/4
ELSE 'add all demands and frequencies
FOR ROW=1 TO 4 DO
SUM.QD(ROW)=SUM.QD(ROW) + QD(ROW)
SUM.FREQ=SUN.FREQ + FREQ(ROW)

3-13
LOOP
ALWAYS
ALWAYS
IF TRACE2=.TRUE
PRINT 1 LINE WITH DMCD.T, QD(1), FREQ(1), QD(2), FREQ(2), QD(3),
FREQ(3), QD(4), FREQ(4) THUS
* ** ** ** ** ** ** **
ALWAYS
LOOP
'' At this point end reading demands in TRLR E so next possible options
'' are another TRLR E, the next NSN header record (OTYP), a TRLR U if
'' POI item, or the end of a file (EOF.V=2) set within loop by ' 2
EOF.V=1
'' so if EOF will stop, not print error & abort
'' Set record pointer to next trailer record
READ TRLR.T AS /, B 1, T 12
UNTIL (TRLR.T = "OTYPE TRLR E") OR (TRLR.T="OTYP NS")
OR (TRLR.T = "OTYPE TRLR U") OR (EOF.V=2)
DO
READ TEST.EOF AS /, B 1, A 1
IF (TEST.EOF=26) OR (EOF.V=2)
'' THEN at end of file (Note a 26 is a ' 2 or DOS EOF indicator
EOF.V=2
ELSE
READ TRLR.T AS B 1, T 12
CALL PRINT.LINE.SEARCH GIVEN TRLR.T
ALWAYS
LOOP
'' set EOF.V back to 0 so if finds unexpected EOF will abort, but if
'' at EOF will get out of read next NSN loop
SUBTRACT 1 FROM EOF.V
END ''routine TRLR.E.DEMAND

ROUTINE TRLR.U.READ.RQMT
'' This routine reads the 3 years of monthly C&T Program Requirement
'' data from trailer U of the Special SCF report for POI items. It
'' calculates the number of months of requirements or MAX.MONTH.
'' Finally, it calculates the percent of recruit training center
'' demand (PER_RTC.DEMAND) by dividing the recruit PICs (last 2 letters
'' of PIC = AA, AW, GB) requirement over the total NSNs requirement
'' from all the PICS.
DEFINE YR, ROW AS INTEGER VARIABLES
DEFINE HOLD.MAT AS AN INTEGER, 1-DIMENSIONAL ARRAY
DEFINE RTC.SUM, TOTAL.RQMT, RQMT.12MTH AS REAL VARIABLE
DEFINE NSN.T AS A TEXT VARIABLE
DEFINE TEST.EOF AS A ALPHA VARIABLE

RTC.SUM=0
TOTAL.RQMT=0
RQMT.12MTH=0
RESERVE CTREQ.MAT(\*), HOLD.MAT AS 36

' Assumes pointer at TYP TRLR U record left from TRLR E or TRLR U prog.
READ NSN.T AS B 98, T 9
IF NSN.T not equal to NIIN.T
    THEN
        PRINT 1 LINE WITH NIIN.T, NSN.T THUS
        $$ ERROR $$$$$$ HEADER NSN ******** NOT EQUAL TO TRLR E ********
        STOP "processing
        ALWAYS
    WHILE (TRLR.T = "OTYPE TRLR U") DO

READ PIC.T, MONTH, YEAR
AS /, B 55, T 2, B 76, I 2, B 79, I 2

FOR YR =1 TO 3 DO
    START NEW INPUT RECORD
    START NEW INPUT RECORD
    START NEW INPUT RECORD
    FOR ROW =1 TO 12
        READ HOLD.MAT(ROW + ((YR -1) * 12))
    END
    LOOP
    FOR ROW =1 TO 36
        CTREQ.MAT(ROW)=CTREQ.MAT(ROW) + HOLD.MAT(ROW)
        IF (PIC.T = "AA") OR (PIC.T = "AW") OR (PIC.T = "GB")
            THEN add to the RTC sum
            FOR ROW = 1 TO 36
                RTC.SUM=RTC.SUM + HOLD.MAT(ROW)
            END
        ALWAYS
        IF TRACE3 = .TRUE
            PRINT 1 LINE WITH PIC.T THUS
            MONTH PIC ** YR 1 ** YR 2 ** YR 3
            FOR ROW = 1 TO 12 DO
                PRINT 1 LINE WITH ROW, HOLD.MAT(ROW), HOLD.MAT(ROW+12),
                HOLD.MAT(ROW+24) THUS
            END
            LOOP
        END
    END

' at this point after a PIC has been read for all three years
' there are 3 possibilities: 1) another TRLR U follows on this
' page or the next, 2) the header page follows (OTYP NSN) for a
' new NSN, 3) all NSNs have been read and at end of file

EOF.V=1
' so if EOF will stop, not print error & abort
' Set record pointer to next trailer record
READ TRLR.T AS /, B 1, T 12
UNTIL (TRLR.T="OTYP NS") OR (TRLR.T = "OTYPE TRLR U")
OR (EOF.V=2) DO
    READ TEST.EOF AS /, B 1, A 1
    IF (TEST.EOF=26) OR (EOF.V=2)
        THEN at end of file (Note a 26 is a ^Z or DOS EOF indicator
        EOF.V=2
        ELSE

3-15
READ TRLR.T AS B 1, T 12
CALL PRINT.LINE.SEARCH GIVEN TRLR.T

ALWAYS
LOOP
'' set EOF.V back to 0 so if finds unexpected EOF will abort, but if
'' at EOF will get out of read next NSN loop
SUBTRACT 1 FROM EOF.V
LOOP '' while same NSN

'' The PGM RQMTs start at Oct = 1, Nov = 2, etc. Below converts month
'' to position in program file
START.MONTH= MOD.F((MONTH + 3), 12)
MAX.MONTH=36 - (START.MONTH - 1)

FOR ROW = 1 TO 36
   TOTAL.RQMT=TOTAL.RQMT + CTREQ.MAT(ROW)
NEXT ROW
PER.RTC.DEMAND = RTC.SUM/TOTAL.RQMT

FOR ROW =START.MONTH TO (START.MONTH + 11)
   RQMT.12MTH=RQMT.12MTH + CTREQ.MAT(ROW)
NEXT ROW
IF (PGM.RQMT.12.MTH LT RQMT.12MTH*0.95) OR
   (PGM.RQMT.12.MTH GT RQMT.12MTH*1.05)
   THEN is NOT w/i +/- 5% of TRLR U sum
   PRINT 3 LINES WITH PGM.RQMT.12.MTH, RQMT.12MTH
   THUS
   ERROR **************** REQUIREMENTS FROM TRLR & HEADER DO NOT EQUAL
   HEADER 12 MONTH REQUIREMENT ** SUM OF TRLR U **
   ALWAYS

IF TRACE6=.TRUE
   PRINT 3 LINE WITH PIC.T, MONTH, YEAR, RTC.SUM, RQMT.12MTH,
   PGM.RQMT.12.MTH,
   START.MONTH, MAX.MONTH, PER.RTC.DEMAND
   THUS
   PIC ** (IF AA, AW, GB then RTC) MTH/YR **/**
   RTCSUM **.* SUM PGM ** RQMT.12.MTH **
   START MTH ** MAX.MONTH ** PER.RTC.DD *.****
   PRINT 1 LINE THUS

FOR THE TOTAL NSN YEAR 1        YEAR 2        YEAR 3
FOR ROW = 1 TO 12 DO
   PRINT 1 LINE WITH ROW, CTREQ.MAT(ROW), CTREQ.MAT(ROW+12),
   CTREQ.MAT(ROW+24)
   THUS
   ** ** **
   LOOP
   ALWAYS

END ''routine TRLR.U.READ.RQMT
CHAPTER 4
C&T VARIABLE SAFETY LEVEL MODEL

This chapter describes the C&T variable safety level (VSL) model. The VSL model is an analytical model that derives the amount of safety stock that each item in a system of items should receive in order to minimize the total number of time-weighted backorders in the system for a given investment in safety level. The program resides in the subdirectory C:\SIM\VSL on the PC's hard disk. The chapter contains four sections: the outline flow of the program, the list of all procedures, a short description of each procedure, and the program source code.

FLOW OUTLINE OF PROGRAM

PREAMBLE
MAIN
SET.OPTIONS
ALLPGC.INITIALIZE
OPTIONAL.ASSUMPTIONS

Until at end of file do
INPUT.SSCF.DATA
PRINT.SSCF.DATA
If not at end of file
'Then just found the SSCF for another PGC so process it
INPUT.MPT01I.DATA
XYZ.PLTS
DO.Q.INCREMENT
STORE.VSL.DATA
Always
loop
PRINT.ASSUMPTIONS
VSL.EQUATION
PRINT.VSLINFO
OUTPUT.VSL
end

LISTING OF PROCEDURES

PREAMBLE
MAIN
ALLPGC.INITIALIZE
DO.Q.INCREMENT
INPUT.MPT01I.DATA
INPUT.SSCF.DATA
OPTIONAL.ASSUMPTIONS
OUTPUT.VSL
PRINT.ASSUMPTIONS
PRINT.SSCF.DATA
PRINT.VSLINFO
SET.OPTIONS
STORE.VSL.DATA
VSL.EQUATION
XYZ.PLTS

DESCRIPTION OF PROCEDURES

PREAMBLE
This is an analytical model to produce VSL in months for a system of items that is a single PGC or many PGCs. It uses some routines directly from the C&T simulation model. The file produced can be automatically read by the simulation. The VSL model input is the SSCF report file and the Management Policy Table 11 file. Its output is the VSL in months by PGC and NSN ("VSL.DAT") and trace information in the file "VSLOUT.DAT".

MAIN
This routine has the basic structure of the VSL analytical model

ALLPGC.INITIALIZE
This routine has the basic structure of the VSL analytical model

DO.Q.INCREMENT
This routine calculates the Q in the VSL formula which usually represents order quantity. However with incremental deliveries the order quantity (divided by 2) is not an accurate representation of the average stock (assumed by the VSL formula). So if the QINC.OPT is true this routine calculates the average stock of an NSN times 2. It uses in that calculation 3 pieces of information: the number of NSN specific deliveries of an item; the months early the first incremental delivery arrives before the forecasted NSN PLT; and the procurement cycle in months. If the QINC.OPT is false it uses the total order quantity as the Q and assumes no incremental deliveries.

INPUT.MPT011.DATA
This routine Reads management policy table 11 and gets the minimum procurement cycle, PGC delivery percents for all delivery increments, 1 of 4 methods of delivery, PGC first delivery in days.

INPUT.SSCF.DATA
This routine reads the required input data to run the simulation and the VSL model originally captured from the Special Supply Control File Report via a SIMSCRIPT program in directory DLADATA. The routine similar to the simulation routine searches for the desired PGC number, and reads in the data into the appropriate variables. If the PGC number is not found the program prints error message and stops.
OPTIONAL ASSUMPTIONS

This routine lets the user override the standard assumptions, options or traces settings found in ALLPGC.INITIALIZE & SET.OPTIONS. It lets the user specify their own by editing the file ASSUMP.MOD & entering 1 in the user query for the selection of an alternate assumption file.

OUTPUT.VSL

This routine outputs the VSL in months just calculated. It can store this file in the simulation directory so CATS can automatically read it, or in this directory so that the information in the CATS directory will not be destroyed. It prints the entire system of NSNs by PGC and then by NSNs within the PGC.

PRINT.ASSUMPTIONS

This routine prints the answers or the assumptions entered by the user during the initial interactive session.

PRINT.SSCF.DATA

This routine prints the SSCF data read in by routine INPUT.SSCF.DATA

PRINT.VSLINFO

This routine prints all the key information needed to solve the VSL formula. The information is all stored in an entity similar to an array with the index including every NSN for all PGCs in the system.

SET.OPTIONS

This key routine is where all options are set, queries are asked, traces are defined and set, and I/O units are declared.

STORE.VSL.DATA

This routine stores the key variables needed to solve the VSL equation: Q, MADLT, COST, K, Demand/yr. It also calculates the MADLT, the sum of all MADLT*COST, and stores the PGCs name, number of NSN, and code.

VSL.EQUATION

This routine solves the VSL equations once the key variables have been derived and stored(for all NSNs in system) and the sum of MADLT * cost for all NSNs is calculated (both done in STORE.VSL.DATA. The routine calculates VSL and makes sure it is less than 3 standard deviations or the mean leadtime-demand. It also calculates EBOs, fill rates by NSN and cumulative for the system.

XYZ.PLTS

This routine determines which NSN are X, Y, or Z items, and based on delivery method 1 to 4, what the NSNs PLTs are. The routine also calculates the average procurement cycle for each NSN.

4-3
SOURCE CODE

PREAMBLE

''CLOTHING AND TEXTILE VARIABLE SAFETY MODEL
'' (directory VSL) basic features:
'' This is an analytical model to produce VSL in months for a system of
'' items that is a single PGC or many PGCs. It uses some routines
'' directly from the C&T simulation model. The file produced can be
'' automatically read by the simulation. The VSL model input is the
'' SSCF report file and the Management Policy Table 11 file. Its
'' output is the VSL in months by PGC and NSN ("VSL.DAT") and trace
'' information in the file "VSLOUT.DAT".

NORMALLY MODE IS UNDEFINED

PERMANENT ENTITIES

'' Originally, each PGC's NSN raw data feed into this entity where
'' data is aggregated into VSL variables stored in SYSTEM.ATTRIBUTES
'' Each time a new PGC is read all below data is overwritten
EVERY NSN.ATTRIBUTES HAS 'key attributes for each NSN by PGC
A PLT.DAY, 'procurement leadtimes in days
A ARS, 'average requisition size
A AVE.FORECAST, 'AMF over course of simulation: CTREQ+QFD/3
A ROP.QTY, 'reorder point in units
A PCP.MONTH, 'procurement cycle period in months
A SAFETY.MONTH, 'safety level in months either VSL or FSL
A STOCK, 'in stock items or onhand at inventory
A OWRM, 'other war reserve material protactable units
A PER.RTC.DEMAND, 'the percentage of recruit to total demand
A VIP.ITEM, '1=yes VIP(monthly ROPT), 0 Not VIP (quarterly)
A MAD, 'mean absolute deviation in QTR demand (monthly if VIP)
A QFD, 'quarterly forecast demands directly from SSCF
A ALPHA, 'alpha factor from SSCF
A Q.INCREMENT, 'order quantity & avg. stock (no safety level)
'' for incremental deliveries
A NSN.NO '' the NSN number

DEFINE PLT.DAY, ROP.QTY, PCP.MONTH, ARS, MAD,
PER.RTC.DEMAND, Q.INCREMENT AS REAL VARIABLES
DEFINE ALPHA,QFD,SAFETY.MONTH, AVE.FORECAST,
FORECAST.MTH, OWRM, STOCK AS REAL VARIABLES
DEFINE VIP.ITEM AS INTEGER VARIABLE
DEFINE NSN.NO AS TEXT VARIABLE

'' Once a PGCs raw data is read in, key variables are calculated and
'' store in this entity which contains all VSL parameters for each
'' NSN in the entire system
EVERY SYSTEM.ATTRIBUTES HAS
A MIXN, '' NSN number identical to NSN.NO variable
A Q.ORDER, 'identical to Q.INCREMENT
A MADLT, 'LT*MAD
A COST.PU, ' cost per unit, identical to COST

4-4
A DMD.YR, "annual unit demand per year
A SD.MEAN, "Standard deviation to mean ratio
A FILRT, "an items unit fillrate
A EBO, "the items time weighted backorders
A K.SAFETY, "the safety level factor for the item
A Z.ESSENTIAL," the essentiality factor = 10 RTC% demand
A VSL.MONTH" the VSL in months

DEFINE NIN AS A TEXT VARIABLE
DEFINE Q.ORDER, MADLT, COST.PU, DMD.YR, SD.MEAN, FILRT, EBO,
K.SAFETY, VSL.MONTH, Z.ESSENTIAL AS REAL VARIABLES

TEMPORARY ENTITIES
EVERY PGC.MEMBER HAS
A NAME, "the PGC name
A CODE, "the procurement grouping code
A NO.ITEMS " the number of NSNs or MAX.NSNs
BELONGS TO THE PGC.SET
DEFINE NAME AS TEXT VARIABLE
DEFINE NO.ITEMS AND CODE AS REAL VARIABLES

THE SYSTEM
OWNS A PGC.SET
DEFINE PGC.SET AS A FIFO SET

" PGC characteristics
DEFINE MAX.MONTH, "number of months IN POI CTREQ forecasts
MAX.NSN AS INTEGER VARIABLE " number of NSNs in PGC
DEFINE COST AS REAL VARIABLES
DEFINE PGC.NAME AS TEXT VARIABLE
DEFINE ICC AS TEXT VARIABLE ""type of requirements calculation
DEFINE FSC AS INTEGER VARIABLE ""federal supply code
DEFINE PGC.NO AS INTEGER VARIABLE ""PGC code number
DEFINE MIN.PC AS A REAL VARIABLE ""min. procurement cycle(MPT 11)
DEFINE ALT.DAY AS A INTEGER VARIABLES
DEFINE PGC.SL.STOCK AS REAL VARIABLE ""PGC safety level stock
DEFINE RUN.ID AS REAL VARIABLE ""ID when run PGC more than once
DEFINE CTREQ.MAT AS A REAL, 3-DIMENSIONAL ARRAY

" NSN specific means and stand. deviation of requirement matrix
DEFINE MEAN.CTREQ AND STD.CTREQ AS A REAL, 1-DIMENSIONAL ARRAYS
DEFINE TARGET.PGC AS INTEGER VARIABLE ""PGC looking for to get data

" ALL PGCs in the SYSTEM variables
DEFINE SUM.MADCT, "the sum of MADLT*COST for all NSNs in system
BETA.BO, " Backorder lines goal
SUM.WGTFILRT, "demand weighted system fill rate
SUM.EBO,"'sum of EBO over all NSNs in the system
SUM.DEMAND, "'sum of the demand for the system
SUM.VSLCT, "'sum of variable safety level * demand * cost
SUM.FSLCT "'sum of fixed safety level * demand * cost
AS REAL VARIABLE
DEFINE MAX.PGC, "'count of PGCs so far included in VSL
AT.EOF, "'when all PGCs are read set to true
MAXDIM.NSN,"'the maximum dimension or NSN a PGC can have

4-5
MAXSYSDIM.NSN, 'the maximum NSNs for the system VSL
SYSTEM.NSN ' the number of NSNs for all PGCs so far
AS INTEGER VARIABLE
DEFINE
MAX.DELIVERIES, 'no. of months of deliveries for the PGC (MPT011)
FIRST.DELIVERY, 'days of PLT before a NSN is delivered
Z.PERCENT, 'Z item <= z% of PC*DEMAND for matrix deliveries
X.PERCENT 'X item >= x% of PC*DEMAND, Y item remainder
AS REAL VARIABLES
DEFINE M1, M2, T AS REAL VARIABLE 'used in procurement cycle PCP
per cent PGC order delivered each month in matrix delivery
DEFINE DELIVERY.PERCENT AS A REAL, 1-DIMENSIONAL ARRAY

'' simulation options & traces below, see SET.OPTIONS for definitions
DEFINE NEWASSUMP.OPT, DELIVERY.OPT, ALLPGC.OPT, QINC.OPT,
DOREQ.OPT, MODIFYDATA.OPT, MOMP0T11.OPT, ADDPGC.OPT, Z.ESENTL.OPT
AS INTEGER VARIABLES
DEFINE TRACE1, TRACE2, TRACE3, TRACE4, TRACE5, TRACE6, TRACE7,
TRACE8, TRACE9, TRACE10, TRACE11, TRACE12, TRACE13, TRACE14,
TRACE15, TRACE16, TRACE17, TRACE18, TRACE19, TRACE20, TRACE21,
TRACE22, TRACE23, TRACE24 AS INTEGER VARIABLES

'' constants
DEFINE .TOTAL TO MEAN 1 'next 3 constants are the columns
DEFINE .RECRUIT TO MEAN 2 'of the CTREQ.MAT array
DEFINE .OTHER TO MEAN 3
DEFINE .TRUE TO MEAN 1
DEFINE .FALSE TO MEAN 0
DEFINE .DPM TO MEAN 30 'DAYS PER MONTH
DEFINE .MINVAL TO MEAN 0.000000000001

END ''PREAMBLE

MAIN
'' This routine has the basic structure of the VSL analytical model
CALL SET.OPTIONS
CALL ALLPGC.INITIALIZE

UNTIL AT.EOF=.TRUE DO 'all PGCs in file (except if VSL w/in PGC)
CALL INPUT.SSCF.DATA

'' At this point AT.EOF is true if only 1 PGC or at EOF & have
'' and no more PGC information to do
IF (AT.EOF = .FALSE) 'means doing more than 1 PGC & not at EOF
OR (MAX.PGC = 1)'means doing 1ST PGC so in first pass
''THEN just found the SSCF for another PGC so process it
CALL INPUT.MPT011.DATA
CALL XYZ.PLTS
CALL STORE.VSL.DATA
ALWAYS
LOOP
CALL PRINT.ASSUMPTIONS
CLOSE UNIT 4 'SSCFSIM.DAT

4-6
CALL VSL.EQUATION
CALL OUTPUT.VSL
END 'MAIN

ROUTINE ALLPGC.INITIALIZE
' This routine initializes those variables set and held constant
' for all PGCs (T,M1,M2, and optional assumptions)

CREATE EVERY NSN.ATTRIBUTES (MAXDIM.NSN)
CREATE EVERY SYSTEM.ATTRIBUTES (MAXSYSDIM.NSN)

IF NEWASSUMP.OPT=.FALSE
   *** PROCUREMENT CYCLE VALUES  ***
   T=365 '' ordering and holding cost constant
   M1=925 '' dollar value quarterly demand floor, < M1 PCP=36 mth
   M2=9999 '' dollar value quarterly demand ceiling, > M2 PCP=6 mth
ELSE 'read file with optional assumptions
   CALL OPTIONAL.ASSUMPTIONS

ALWAYS

LINES.V=0
PRINT 6 LINES WITH RUN.ID THUS

THE DETAIL VSL TRACE OUTPUT REPORT

(FILE: VSLOUT.DAT)

(ID NUMBER OF RUN  **)

END' ' routine ALLPGC.INITIALIZE

ROUTINE DO.Q.INCREMENT GIVEN NSN AND PERCENT.PCP
' This routine calculates the Q in the VSL formula which usually
' represents order quantity. However with incremental deliveries the
' order quantity (divided by 2) is not an accurate representation of
' the average stock (assumed by the VSL formula). So if the QINC.OPT
' is true this routine calculates the average stock of an NSN times 2.
' It uses in that calculation 3 pieces of information:
' the number of NSN specific deliveries of an item; the months early
' the first incremental delivery arrives before the forecasted NSN PLT;
' and the procurement cycle in months. If the QINC.OPT is false it
' uses the total order quantity as the Q and assumes no incremental
' deliveries.

DEFINE MONTHS.early, DELIVERIES, NSN AS INTEGER VARIABLE
DEFINE PERCENT.PCP, DELIV.RATIO AS REAL VARIABLE
IF QINC.OPT=.TRUE
"'THEN calculate an average stock onhand (no safety stock) for Q
DELIIV.RATIO = MAX.DELIVERIES/6  ' In case PGC deliveries > 6
"' First calculate the common deliveries and months then exceptions
IF (PERCENT.PCP <= Z.PERCENT)
"'THEN Z item
MONTHS.EARLY=0
DELIVERIES=1
ELSE "'X & Y item
MONTHS.EARLY=3 * DELIV.RATIO
DELIVERIES=6 * DELIV.RATIO
ALWAYS
SELECT CASE DELIVERY.OPT ' exceptions for months early & deliver.
CASE 1
MONTHS.EARLY=0 * DELIV.RATIO
DELIVERIES=1
CASE 2
IF (PERCENT.PCP < X.PERCENT) AND (PERCENT.PCP > Z.PERCENT)
"'THEN Y item that starts deliveries in middle of schedule
MONTHS.EARLY = 1 * DELIV.RATIO
DELIVERIES = 3 * DELIV.RATIO
ALWAYS
CASE 3
IF (PERCENT.PCP <= Z.PERCENT)
"'THEN Z item that starts deliveries 1/3 way into schedule
MONTHS.EARLY = 2 * DELIV.RATIO
DELIVERIES = 4 * DELIV.RATIO
ALWAYS
DEFAULT
ENDESELECT
IF TRACE1=.TRUE
PRINT 1 LINE WITH NSN, MONTHS.EARLY, DELIVERIES,
PERCENT.PCP THUS
NSN ** EARLY MTHS ** DELIVERIES ** & PCP **,**
ALWAYS
' calculate Q - order quantity
Q.INCREMENT(NSN)=(((PCP.MONTH(NSN) - (DELIVERIES-1))/2)
+ MONTHS.EARLY) * AVE.FORECAST(NSN) * 2
ELSE 'calculate standard order quantity
Q.INCREMENT(NSN) = PCP.MONTH(NSN) * AVE.FORECAST(NSN)
ALWAYS
END 'routine DO.Q.INCREMENT

ROUTINE INPUT.MPT011.DATA
' 'This routine Reads management policy table 11 and gets the minimum
' ' procurement cycle, PGC delivery percents for all delivery
' ' increments, 1 of 4 methods of delivery, PGC first delivery in days.
DEFINE TEST.TEXT, TEST2 AS TEXT VARIABLE
DEFINE I, PGC.NUM, MONTH AS INTEGER VARIABLE
DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE PGC.PERCENT AS REAL VARIABLE

USE UNIT 11 FOR INPUT
'' USE 6 FOR OUTPUT
EOF.V=1

'' **** PHASED DELIVERY SET UP ********
MAX.DELIVERIES=12
RESERVE DELIVERY.PERCENT(*) AS MAX.DELIVERIES

UNTIL PGC.NUM = TARGET.PGC  DO 'loop to find PGC target number
TEST.TEXT="NEW PGC"
UNTIL TEST.TEXT="ROUP" DO ' loop to find GROUP label
START NEW INPUT RECORD
READ TEST.EOF ''
IF ((TEST.EOF<>26) AND (EOF.V<>2))
''THEN look for GROUP in file to find PGC NUM
READ TEST.TEXT
ELSE '' at end of file without finding PGC's MPT 011 file
WRITE AS "## ERROR: TARGET PGC MPT011 FILE NOT FOUND",
/ USING 6
STOP
REGARDLESS
LOOP
'' have found the GROUP label now read PGC.NUM
START NEW INPUT RECORD
READ PGC.NUM, I, MIN.PC, TEST.TEXT, TEST2
LOOP
FOR MONTH = 1 TO MAX.DELIVERIES
READ DELIVERY.PERCENT(MONTH)
MONTH=1
WHILE (((MONTH<= MAX.DELIVERIES) AND (DELIVERY.PERCENT(MONTH) > 0))
DO ''no. incremental deliveries
DELIVERY.PERCENT(MONTH) = DELIVERY.PERCENT(MONTH)/10 ''make a %
PGC.PERCENT = PGC.PERCENT + DELIVERY.PERCENT(MONTH)
MONTH=MONTH + 1
LOOP
MAX.DELIVERIES=MONTH - 1
IF (PGC.PERCENT < 99.99) OR (PGC.PERCENT > 100.01)
''THEN
WRITE AS "## ERROR: PGC DELIVERY PERCENT NOT EQUAL TO 100",
/ USING 6
STOP
REGARDLESS
START NEW INPUT RECORD

FOR I=1 TO 3
READ TEST2
READ DELIVERY.OPT
FOR I=1 TO 4
READ TEST2
READ FIRST.DELIVERY
FOR I=1 TO 3

4-9
READ TEST2
READ X.PERCENT, Z.PERCENT

* INT 4 LINES WITH PGC.NUM, DELIVERY.OPT, FIRST.DELIVERY, X.PERCENT, Z.PERCENT, MIN.PC THUS

================================== MANAGEMENT POLICY TABLE 11 FILE INPUT ===============
PGC ** METHOD OF DELIVERY ** PGC FIRST DELIVERY DAYS **
X = **% Z = **% MINIMUM PROC CYCLE **

FOR MONTH = 1 TO MAX.DELIVERIES DO
PRINT 1 LINE WITH MONTH, DELIVERY.PERCENT(MONTH) THUS
MONTH = ** DELIVERY.PERCENT **
LOOP
REWIND UNIT 11 'for next PGC

END ''routine INPUT.MPT011.DATA

ROUTE INPUT.SSCF.DATA
'' This routine reads the required input data to run the simulation
'' and the VSL model originally captured from the Special Supply
'' Control File Report via a SIMSCRIPT program in directory DLA\DATA.
'' The routine similar to the simulation routine searches for the
'' desired PGC number, and reads in the data into the appropriate
'' variables. If the PGC number is not found the program prints
'' error message and stops
DEFINE TEST.EOF AS ALPHA VARIABLE
DEFINE MONTH, CCL, NSN AS INTEGER VARIABLE
DEFINE TEST.TEXT AS TEXT VARIABLE
USE UNIT 4 FOR INPUT ''C:\SIM\DLA\SSCSIM.DAT/MOD
EOF.V=1
'' ******** Find target PGC's beginning of data input ****
PRINT 1 LINE WITH PGC.NO, TARGET.PGC THUS
BEGINNING OF IN SSCF NUM ** TARGET **
PGC.NO=9999
UNTIL PGC.NO = TARGET.PGC DO ''loop to find PGC target number
TEST.TEXT="NEW PGC"
UNTIL TEST.TEXT="PROC.GR.CD" DO '' loop to find PROC.GR.CD label
START NEW INPUT RECORD
READ TEST.EOF ''
IF ((TEST.EOF<>26) AND (EOF.V<>2)) '' THEN look for GROUP in file to find PGC NUM
READ TEST.TEXT
''
PRINT 1 LINE WITH TEST.TEXT THUS
''
TEST TEXT *
ELSE '' at end of file
IF ALLPGC.OPT = .TRUE
'' THEN at end of file so continue with rest of VSL program
AT.EOF=.TRUE
RETURN
ELSE '' can not find target PGC in SSCF report file
WRITE AS "### ERROR: TARGET PGC NOT IN SSCF REPORT ",

4-10
/ USING 6
STOP
REGARDLESS
LOOP
'.'
have found the GROUP label now read PGC.NO
READ PGC.NO, TEST.TEXT, MAX.NSN
IF ALLPGC.OPT = .TRUE
'.' THEN want to use data from each PGC found
TARGET.PGC = PGC.NO
ALWAYS
'.'
PRINT 1 LINE WITH PGC.NO, TEST.TEXT, MAX.NSN, MAX.PGC
'.'
PGC NO. ** TEXT ******** MAX.NSN ** PGC **
LOOP

''******** Start reading PGC related data ***************

MAX.PGC = MAX.PGC + 1
READ PGC.NAME AS /,,B 1,T 20
READ FSC, ICC, ALT.DAY, COST, MAX.MONTH
'.' AS /,,B 1,T 17, B 22,I 7, B 29,T 3, B 34,I 6, B 41, D(10,2), B 57, I 5
SKIP 2 RECORDS

''******** Read NSN specific data ***********************
FOR NSN = 1 TO MAX.NSN
READ NSN, NSN.NO(NSN), PLT.DAY(NSN), VIP.ITEM(NSN),
SAFETY.MONTH(NSN), QFD(NSN)
SKIP 3 RECORDS

FOR NSN = 1 TO MAX.NSN
READ NSN, MAD(NSN), OWRM(NSN), ALPHA(NSN), ARS(NSN),
PER.RTC.DEMAND(NSN)
IF ICC = "P"
'.' THEN Read C&T requirements matrix ***********
IF MAX.PGC = 1
'.' THEN first PGC so set up requirements & statistics matrices
RESERVE CTRREQ.MAT(*,*,*) AS MAXDIM.NSN BY 36 BY 1
RESERVE MEAN.CTRREQ(*) AND STD.CTRREQ(*) AS MAXDIM.NSN
ALWAYS
FOR NSN = 1 TO MAX.NSN DO
SKIP 3 RECORDS
FOR COL = 1 TO MAX.MONTH
READ CTRREQ.MAT(NSN,COL,.TOTAL)
LOOP
FOR NSN = 1 TO MAX.NSN DO
FOR MONTH = 1 TO MAX.MONTH DO
COMPUTE
MEAN.CTRREQ(NSN) AS THE MEAN AND
STD.CTRREQ(NSN) AS THE STD.DEV OF
CTREQ.MAT(NSN,MONTH,.TOTAL)
LOOP
AVE.FORECAST(NSN) = MEAN.CTRREQ(NSN) + (QFD(NSN)/3)
LOOP
ELSE 'do QFD item only
FOR NSN=1 TO MAX.NSN
  AVE.FORECAST(NSN) = (QFD(NSN)/3)
ALWAYS

IF ALLPGC.OPT = .FALSE
  'THEN done reading one PGC from file
  AT.EOF = .TRUE
ALWAYS
CALL PRINT.SSCF.DATA
END 'routine INPUT.SSCF.DATA

ROUTINE OPTIONAL.ASSUMPTIONS
' 'This routine lets the user override the standard assumptions, options
' 'or traces settings found in ALLPGC.INITIALIZE & SET.OPTIONS. It lets
' 'the user specify their own by editing the file ASSUMP.MOD & entering
' '1 in the user query for the selection of an alternate Assumption file.

DEFINE TEST.TEXT AS TEXT VARIABLE

USE UNIT 3 FOR INPUT ' 'ASSUMP.DAT
UNTIL TEST.TEXT="T" DO
  READ TEST.TEXT
  START NEW INPUT RECORD
  LOOP
  READ T, M1, M2
  UNTIL TEST.TEXT="TRACES" DO
    START NEW INPUT RECORD
    READ TEST.TEXT
    LOOP
    START NEW INPUT RECORD
    READ TRACE17
    SKIP 3 INPUT RECORD
    READ DOREQ.OPT
    CLOSE UNIT 3

    PRINT 1 LINE WITH TRACE17 AND DOREQ.OPT
    TRACE 17 IS ** DO REQ OPTIONS **
END 'routine OPTIONAL.ASSUMPTIONS

ROUTINE OUTPUT.VSL
' ' This routine outputs the VSL in months just calculated. It
' 'can store this file in the simulation directory so CATS can
' 'automatically read it, or in this directory so that the information
' 'in the CATS directory will not be destroyed. It prints the
' 'entire system of NSNs by PGC and then by NSNs within the PGC.

DEFINE COUNT,ANS, NSN, LAST.NSN AS INTEGER VARIABLE
USE UNIT 6 FOR OUTPUT
PRINT 6 LINE THUS
ENTER THE DIRECTORY WHERE YOU WANT THE VSL VALUES (IN MONTHS) FILE
TO BE STORED:
0 TO PLACE THE FILE DIRECTLY INTO THE SIMULATION DIRECTORY
SO THAT THE SIMULATION MODEL WILL AUTOMATICALLY USE THE VSL.
1 TO PLACE THE FILE IN THIS DIRECTORY SO THAT IT WILL NOT
OVERWRITE AND DESTROY THE EXISTING VSL INFORMATION.
READ ANS
IF ANS = 0
' THEN.store in simulation directory
OPEN UNIT 12 FOR OUTPUT, FILE NAME IS "C:\SIM\DLA\VSL.DAT"
ELSE 'store info in this directory so as not to destroy old VSL
OPEN UNIT 12 FOR OUTPUT, FILE NAME IS "C:\SIM\VSL\VSL.DAT"
ALWAYS
USE UNIT 12 FOR OUTPUT
LINES.Y=0
PRINT 4 LINES THUS
+++++++++++++++++++++++++++++++++++++++++++++++
VSL DATA BY PGC AND NSN IN MONTHS
+++++++++++++++++++++++++++++++++++++++++++++++ 

' WHILE PGC.SET IS NOT EMPTY DO
FOR EACH PGC.MEMBER IN PGC.SET DO
COUNT = COUNT + 1
' REMOVE FIRST PGC.MEMBER FROM PGC.SET
PRINT 3 LINES WITH CODE, NAME, COUNT, MAX.PGC, NO.ITEMS THUS

P PGC ** PGC NAME ******************** PGC ** OUT OF ** SYSTEM PGCs
NSN VSL(MONTHS) NIIN NSNs WITHIN PGC **
' Do next PGC from last NSN is system done plus an additional
' MAX.NSN for the next PGC
FOR NSN= (LAST.NSN + 1) TO (LAST.NSN + NO.ITEMS) DO
PRINT 1 LINE WITH NSN, VSL.MONTH(NSN), NIIN(NSN) THUS
** ****** ********************
LOOP
LAST.NSN = LAST.NSN + NO.ITEMS
' DESTROY PGC.MEMBER
LOOP 'for next member in PGC.SET

END ''OUTPUT.VSL

ROUTINE PRINT.ASSUMPTIONS
'Prints the answers or the assumptions entered by the user during
''the initial interactive session.

PRINT 2 LINES THUS
+++++++++++++++++++++++++++++++++++++++++++++++++++
PRINT 11 LINES WITH ALLPGC.OPT, TARGET.PGC, BETA.B0, MODIFYDATA.OPT,
MODMPT011.OPT, NEWASSUMP.OPT, QINC.OPT, 2.ESNTL.OPT, MAXDIM.NSN,
MAXSYSDIM.NSN THUS

======== MODEL OPTION ASSUMPTIONS (true=1 and false=0) ==========

0 ALL PGCs IN SSCF IN SYSTEM VSL** (0:FALSE= VSL within PGC for below)
1) PGC NUMBER **
2) BETA VALUE FOR FIRST PASS **
3) EDITED THE SSCF DATA ** (0:FALSE= use standard data with no change)
4) EDITED MPTOLL TABLE ** (0:FALSE= use standard data with no change)
5) EDITED ASSUMPTIONS ** (0:FALSE = standard assumptions, no change)
6) INCREMENTAL DELIVERY Q ** (0:FALSE= Q is order quantity)
7) ESSENTIALITY FACTOR ZE ** (0:FALSE ZE = 1, else ZE = %RTC demand + 0.5)
8) MAXIMUM SYSTEM NSNs ** MAXIMUM NSNs IN ANY PGC **

END "routine PRINT.ASSUMPTIONS"

ROUTINE PRINT.SSCF.DATA

"This routine prints the SSCF data read in by routine INPUT.SSCF.DATA
DEFINE NSN, COL AS INTEGER VARIABLE

IF TRACE17=.TRUE
PRINT 6 LINE WITH PGC.NO, MAX.NSN THUS

==================================
==================================
PGC SPECIAL SUPPLY CONTROL FILE INPUT DATA =========

PROC.GR.CD ** NUMBER OF NSN **

PRINT 2 LINES WITH PGC.NAME, FSC, ICC, ALT.DAY, COST, MAX.MONTH
THUS
ITEM NAME FSC ICC ADM.LT STANDARD.PRICE MAX.MONTH
*************** ** ** ** ** **

PRINT 1 LINE THUS
NSN NIIN PRO.LT VIP(1=Y) FIX.SAFE QFD

FOR NSN = 1 TO MAX.NSN
PRINT 1 LINE WITH NSN, NSN.NO(NSN), PLT.DAY(NSN), VIP.ITEM(NSN),
SAFETY.MONTH(NSN), QFD(NSN) THUS
** ******** ** ** ** **

PRINT 2 LINES THUS

NSN MAD OWRMRP ALPHA ARS PER.RTC.DEMAND

FOR NSN = 1 TO MAX.NSN
PRINT 1 LINE WITH
NSN, MAD(NSN), OWRM(NSN), ALPHA(NSN), ARS(NSN),
PER.RTC.DEMAND(NSN) THUS
** ** ** ** ** ** ** **

IF ICC="P"
"THEN POI item and print CTREQ matrix
FOR NSN = 1 TO MAX.NSN DO
PRINT 2 LINE WITH NSN, NSN.NO(NSN), MAX.MONTH THUS
NSN ** NIIN ******** CT REQUIREMENT MATRIX FOR ** MONTHS ======
MONTHS: 1 -------- 2 -------- 3 -------- 4 -------- 5 -------- 6
BEGIN REPORT PRINTING
FOR COL= 1 TO MAX.MONTH IN GROUPS OF 6
PRINT 1 LINE WITH A GROUP OF CTREQ.MAT(NSN,COL,.TOTAL) FIELDS
THUS
**     **     **     **     **     **
END 'REPORT
LOOP

PRINT 3 LINE THUS

SUMMARY ON MONTHLY TOTAL FORECAST AND C&T 36 MONTH POI FORECASTS
NSN TOTAL AMF POI AMF POI STD % POI STD/POI AMF
FOR NSN=1 TO MAX.NSN DO
PRINT 1 LINE WITH NSN, AVE.FORECAST(NSN),
MEAN.CTREQ(NSN), STD.CTREQ(NSN),
(100*STD.CTREQ(NSN)/MEAN.CTREQ(NSN)) THUS
**     **     **     **     **     **
LOOP
ALWAYS 'CTREQ print
ALWAYS 'trace block
END 'routine PRINT.SSCF.DATA

ROUTINE PRINT.VSLINFO GIVEN FAC
' This routine prints all the key information needed to solve the
'VSL formula. The information is all stored in an entity similar
'to an array with the index including every NSN for all PGCs
'in the system.

DEFINE NSN, FAC AS INTEGER VARIABLE
USE UNIT 1 FOR OUTPUT
LINES.V=0
PRINT 3 LINES WITH FAC THUS
*************** FINAL SUMMARY RESULTS FOR THE SYSTEM ***************
<BACKORDERS-EBO> <AVAILABILITY> <COST IN ** DOLLARS > DEMAND
<BETA MODEL> <% FILL RATE > <VSL> FSL MADLT > AMP/1000
PRINT 1 LINE WITH BETA.BO, SUM.EBO, (SUM.WGTFILRT/SUM.DEMAND),
SUM.VSLCT/FAC, SUM.FSLCT/FAC, SUM.MADCT/FAC, SUM.DEMAND/1000 THUS
**, **     **     **     **     **
PRINT 3 LINE WITH MAX.PGC, SYSTEM.NSN, SUM.MADCT THUS
*************** SUMMARY VSL INFORMATION ***************
NUMBER OF PGCs ** NSNs IN SYSTEM ** SUM OF MADLTxCT **
NSN Q COST MADLT DMD/YR %LTSD/D EBO FILLRT K VSL.MT ZE
FOR NSN=1 TO SYSTEM.NSN DO
PRINT 1 LINE WITH NSN, Q.ORDER(NSN), COST.PU(NSN),
MADLT(NSN), DMD.YR(NSN), 100*SD.MEAN(NSN), EBO(NSN), FILRT(NSN),
ROUTE SET.OPTIONS
' This key routine is where all options are set, queries are asked,
' traces are defined and set, and I/O units are declared.
DEFINE DETAIL.OPT, ANS AS INTEGER VARIABLE

USE UNIT 5 FOR INPUT

' ****************** OPTIONS SET BELOW **********************
MAXDIM.NSN = 200 'the maximum number of NSNs for any 1 PGC
MAXSYSDIM.NSN = 300 'the maximum number of NSNs sum over all PGCs
PRINT 9 LINES WITH MAXDIM.NSN, MAXSYSDIM.NSN THUS
=============== VSL ASSUMPTIONS ===============
1) NO PGC HAS MORE THAN ** NSNs
2) TOTAL NUMBER OF NSNs FOR ALL PGCs IS NO MORE THAN ** NSNs
3) MATRIX DELIVERY INCONSISTENCIES THAT MAKE VSL FORMULA UNCERTAIN
   - IF DELIVERY METHOD IS #1: INCONSISTENCY BETWEEN ROP & DELIVERED PLTs
   - DELIVERIES > 6, HAVE ROP PLTs DIFFERENT THAN DELIVERED PLTs
4) VIP item alpha = .05, non VIP item alpha = .15 (or a & b factors are .7 & .36 for VIP; .57 & .46 for non VIP, respectively

PRINT 10 LINE THUS
1) ENTER #### NUMBER 0 TO 5 FOR THE PGC SELECTED TO RUN ****************************

<table>
<thead>
<tr>
<th>NAME</th>
<th>SERVICE</th>
<th>MAX</th>
<th>NSN</th>
<th>PGC NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 - DEMO PGC (MAN'S SHIRT)</td>
<td>ARMY</td>
<td>3</td>
<td>1672</td>
<td></td>
</tr>
<tr>
<td>1 - MAN'S COAT</td>
<td>ARMY</td>
<td>65</td>
<td>1765</td>
<td></td>
</tr>
<tr>
<td>2 - WOMAN'S SHIRT</td>
<td>AIR FORCE</td>
<td>21</td>
<td>1671</td>
<td></td>
</tr>
<tr>
<td>3 - WOMAN'S SKIRT</td>
<td>ARMY</td>
<td>80</td>
<td>1748</td>
<td></td>
</tr>
<tr>
<td>4 - MEN'S SHOE</td>
<td>ALL</td>
<td>113</td>
<td>1505</td>
<td></td>
</tr>
<tr>
<td>5 - MEN &amp; WOMEN GLOVES</td>
<td>ALL</td>
<td>17</td>
<td>1834</td>
<td></td>
</tr>
<tr>
<td>6 - WANT TO ENTER AN ALTERNATE PGC NUMBER</td>
<td>99 - FOR ALL PGCS IN THE SSCF report (file &quot;SSCFSIM.DAT&quot;)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

READ ANS
'TIME.VAL is real time minutes to run ALL NSNs for a simulation year
SELECT CASE ANS
CASE 0
   TARGET.PGC=1672
CASE 1
   TARGET.PGC=1765
CASE 2
   TARGET.PGC=1671
CASE 3
   TARGET.PGC=1748
CASE 4
TARGET.PGC=1505
CASE 5
TARGET.PGC=1834
CASE 6
PRINT 2 LINE THUS
1a) ENTER THE PGC NUMBER (NOTE: BOTH THE SSCFSIM.DAT/MOD AND THE
MP011.DAT/MOD FILES MUST ALREADY HAVE THIS PGC'S DATA WITHIN
READ TARGET.PGC
DEFAULT
PRINT 5 LINES THUS
ASSUMPTIONS FOR VSL WITH MULTIPLE PGCS
ASSUMES 1) ONLY PGCs FOR VSL IN SSCF "SSCFSIM.DAT"
2) THOSE PGCs ALSO IN MPT011 FILE (THOUGH THE MPT011
CAN HAVE PGCs IN DIFFERENT ORDER AND CAN HAVE PGCs
NOT INCLUDED IN THE SSCF)
ALLPGC.OPT=.TRUE
ENDSELECT
''1 FOR AN ORDER QUANTITY (Q) CONSIDERING INCREMENTAL DELIVERIES
''0 FOR A Q EQUAL TO THE PROCUREMENT CYCLE x MONTHLY FORECAST
QINC.OPT = 1 'default equals incremental deliveries.
PRINT 1 LINE THUS
2) ENTER BETA OR THE BACKORDER LINES ON-HAND GOAL
READ BETA.BO

'*************** TRACE OPTIONS SET BELOW *****************************
' in routine PRINT.DEMANDS for trace 2 & 3
TRACE17=.FALSE 'prints the values read in from the SSCF file
TRACE7=.TRUE 'prints the first CTREQ.MAT matrix
TRACE14=.FALSE 'prints PLT stored values, runs PLT 1000 times

TRACE22=.TRUE 'Matrix delivery PLTs, %PCP, XYZ vectors & NSNs,
TRACE1=.FALSE 'prints the months early and NSN deliveries
TRACE2=.FALSE 'T=months of leadtime, MADLT, sum MADLT*COST
TRACE3=.TRUE 'VSL equation,3SD, MADLTDMD, & fillrate, exp factors
'*************** DETAIL QUERIES SET BELOW *****************************
PRINT 2 LINE THUS
3)ENTER 1 FOR FURTHER INPUT SPECIFICATIONS (QUERIES 4 TO 8)
0 FOR NO FURTHER CHANGE AND RUN
READ DETAIL.OPT
IF DETAIL.OPT=.TRUE
'THEN ******** do DETAIL QUERY for graphs, files, phasing

PRINT 2 LINE THUS
4)ENTER 1 FOR OPTIONAL SCF INPUT DATA
0 FOR STANDARD SCF INPUT DATA [D]
READ MODIFYDATA.OPT

PRINT 2 LINES THUS
5)ENTER 1 FOR OPTIONAL MANAGEMENT POLICY TABLE INPUT DATA (MPT011)
0 FOR STANDARD MANAGEMENT POLICY TABLE INPUT DATA [D]
READ MODMPT011.OPT

4-17
PRINT 2 LINE THUS
6) ENTER 1 FOR OPTIONAL ASSUMPTION FILE: M1, M2, T, OPTIONS, TRACES
    0 FOR STANDARD ASSUMPTIONS [D]
    READ NEWASSUMP.OPT

PRINT 2 LINE THUS
7) ENTER 1 FOR ORDER QUANTITY(Q) TO CONSIDER INCREMENTAL DELIVERIES [D]
    0 FOR A Q EQUAL TO THE PROCUREMENT CYCLE x MONTHLY FORECAST
    READ QINC.OPT

PRINT 2 LINES THUS
8) ENTER 1 FOR THE ESSENTIALITY FACTOR ZE = % DEMAND FOR RTC + 0.5
    0 FOR NO ESSENTIALITY CONSIDERATIONS OR ZE = 1[D]
    READ Z.ESNTL.OPT ** Essentiality factor 1 through 9

PRINT 4 LINES WITH MAXSYSDIM.NSN THUS
9a) ENTER 1 TO CHANGE MAXIMUM NUMBER OF NSNs IN SYSTEM (NOW AT **)
    0 TO KEEP MAX NUMBER AT CURRENT CEILING VALUE [D]
    (NOTE: FOR MODEL TO ALLOCATE ENOUGH SPACE THIS VALUE MUST BE
    GREATER THAN OR EQUAL TO THE NO. OF NSNs FOR ALL THE PGCs)
    READ ANS
    IF ANS = 1
        PRINT 1 LINE THUS
        ENTER THE MAXIMUM NUMBER OF NSNs YOU WANT INSTEAD
        READ MAXSYSDIM.NSN
        ALWAYS
        ALWAYS
        ALWAYS
        CALL PRINT.ASSUMPTIONS

        ********** INPUT/ OUTPUT SPECIFICATIONS **********
        OPEN UNIT 1 FOR OUTPUT, FILE NAME IS "C:\SIM\VSL\VSLOUT.DAT"
        USE UNIT 1 FOR OUTPUT

        IF TARGET.PGC=1672
            THEN use sample input file
            IF MODIFYDATA.OPT=.TRUE
                OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\DEMOPGC.MOD"
            ELSE
                OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\DEMOPGC.DAT"
            ALWAYS
            ELSE
                IF MODIFYDATA.OPT=.TRUE
                    OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\SSCFSIM.MOD"
                ELSE
                    OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\DLA\SSCFSIM.DAT"
                ** TEST FILE
                OPEN UNIT 4 FOR INPUT, FILE NAME IS "C:\SIM\VSL\SSCF.TST"
            ALWAYS
            ALWAYS
        IF NEWASSUMP.OPT=.TRUE
            OPEN UNIT 3 FOR INPUT, FILE NAME IS "C:\SIM\DLA\ASSUMP.MOD"
            ALWAYS
            ** matrix delivery schedule info. and first delivery PLT

4-18
IF MDMPT011.OPT=.TRUE
    OPEN UNIT 11 FOR INPUT, FILE NAME IS "C:\SIM\DLA\MPT011.MOD"
ELSE
    OPEN UNIT 11 FOR INPUT, FILE NAME IS "C:\SIM\DLA\MPT011.DAT"
ALWAYS

' NOTE: UNIT 12 will be OPENed in the "OUTPUT.VSL" routine.

END ''ROUTINE OPTIONS

ROUTINE STORE.VSL.DATA
' ' This routine stores the key variables needed to solve the VSL
' 'equation: Q, MADLT, COST, K, Demand/yr. It also calculates the
' 'MADLT, the sum of all MADLT*COST, and stores the PGCs name,
' 'number of NSN, and code.
DEFINE NSN, SYS.NSN AS INTEGER VARIABLES
DEFINE T.LT AS REAL VARIABLE
CREATE A PGC.MEMBER
NAME = PGC.NAME
CODE = PGC.NO    ' ' the procurement grouping code number
NO.ITEMS = MAX.NSN    ' ' the number of NSNs or MAX.NSNS
FILE PGC.MEMBER IN PGC.SET
SYS.NSN=SYSTEM.NSN
FOR NSN=1 TO MAX.NSN DO ''store NSN info into system entity
    ADD 1 TO SYS.NSN
    NIIN(SYS.NSN)= NSN.NO(NSN)    ' ' NSN number
    Q.ORDER(SYS.NSN)= Q.INCREMENT(NSN)
    COST.PU(SYS.NSN)= COST    ' ' cost per unit
    DMD.YR(SYS.NSN)=AVE.FORECAST(NSN) * 12    ' 'annual unit demand per year
    IF (Z.ESNTL.OPT = .TRUE)
        ' 'THEN use essentiality Z factor of ten times % RTC demand
        Z.ESSENTIAL(SYS.NSN) = PER.RTC.DEMAND(NSN) + 0.5
    ELSE ''assume no essentiality factor or all 1
        Z.ESSENTIAL(SYS.NSN) = 1
    ALWAYS
    IF VIP.ITEM(NSN)=1 ''
        ' ' THEN VIP alpha is 0.05
        T.LT=(ALT.DAY + PLT.DAY(NSN))/DPM
        MADLT(SYS.NSN)=(.7 + 0.36 * T.LT) * MAD(NSN)
        SD.MEAN(SYS.NSN)=(1.25*MADLT(SYS.NSN))
            /(AVE.FORECAST(NSN) * T.LT)
    ELSE ''NON-VIP alpha is 0.15
        T.LT=(ALT.DAY + PLT.DAY(NSN))/ (DPM * 3)    ' 'quarterly info
        MADLT(SYS.NSN)=(0.57 + 0.46 * T.LT) * MAD(NSN)
        SD.MEAN(SYS.NSN)=(1.25 * MADLT(SYS.NSN))
            /(3 * AVE.FORECAST(NSN) * T.LT)
    ALWAYS
    SUM.DEMAND=SUM.DEMAND + AVE.FORECAST(NSN)
    SUM.MADCT =SUM.MADCT + (MADLT(SYS.NSN) * COST)
SUM.FSLCT = SUM.FSLCT +
   (SAFETY.MONTH(NSN) * AVE.FORECAST(NSN) * COST)
IF TRACE2 = .TRUE  THEN
   PRINT 1 LINE WITH SYS.NSN, T.LT, MADLT(SYS.NSN), SUM.MADCT
   THUS NSN ** T.LT **.**** MADLT **.**** SUMMT **
   ALWAYS
END LOOP
SYSTEM.NSN = SYS.NSN

ROUTINE VSL.EQUATION
' This routine solves the VSL equations once the key variables
' have been derived and stored (for all NSNs in system) and the
' sum of MADLT * cost for all NSNs is calculated (both done in
' STORE.VSL.DATA). The routine calculates VSL and makes sure it is less
' than 3 standard deviations or the mean leadtime demand. It also
' calculates EBOs, fill rates by NSN and cumulative for the system.

DEFINE 1.EXP.QMD, EXP.EXP AS REAL VARIABLE
DEFINE BETA.TEST, VSL.EQU, VSL.3SD, VSL.MLTD, AMF AS REAL VARIABLES
DEFINE PASS, NSN, FAC AS INTEGER VARIABLE
   FAC = 1000  'factor to get dollar cost of levels into 1000s
   BETA.BO = BETA.TEST
   USE UNIT 6 FOR OUTPUT
   PRINT 3 LINES WITH FAC
   =============== FINAL SUMMARY RESULTS FOR THE SYSTEM ===============
   <BACKORDERS-EBO> <AVAILABILITY> <COST IN ** DOLLARS > DEMAND
   PASS <BETA MODEL> <% FILL RATE > <VSL FSL MADLT >AMF/1000

   WHILE (BETA.TEST > 0) DO
   BETA.BO = BETA.TEST
   SUM.WGTFILRT = 0
   SUM.EBO = 0
   SUM.VSLCT = 0
   ADD 1 TO PASS
   '*************** VSL EQUATION FOR A GIVEN BETA PASS ***************
   FOR NSN = 1 TO SYSTEM.NSN DO
     1.EXP.QMD = 1 - EXP.F(-1.1313 * (Q.ORDER(NSN)/MADLT(NSN)))
     ' Next Equation from "Review of SAMMS Requirements", M.K.Cyrus,
     ' 8/85. Note: Unit not requisition demand so ARS not needed .
     K.SAFETY(NSN) = -.7071 * LOG.E.F(1/2.56 * Q.ORDER(NSN) * COST.PU(NSN) * BETA.BO)
     AMF = DMD.YR(NSN)/12
     AMF = -DMD.YR(NSN)/12
     VSL.EQU = (1.25 * K.SAFETY(NSN) * MADLT(NSN)) / AMF
     IF VSL.EQU < 0
       THEN negative VSL and set to zero
       VSL.MONTH(NSN) = 0
     ELSE ' ' make sure VSL is below constraints
   END FOR NSN
   END WHILE
   END "routine STORE.VSL.DATA
VSL.3SD = (3 * 1.25 * MADLT(NSN)) / AMF
VSL = mean leadtime demand (derived from ratio of SD/mean demand in a leadtime)
VSL.MLTD = (MADLT(NSN) * 1.25) / SD.MEAN(NSN) / AMF
VSL.MONTH(NSN) = MIN.F (VSL.EQU, VSL.3SD, VSL.MLTD)

ALWAYS
In case not using VSL.EQU resolve for K
K.SAFETY(NSN) = VSL.MONTH(NSN) * AMF / (1.25 * MADLT(NSN))
Calculates nonfill rate (availability) & time weighted backorders
via Presutti & Trepp article equations 8 & 10, respectively
1.4142 = SQRT(2)
EXP.EXP=1.EXP.QMD * EXP.F(-1.4142 * K.SAFETY(NSN))
*** Fill Rate (supply availability) where .35355 =(0.5 / 1.4142)
1 - non fill rate \/
FILRT(NSN)=100 * (1 - ((0.35355 * 1.25*MADLT(NSN) * EXP.EXP)
/Q.ORDER(NSN)))

IF TRACE3=.TRUE
USE UNIT 1 FOR OUTPUT
PRINT 1 LINE WITH NSN, VSL.EQU, VSL.3SD, VSL.MLTD,
K.SAFETY(NSN) THUS
NSN ** VSL: EQU **.** 3SD **.** MLTD **.** K **.**
PRINT 1 LINE WITH NSN, FILRT(NSN), EXP.EXP, 1.EXP.QMD THUS
NSN ** FILR **.***** E.E **.*** 1.EX **.***
USE UNIT 6 FOR OUTPUT
ALWAYS
Time weighted backorders (EBOs)
EBO(NSN)=(.5/2)* (((1.25*MADLT(NSN))**2)/Q.ORDER(NSN)) * EXP.EXP
SUM.WGTFLRT= SUM.WGTFLRT + (AMF * FILRT(NSN))' weighted fill
SUM.EBO = SUM.EBO + EBO(NSN)' sum of sys. time weighted BOs
SUM.VSLCT = SUM.VSLCT + (VSL.MONTH(NSN) * AMF * COST.PU(NSN))
LOOP '************ END OF VSL EQUATION FOR BETA PASS ************

PRINT 1 LINE WITH PASS, BETA.BO, SUM.EBO,(SUM.WGTFLRT/SUM.DEMAND),
SUM.VSLCT/FAC, SUM.FSLCT/FAC, SUM.MADCT/FAC, SUM.DEMAND/1000 THUS
** **.** **.** **.*** **.** **.** **.**
WRITE AS "ENTER NEW BETA (TO STOP ENTER 0)" +
READ BETA.TEST
LOOP
CALL PRINT.VSLINFO GIVEN FAC
END 'VSL.EQUATION

ROUTINE XYZ.PLTS
'This routine determines which NSN are X, Y, or Z items, and based
' on delivery method 1 to 4, what the NSNs PLTs are. The routine
' also calculates the average procurement cycle for each NSN.
DEFINE NSN AS INTEGER VARIABLE
DEFINE PERCENT.PCP, PGC.PCP, DVQD AS REAL VARIABLE
PRINT 5 LINE THUS

**************************************************************************
**************** SIMULATION DATA DESCRIPTION ****************************
**************************************************************************
FOR NSN=1 TO MAX.NSN DO
DVQD=TRUNC.F(AVE.FORECAST(NSN)*3)*COST
  IF DVQD <= M1
    THEN DVQD set for a 36 month procurement cycle
    PCP.MONTH(NSN)=36
  ELSE
    IF (DVQD > M1) AND (DVQD <= M2)
       THEN between M1 & M2 so use Wilson Lot Size equation
       PROCURE CYCLE (MONTHS) = EOQ / MONTHLY DEMAND
       PCP.MONTH(NSN)=TRUNC.F((3*T)*(DVQD**(-.0.5)))
       ELSE "greater than M2 or use 6 month PCP"
       PCP.MONTH(NSN)=6
     ALWAYS
     ALWAYS
   END IF
   LOOP
FOR NSN= 1 TO MAX.NSN " sum to use as average order quantity
PGC.PCP=PGC.PCP + (AVE.FORECAST(NSN)*PCP.MONTH(NSN))
FOR NSN = 1 TO MAX.NSN DO
PERCENT.PCP = (100 * (AVE.FORECAST(NSN)*PCP.MONTH(NSN)))/PGC.PCP

SELECT CASE DELIVERY.OPT "calculate the NSN specific PLTs
CASE 1: "************ METHOD 1 DELIVERY OPTION ************
  IF (PERCENT.PCP >= X.PERCENT)
     THEN X item
     PLT.DAY(NSN)= FIRST.DELIVERY +
       (((1/3)*MAX.DELIVERIES * .DPM)
  ELSE
    IF (PERCENT.PCP <= Z.PERCENT)
      THEN Y item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        (((5/6)*MAX.DELIVERIES * .DPM)
    ELSE 'Y item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        (((2/3)*MAX.DELIVERIES * .DPM)
  ALWAYS
  ALWAYS
CASE 2 "************ METHOD 2 DELIVERY OPTION ************
  IF (PERCENT.PCP >= X.PERCENT)
     THEN X item
     PLT.DAY(NSN)= FIRST.DELIVERY +
       (((1/2)*MAX.DELIVERIES * .DPM)
  ELSE
    IF (PERCENT.PCP <= 2.PERCENT)
      THEN 2 item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        (((5/6)*MAX.DELIVERIES * .DPM)
    ELSE 'Y item
      PLT.DAY(NSN)= FIRST.DELIVERY +
        (((2/3)*MAX.DELIVERIES * .DPM)
  ALWAYS
  ALWAYS
ALWAYS
4-22
CASE 3 '************ METHOD 3 DELIVERY OPTION ************

IF (PERCENT.PCP >= X.PERCENT) ' THEN X item

PLT.DAY(NSN)= FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * .DPM)

ELSE

IF (PERCENT.PCP <= Z.PERCENT) ' THEN Z item

PLT.DAY(NSN)= FIRST.DELIVERY +
((2/3)*MAX.DELIVERIES * .DPM)

ELSE ' Y item

PLT.DAY(NSN)= FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * .DPM)

ALWAYS

ALWAYS

CASE 4 '************ METHOD 4 DELIVERY OPTION ************

IF (PERCENT.PCP >= X.PERCENT) ' THEN X item

PLT.DAY(NSN)= FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * .DPM)

ELSE

IF (PERCENT.PCP <= Z.PERCENT) ' THEN Z item

PLT.DAY(NSN)= FIRST.DELIVERY +
((5/6)*MAX.DELIVERIES * .DPM)

ELSE ' Y item

PLT.DAY(NSN)= FIRST.DELIVERY +
((1/2)*MAX.DELIVERIES * .DPM)

ALWAYS

ALWAYS

ENDSELECT

CALL DO.Q.INCREMENT GIVEN NSN AND PERCENT.PCP

END

''USE UNIT 6 FOR OUTPUT

IF TRACE22=.TRUE

PRINT 2 LINES WITH PGC.PCP, FIRST.DELIVERY, QINC.OPT

PGC TOTAL PCP ** 1ST DEL ** QINC.OPT **

NSN PCP PLT Q INCREMENT Q MONTHS AVE.FOR DVQD

FOR NSN=1 TO MAX.NSN DO

PRINT 1 LINE WITH NSN, PCP.MONTH(NSN), PLT.DAY(NSN),
Q.INCREMENT(NSN),(Q.INCREMENT(NSN)/ AVE.FORECAST(NSN)),
AVE.FORECAST(NSN),(TRUNC.F(AVE.FORECAST(NSN)*3)*COST) THUS

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

LOOP

ALWAYS

END ''routine XYZ.PLTS