

NUREG/CR-1308

SAND79-2242

Unlimited Release

AN

Fixed Site Neutralization Model Programmer's Manual

Volume II (Listings)

**Dennis Engi, Leon D. Chapman, Sandia Laboratories
W. Judnick, R. Blum, L. Broegler, J. Lenz, A. Weintraub,
D. Ballard, Vector Research, Inc.**

Prepared by Sandia Laboratories, Albuquerque, New Mexico 87185
and Livermore, California 94550 for the United States Department
of Energy under Contract DE-AC04-76DPO-139

Printed December 1979

1
2
3
4
5
6
7
8
9

DOCUMENT CONTROL DESK

016

5



Sandia Laboratories

SF 2900 Q(7-73)

THIS DOCUMENT CONTAINS
POOR QUALITY PAGES

8010090008

NOTICE

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, or any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use, or the results of such use, of any information, apparatus, product or process disclosed in this report, or represents that its use by such third party would not infringe privately owned rights.

The views expressed in this report are not necessarily those of the U. S. Nuclear Regulatory Commission

Available from
National Technical Information Service
Springfield, Virginia 22161

THIS DOCUMENT CONTAINS
POOR QUALITY PAGES

NUREG/CR-1308
SAND79-2242
Unlimited Release
AN

FIXED SITE NEUTRALIZATION MODEL
PROGRAMMER'S MANUAL
VOLUME II (LISTINGS)

Dennis Engi, Leon Chapman
Sandia Laboratories
Albuquerque, NM 87185

W. Judnick, R. Plum, L. Broegler,
J. Lenz, A. Weintraub, D. Pallard
Vector Research, Incorporated
Ann Arbor, Michigan 48106

Date Published: December 1979

Submitted by
Vector Research
Ann Arbor, Michigan 48106
under
Contract Document Nos. 07-5018 and 07-7098
to
Sandia Laboratories
Albuquerque, New Mexico 87185
operated by
Sandia Corporation
for the
U. S. Department of Energy

Prepared for
Division of Safeguards, Fuel Cycle and Environmental Research
Office of Nuclear Regulatory Research
U. S. Nuclear Regulatory Commission
Washington, DC 20555
Under Interagency Agreement DOE 40-550-75
NFC FIN No. A1060

7.0 SOURCE LISTINGS

This chapter presents the source listings for the Fixed Site Neutralization Model and its supporting modules, the Plex Preprocessor and the Data Preprocessor. These three programs employ identical common blocks and block data subroutines; their listings are presented once, as an economy and convenience, in section 7.1. In the listings in the subsequent sections, lines of code detailing the contents of common blocks will be abbreviated to the form "COMMON /NAME/ " with the understanding that the full contents of /NAME/ are accessible in section 7.1.

The Plex Preprocessor listings are in section 7.2; the Data Preprocessor listings, section 7.3; and the Fixed Site Neutralization Model listings, section 7.4. The utility routines are called by both the Plex Preprocessor and the Fixed Site Neutralization model; for economy and convenience, these are listed but once, in alphabetical order, in section 7.5.

Throughout the listings the reader will find liberal use of comment cards. Detailed verbal descriptions of the logic are available in volume 1 of this manual, as follows: the Plex Preprocessor, section 2.1; the Data Preprocessor, section 2.2; and the Fixed Site Neutralization Model, section 2.3.

7.1 Common and Block Data

This subsection contains a listing of common blocks and the block data subroutine that are shared by two or more programs in the system. The common blocks are presented first because they are referred to in the block data routine.

ISN

COMMON BLOCKS ~ ALL PROGRAMS

1

```

COMMON /STATEV/ DTMIN, FORCES(4,2), IPLAYR, ISIDE, ISECPL(2), NMSTAT,
+      TMIN, FMRESP,
+      LISTND(2,4000), LOCATN(4,800), CHANGE(4,200), SITE(10,6),
+      BLDG(10,8), FLOOR(12,12), BARRIER(12,30), YARD(15,60),
+      ROOM(13,40), HALL(13,12), ROOF(11,10), STAIRS(10,20),
+      DOOR(17,120), WINDOW(16,100), WEAPON(2,600), EQUIP(2,720),
+      VEHICLE(7,36), PERSON(18,350), SENSOR(6,720), ACTIVD(4,20),
+      COMNET(3,20), PRCEPT(4,900), MESSAGE(5,300), ACTION(6,500),
+      RELN(3,300), GROUP(4,10), GOALND(6,10), NIMPAC(5,720),
+      FORCE(8,24)

```

2

DIMENSION ITEM(41900), DITEM(41900)

3

EQUIVALENCE (DTMIN, ITEM(1), DITEM(1))

4

COMMON /PARS/ DTPNAM(30), FLDNAM(250), FORMOT(12,30),

```

+      DTPABR(30), TAVAIL(30), IDTOFF(30), IDTPOK(30), IFDNPT(30),
+      IPHSOB(6), IRCOFF(30), IRECIO(30), LSTREC(30), MACTTP(3,8),
+      MRELTP(2,20), NFLDS(30), NRECS(30), NRECS0(30),
+      IFAIL, IOK, IRN(6), IRTNCD, NULL

```

5

EQUIVALENCE (NULL, NULL), (IFAIL, FAIL)

6

REAL*8 DTPNAM, FLDNAM, FORMOT

7

COMMON /PARS1/

```

+      JACSON, JACTIV, JADVRQ, JADVRS, JALLEG, JAMT, JATTR, JBARIN,
+      JBARRS, JBLDGS, JCBARS, JCBDGS, JCMNTS, JCONT, JCONTS, JCOV,
+      JCOVRG, JCOV1, JCOV2, JCOV3, JCOV4, JCRED, JDOER, JDRACC,
+      JDRI VR, JELEMS, JENVIR, JEQPRQ, JFALEG, JFLDR, JFLOOR, JFLRS,
+      JFPLNS, JFRAC, JFSOPS, JFSTAT, JGARDS, JHALLS, JHIGH, JHIST,
+      JHORIZ, JID, JINCOM, JINFLR, JLEADER, JLOCK, JLOCN, JLOS, JMESS,
+      JMODS, JNBRS, JNSGEN, JNUM, JNVAL, JNXT, JOWNER, JPAR,
+      JPAE1, JPAR2, JPAR3, JPEN, JPLACE, JPLANS, JPORT, JPORTS,
+      JPOSTR, JPRCPS, JPROPS, JPSTAT, JRCVRS, JRMACC, JROOFS, JROOMS,
+      JRTYPE, JSENS, JSINK, JSITN, JSOPS, JSOURC, JSTAIR, JSTARS,
+      JSTAT, JSUBJ, JSUBS, JSUCHS, JSUPRN, JTIM, JTIME,
+      JTINRQ, JTREQ, JTYPE, JVAL, JVEHS, JVEL, JVERT, JVIEW,
+      JVIS, JWALLS, JWEAPS, JXCO, JKCO1, JKCO2, JXLEN, JYARDS, JYCO,
+      JYCO1, JYCO2, JYLEN, JZCO, JZLEN, KAFE, KAIR, KALARM, KAPC

```

8

COMMON /PARS2/

ISH

COMMON BLOCKS - ALL PROGRAMS

+ KBARS, KBINFL, KBINRC, KBINRY, KBRBLD, KBRICK, KCAPNG, KCAPTR,
+ KCAR, KCLEAN, KCLOSD, KCOMAL, KCOMBT, KCOMC3, KCOMIF, KCOMNG,
+ KCOMPH, KCPTRG, KCRAHL, KCRDSN, KCRED1, KCRED2, KCRED3, KDANGR,
+ KDARK, KDEAD, KDESTR, KDETFL, KDETIV, KDETNG, KDETPN, KDETSK,
+ KDETST, KDETVE, KECH1, KECH2, KECH3, KEFFCL, KEQPCL,
+ KFIGRN, KFILAW, KFIMG, KFIRIP, KFIRNG, KFIRSK, KFRTSK, KGASS,
+ KGASK, KGRNAD, KHEL, KIMMOB, KKEY, KKEYLK, KLAMIN, KLATER,
+ KLAW, KLIGHT, KLIM1, KLIM2, KLIM3, KLIM4, KLIM5, KLOCKD, KLOCKR,
+ KLSPKR, KMCGUN, KMETSN, KMODNS, KMOVE, KMOVNG, KMOVSK, KNGAS,
+ KNKYLK, KNLOCK, KNOISY, KNVGAS, KOPEN, KOPERL, KORDER, KPBLIC,
+ KPHONE, KPICFI, KEICHIC, KPISTL, KPLAN, KPLNNG, KPOPAQ, KPORCL,
+ KPORSN, KPPKEY, KPPOPE, KPPUNL, KPPWEP, KPSN, KPSNCL,
+ KPV1SN, KPV2SN, KPNHOL, KPVOUN, KQUIET, KRADAL, KRADIO, KRADO2,
+ KREST, KRIPLE, KRSTNG, KRUN, KSABOT, KSAFE, KSCALE, KSCALR,
+ KSECRT, KSECUR, KSENAC, KSENCL, KSF, KSHBUL, KSHGAS,
+ KSIGNL, KSITN1, KSITN2, KSITN3, KSITN4, KSITN5, KSNM, KSNMON

9 COMMON /PARS3/

+ KSNMSN, KSTAT, KSTEEL, KSTOLN, KTGARV, KTGAS, KTGP03, KTGP06,
+ KTGP07, KTGPSN, KTGSEN, KTGVHE, KTOPAQ, KTRANP, KTRGAS, KTRIPD,
+ KTRUCK, KVEH, KVEHCL, KVEST, KVFAST, KVISFI, KVISMV, KVISST,
+ KVNORM, KVOICE, KVOLUN, KWALK, KWALL, KWEPC, KWHOLE, KWIRE,
+ KWOOD, KWORKR, KWOUND, KXPLSN, KXPRES, LACTD, LACTN, LBARR,
+ LBLDG, LCHNG, LCOMNT, LDOOR, LEQUIP, LFLOOR, LFORCE, LFUNC, LGOAL,
+ LGROUP, LHALL, LINODE, LLIST, LLOCN, LMESS, LPERCP, LPERSN,
+ LRELN, LROOF, LROOM, LSENS, LSITE, LSTAIR, LVEHIC, LWEAP,
+ LWIND, LYARD, NACPAR, NACSTP, NACTTP, NACTVY, NALLEG, NATTRK,
+ NCHEM, NCOMRK, NCOMTP, NDETCI, NDETTP, NDTYPE, NDWPTP, NENG, NEQPTP,
+ NEVAL, NFDKRC, NFEATR, NFORWD, NGENRC, NHFLIM, NINFRQ, NIHSID,
+ NLIGHT, NLOCK, NMOCVCL, NMOPOTP, NNMSIT, NNOISE, NOBJCL, NOCTNT,
+ NPCOND, NPCRED, NPEN, NPERSN, NPHCAT, NPORST, NPOTP, NPSITN, NPSNTP,
+ NRELTP, NRLPAR, NRNGCL, NRNGAS, NROLTP, NSECRQ, NSENTP, NSHDTP,
+ NSIDE, NSKLTP, NSNRTP, NSPOTP, NSTATE, NSTDDV, NTGTTP, NVEHTP,
+ NVOLOH, NVOPCL, NVOPTP, NWEPRK, NVIS, NWEPTP, NWTCI,
+ NDATAV, NPARSQ

10 COMMON /DATAV/ ACTRAT(25), AREA(25), BASLOD(7,2), COMDLA(7),
+ DEGVOL(7,2), DETANG(8), DETPHI, DETRAT(3), DETSEN(17,7),

COMMON BLOCKS - ALL PROGRAMS

ISN

```
+ DNGDET(9,2),DNGWEP(5,8),DTPLAN,DTSEC,EVALN(60),HFLIMS(5,2),
+ IACTTP(25),IARMTDP(4),ICOMRQ(2,5,25),IFIRTP(7),IRNKWP(7,7),
+ ISECAC(25),ISECRQ(4,2),LABEL(25),MINPLR(2),MPEN(5),
+ NDTSEC(25),PK(7,7),POBATT(31),POBPSN(2,2,7),
+ POBSEN(2),POBVEH(2,2),POSTUR(25),PSIG(8,7,7),
+ PWOUND(7,7),RESFIR(50),RESMOV(50),RESOBS(50),
+ RESPLN(50),RESSUR(50),RESPTM,RESWND(50),RISK(4,2),
+ RNGLET(7),RNGMAX(7),RNGMIN(7),SENRNG(7),
+ SKILL(4,2,50),SUPDEC(2),SUPFIR(2),SUPMOV(2),SUPOBS(2),
+ TFIN,TSTALE,WIDTH(5)
11   DIMENSION ACTTIM(25)
12   EQUIVALENCE (ACTTIM(1),ACTRAT(1))
13   COMMON /RECREF/ IGOALS,IRBSITE,ISITE,IMNOBS(4),IOBSV(50),LASMNT,
+     LINTCP(50),LNODIN(2),LOCAL(5),NOLDSN,NWSNLT
14   DIMENSION RECRFQ(140)
15   EQUIVALENCE (RECRFQ(1),IGOALS)
16   COMMON /GARCOL/ ACTIVE(9800),ICRECS(30),IQ,IRECQ(1500),
+     IVRECS(30),NCOLCT,NRECRF,NSTACK
17   LOGICAL*1 ACTIVE
18   COMMON /PARS4/ JTMIN,KAND,KEQ,KEQF,KGE,KGEF,KGT,KGTF,KIF,KLE,
+     KLEF,KLT,KLTF,KNE,KNEF,KNOT,KOR,KUNTIL,KWHEN
```

BLOCK DATA - ALL PROGRAMS

ISN

```

1      BLOCK DATA
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /GACCOL/
9      LOGICAL*1 ACTIVE
10     COMMON /PARS4/

```

C

11 DATA

```

+      JACSON/18/, JACTIV/15/, JADVRQ/6/, JADVRS/5/, JALLEG/4/,
+      JAMT/2/, JATTR/3/, JBARIN/17/, JBARRS/8/, JBLDGS/1/,
+      JCBAKS/3/, JCBDGS/12/, JCMNTS/12/, JCONT/5/, JCONTS/4/,
+      JCOV/11/, JCOVRG/4/, JCOV1/12/, JCOV2/13/, JCOV3/14/,
+      JCOV4/15/, JCRED/5/, JDOER/6/, JDRACC/13/, JDRIVR/7/,
+      JELEMS/3/, JENVIR/7/, JEQPRQ/5/, JFALEG/2/, JFLDR/3/,
+      JFLOOR/13/, JFLRS/4/, JFPLNS/7/, JFRAC/3/, JFSOPS/8/,
+      JPSTAT/5/, JGARDS/4/, JHALLS/5/, JHIGH/10/, JHIST/3/,
+      JHOBIZ/14/, JID/1/, JINCOM/5/, JINFLR/9/, JLEADR/10/,
+      JLOCK/9/, JLOCN/3/, JLOS/10/, JMESS/3/, JMODS/5/,
+      JNBRS/6/, JNSGEN/5/, JNUM/2/, JNVAL/4/, JNXT/1/, JOWNER/5/,
+      JPAP/2/, JPAR1/2/, JPAR2/3/, JPAR3/4/, JPEN/11/, JPLA/2/4/,
+      JPLANS/16/, JPORT/16/, JPORTS/9/, JPOSTR/8/, JPRCPS/13/,
+      JPROPS/4/, JPSTAT/2/, JRCVRS/2/, JRMACC/12/, JROOFS/6/,
+      JROOMS/4/, JRTYPE/6/, JSENS/5/, JSINK/2/, JSITN/6/,
+      JSOPS/17/, JSOURC/1/, JSTAIR/8/, JSTARS/6/, JSTAT/8/

```

12 DATA

```

+      JSUBJ/2/, JSUBS/11/, JSUCRS/2/, JSUPRN/7/, JTIM/4/,
+      JTME/4/, JTMRQ/3/, JTREQ/14/,
+      JTYPE/1/, JVAL/2/, JVHS/6/, JVEL/6/, JVERT/15/, JVVIEW/2/,
+      JVIS/12/, JWALLS/5/, JWEAPS/9/, JXCO/1/, JXC01/1/,
+      JXCO2/3/, JXLEN/8/, JYARDS/2/, JYCO/2/, JYCO1/2/, JYCO2/4/,
+      JYLEN/9/, JZCO/3/, JZLEN/10/, KAFE/2/, KAIR/1/, KALARM/6/

```

BLOCK DATA - ALL PROGRAMS

ISN

+ KAPC/3/, KBARS/5/, KBINFL/1/, KBINRC/2/, KBINRY/1/,
+ KBRICK/7/, KCAPNG/4/, KCAPTR/2/, KCAR/1/, KBRBLD/11/,
+ KCLEAN/1/, KCLOSD/3/, KCOMAL/22/, KCOMBT/3/, KCOMCB/21/,
+ KCOMIF/24/, KCOMNG/6/, KCCMPH/23/, KCPTRG/20/, KCRAWL/1/,
+ KCRDSN/7/, KCRED1/1/, KCRED2/2/, KCRED3/3/, KDANGR/2/,
+ KDARK/1/, KDEAD/1/, KDESTR/1/, KDETFI/18/, KDETMV/17/,
+ KDETNG/3/, KDETNP/2/, KDETSK/4/, KDETST/16/, KDETVE/1/,
+ KECH1/1/, KECH2/2/, KECH3/3/, KEFFCL/6/, KEQPCL/2/,
+ NEQPTP/5/, KFIGRN/15/, KFILAW/14/, KFIMG/13/, KFIRIF/12/,
+ KFIRNG/2/, KFIRSK/3/, KFRTSK/2/, KGLASS/2/, KGMASK/5/

13

DATA

+ KGHNAD/5/, KHEL/4/, KIMMOB/2/, KKEY/1/, KKEYLK/2/,
+ KLAMIN/9/, KLATER/1/, KLAW/4/, KLIGHT/2/, KLIM1/1/,
+ KLIM2/2/, KLIM3/3/, KLIM4/4/, KLIM5/5/, KLOCKD/4/,
+ KLOCKR/1/, KLSPEKR/7/, KMCGUN/3/, KMETSN/2/, KMODNS/2/,
+ KMOVE/2/, KMOVNG/1/, KMOVSK/1/, KNGAS/3/, KNKYLK/3/,
+ KNLOCK/1/, KNOISY/3/, KNVGAS/7/, KOPEN/2/, KOPERL/3/,
+ KORDER/2/, KPBLIC/1/, KPHONE/1/, KPICFL/3/, KPICBC/4/,
+ KPISTL/1/, KPLAN/25/, KPLNNNG/7/, KPOPAQ/1/, KPORCL/7/,
+ KPORSN/4/, KPPKEY/9/, KPPOPE/7/, KPPUNL/8/, KPPWEF/10/,
+ KPSN/2/, KPSNCL/4/, NPSNTP/1/, KPV1SN/5/, KPV2SN/6/,
+ KPWHOL/2/, KPWOUN/1/, KQUIET/1/, KRADAL/2/, KRADIO/2/,
+ KRADC/3/, KREST/19/, KRIFLE/2/, KRSTNG/5/, KRUN/3/,
+ KSABOT/3/, KSAFE/1/, KSCALE/4/, KSCALR/2/, KSECRT/2/,
+ KSECUR/5/, KSENAC/2/, KSENCL/5/, KSFE/1/, KSHBUL/1/,
+ KSHGAS/2/, KSTGNL/5/, KSITN1/1/, KSITN2/2/,
+ KSITN3/3/, KSITN4/4/, KSITN5/5/, KSNM/3/, KSNMON/2/

14

DATA

+ KSNMSN/3/, KSTAT/1/, KSTEEL/8/, KSTOLN/4/, KTGARV/1/,
+ KTGAS/2/, KTGP03/5/, KTGP06/6/, KTGP07/7/, KTGPSN/3/,
+ KTGSEN/4/, KTGVFH/2/, KTOPAQ/2/, KTRANP/3/, KTRGAS/6/,
+ KTRIPD/2/, KTRUCK/2/, KVEH/1/, KVEHCL/3/, KVEST/4/,
+ KVFAST/6/, KVISFI/3/, KVISMV/2/, KVISST/1/, KVNORM/5/,
+ KVOICE/4/, KVOLUN/1/, KWALK/2/, KWALL/6/, KWEPCCL/1/,
+ KWHOLE/4/, KWIRE/3/, KWOOD/4/, KWORKR/1/, KWOUND/3/,
+ KXPLSN/1/, KXPRES/2/, LACTD/21/, LACTN/25/, LBARR/8/

BLOCK DATA - ALL PROGRAMS

ISN

```

+      LBLDG/6/, LCHNG/4/, LCOMNT/22/, LDOOR/14/, LEQUIP/17/,  

+      LFLOOR/7/, LFORCE/30/, LGOAL/28/, LGROUP/27/, LHALL/11/,  

+      LINODE/29/, LLIST/2/, LLOCN/3/, LMESS/24/, LPERCP/23/,  

+      LPERSN/19/, LRELN/26/, LROOF/12/, LROOM/10/, LSENS/20/,  

+      LSITE/5/, LSTAIR/13/, LVEHIC/18/, LWEAP/16/, LWIND/15/,  

+      LYARD/9/, NACPAR/3/, NACSTP/25/, NACTTP/8/, NACTVY/7/,  

+      NALLEG/2/, NATTRK/5/, NCHEM/3/, NCOMRK/7/, NCOMTP/7/,  

+      NDETCL/3/, NDETTP/9/, NDTYPE/30/, NDWPTP/8/, NENG/3/  

15     DATA  

+      NEVAL/60/, NFDXRC/250/, NFEATR/31/, NFORWD/12/, NGENRC/4/,  

+      NHFLIM/5/, NINFRQ/2/, NIXSID/4/, NLIGHT/2/, NLOCK/3/,  

+      NMOVCL/2/, NMPOTP/17/, NNMSIT/4/, NNOISE/3/, NOBJCL/7/,  

+      NOCTNT/8/, NPCOND/2/, NPCRED/3/, NPEN/9/, NPERSN/50/,  

+      NPHCAT/4/, NPORST/5/, NPOTP/25/, NPSITN/4/, NRELTP/20/,  

+      NRLPAR/2/, NRNGCL/4/, NRNGKS/8/, NROLTP/3/, NSECRQ/2/,  

+      NSENTP/7/, NSHDTP/2/, NSIDE/2/, NSKLTP/4/, NSNRTP/4/,  

+      NSPOTP/8/, NSTDDV/2/, NTGTTP/7/, NVEHTP/4/, NVIS/3/,  

+      NVOLOR/2/, NVOPCL/2/, NVOPTP/5/, NWEPRK/7/, NWEPTP/7/,  

+      NWTCI/4/, NSTATE/41900/, NPARSQ/1690/, NDATAV/2103/, LPUNC/31/,  

+      JTMIN/15/  

16     DATA NCOLCT/9610/, NRECRF/140/, NSTACK/1500/  

17     DATA NRECS/1,4000,800,200,6,8,12,30,60,40,12,10,20,120,100,  

+      600,720,36,350,720,20,20,900,300,500,300,10,10,120,24/  

18     DATA NFLDS/16,2,4,4,10,10,12,12,15,13,13,11,10,17,16,  

+      2,2,7,18,6,4,3,4,5,6,3,4,6,5,8/  

19     DATA MACTP/3,3,5, 3,3,2, 2,3,2, 3,5,2/  

20     DATA MRELTP/24*4,12*3/  

21     DATA NULL/-9999/, IFAIL/-9998/, IOK/0/  

22     DATA IDTPOK/8*0,7*1,2*0,3*2,10*0/  

23     DATA  

+      KLT/1/, KLE/2/, KEQ/3/, KNE/4/, KGT/5/, KGE/6/, KLTF/7/,  

+      KLEF/8/, KEQF/9/, KNEF/10/, KGTF/11/, KGEF/12/, KAND/13/,  

+      KOR/14/, KNOT/15/, KIF/16/, KUNTIL/17/, KWHEN/18/  

24     END

```

7.2 *Plex Preprocessor*

This subsection contains listings for the Plex Preprocessor. Detailed contents of the common blocks for this program are displayed in section 7.1. Utility routines, used to process the plex structure and associated lists, are in section 7.5.¹ All other subprograms are listed following the main program. Detailed verbal discussions of the logic of the main program and its subprograms may be found in section 2.1, in volume 1 of this manual.

¹Essentially, those routines needed to accomplish the processing as described in section 3.3, in volume 1 of this manual.

PLEX PREPROCESSOR -MAIN PROGRAM

C MAIN ROUTINE OF THE PLEX PREPROCESSOR. THE PLEX PREPROCESSOR
C READS IN THE PLEX RECORD DESCRIPTION, THE INITIAL PLEX DATA FILE,
C AND THE RESPONSE FORCE DATA FILE, CONVERTING THEM TO INTERNAL FORM
C AND STORING THEM IN COMMON /STATEV/. AS IT READS IN THE PLEX
C STRUCTURE IT DOES ERROR CHECKING AND ALERTS THE USER TO ANY ERRORS
C ENCOUNTERED. THE PLEX PREPROCESSOR ALSO GENERATES THE VALUES OF
C CERTAIN VARIABLES BASED ON THE USER INPUT, RELIEVING HIM OF THE
C NECESSITY OF INPUTTING DATA WHICH CAN BE INFERRED FROM A BASIC SET.
C AFTER THE PLEX PREPROCESSOR HAS READ AND STORED THE PLEX STRUCTURE,
C IT WRITES IT OUT. A COPY OF THE PLEX STRUCTURE IS WRITTEN IN
C EXTERNAL FORM SIMILAR TO THE INPUT PLEX STRUCTURE, TO ALLOW THE
C USER TO CHECK THAT THE PROCESSING WAS SATISFACTORY AND TO ALLOW HIM
C TO OBSERVE ANY CORRECTIONS MADE. THE PLEX PREPROCESSOR ALSO
C WRITES SEVERAL LARGE BINARY DATA BLOCKS SO THAT IT CAN TRANSMIT
C THE INTERNAL PLEX STRUCTURE BUILT TO THE MAIN SIMULATION. A THIRD
C OUTPUT PRODUCED IS A MAP OF THE SITE DEFINED BY THE PLEX STRUCTURE.
C

C INPUT FILES:

C 3 RECORD DESCRIPTION FILE

C 4 RECORD DATA FILE

C 5 RESPONSE FORCE FILE

C OUTPUT FILES

C 6 BINARY PLEX STRUCTURE

C 7 ERROR COMMENTS

C 8 PLEX STRUCTURE IN EXTERNAL FORM

C 9 MAP OF SITE

C

1 COMMON /STATEV/

2 D1NENSTON ITEM(1900),DITEM(41900)

3 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

4 COMMON /PARS/

5 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)

6 FAIL*R DTPNAM,FDNAM,FORMOT

7 COMMON /PARS1/

8 COMMON /PARS2/

FLEX PREPROCESSOR -MAIN PROGRAM

```
1 ISN
2      COMMON /PARS3/
3      COMMON /RECREF/
4      DIMENSION RECRE0(140)
5      EQUIVALENCE (RECRE0(1),IGOALS)
6      COMMON ID,TA(50),IMAX(50),IMIN(50),IUSED(101001),MAXLIS
7      DIMENSION XMIN(50),XMAX(50),DATA(50)
8      EQUIVALENCE (XMAX(1),IMAX(1)),(XMIN(1),IMIN(1))
9      EQUIVALENCE (DATA(1),IDATA(1))
10     LOGICAL ENDFL1,ENDFL2
11
12     C ... TEMPORARY CODE UNTIL PARS UPDATED
13     MNSDR = 5
14     KTNSDR = 4
15
16     C INITIALIZE VARIABLES
17
18     CALL INITPP
19
20     C ESTABLISH INTERNAL BUFFER
21     CALL FTNCMD("BUFFER $ LENGTH=300",20)
22
23     C INPUT RECORD DESCRIPTIONS
24     NDTYPE=1
25     CALL INDESC(3)
26
27     C ... SET UP FOR INPUT OF PLEX DATA
28
29     CALL FTNCMD("SET NODECHECK=OFF",17)
30     MAXLIS=0
31     ENDFL1=.FALSE.
32     ENDFL2=.FALSE.
33     N=1
34     DO 5 I=2,NDTYPE
35     N=N+NREC3(I)
```

PLEX PREPROCESSOR -MAIN PROGRAM

ISN

31 CALL NULIFY(IUSED,N)

 C

 C ... GET FIXED INITIAL DATA

 C

32 10 CALL INPLEX(4, IDTYP)

33 IF(IDTYP.EQ.LSITE) ISITE=NEWREF(LSITE,LSTREC(LSITE),0)

34 IF(IDTYP.EQ.LPERCP) GO TO 20

35 IF(IDTYP.NE.NULL) GO TO 10

36 ENDFL1=.TRUE.

 C

 C ... GET FIXED RESPONSE FORCE DATA

 C

37 20 CALL TNPLEX(5, IDTYP)

38 IF(IDTYP.EQ.LPERCP) GO TO 205

39 IF(IDTYP.EQ.LSITE) IRSITE=NEWREF(LSITE,LSTREC(LSITE),0)

40 IF(IDTYP.NE.NULL) GO TO 20

41 ENDFL2=.TRUE.

 C

 C -- CHECK CONTENT FIELD OF ALL REGIONS FOR LOCATIONS
 OF PLAYERS, VEHICLES, SENSORS, AND ACTIVATED DELAYS

 C DO PERSONS AND VEHICLES FIRST. WHILE DOING PERSONS

 C PERFORM INSIDER CHECK ALSO

 C

42 205 NN=0

43 DO 220 JL=JGARDS,JVEHS

44 IPLRST=IVAL(JL,ISITE)

45 IPLARF=IFIRST(IPLRST,KA)

46 210 IF(IPLARF.EQ.NULL) GO TO 220

47 CALL KONTNT(IPLARF)

48 CALL PARSRC(IPLARF, ID,IREC, IDUM)

49 IF(ID .NE. LPERSN) GO TO 212

50 IF(IVAL(LTYPE,IPLARF) .NE. KNSDR) GO TO 212

51 NN = NN + 1

52 IF(NN .LE. KNSDR) GO TO 211

53 100? FORMAT('0*** MORE THAN',I2,' INSIDERS ENCOUNTERED'//
 +' PERSON',I3,' CHANGED TO COMBATANT')

FLEX PREPROCESSOR -MAIN PROGRAM

1SN
54 WRITE(7,1002) NINSDR,IREC
55 GO TO 212
56 211 CALL INCHK(IPLARF,NN)
57 212 IPLARF=NEXT(IPLRST,KA)
58 GO TO 210
59 220 CONTINUE
C
C INSURE SENSORS AND ACTIVATED DELAYS ON APPROPRIATE
C CONTENTS LISTS. ALSO FOR SENSORS, INSURE THEY ARE
C ON APPROPRIATE SENSOR LISTS
60 DO 250 IDTYP=LSENS,LACTD
61 MN=LSTREC(IDTYP)
62 IF(MN .LE. 0) GO TO 250
63 DO 240 I=1,MN
64 IRECRF=NEWREF(IDTYP,I,0)
65 CALL KENTNT(IRECRF)
66 IF(IDTYP .EQ. LACTD) GO TO 240
C
C MARK COVERAGE FIELDS FOR SENSORS
C
67 ICOVLS=IVAL(JCOVRG,IRECRF)
68 ILDC=IFIRST(ICOVLS,KC)
69 230 IF(ILDC .EQ. NULL) GO TO 240
70 ILIST=IVALL(JSENS,ILDC)
C
C SEE IF ILDC HAS THIS SENSOR ON COVERAGE LIST
C
71 ION=LSERCH(ILIST,IREFCRF,NULL,NULL,NULL)
72 IF(ION .NE. NULL) GO TO 235
C
C ADD TO COVERAGE LIST
C
73 ILIST=ISTACK(IREFCRF,ILIST)
74 IPTP=IREFF1(ILDC)
75 ITEM(IPTP+JSENS)=ILTST
76 235 ILDC=NEXT(ICOVLS,KC)

PLEX PREPROCESSOR -MATHN PROGRAM

ISN
77 GO TO 230
78 240 CONTINUE
79 250 CONTINUE
C
C ADD FORCE REFERENCE TO JACSON FIELD OF PERSONS
C
80 NM=LSTREC(LFORCE)
P1 IF(NM .LE. 0) GO TO 30
82 DO 260 I=1,NM
83 IFORC=NEWREF(LFORCE,I,0)
84 ILIST=IVAL(JCANTS,IFORC)
85 IPLARF=IFIRST(ILIST,KP)
86 255 IF(IPLARF .EQ. NULL) GO TO 260
C
C ONLY CONCERNED WITH PERSONS
C
87 CALL PARSRF(IPLARF,INTYP,IRECNO,IDUM)
88 IF(INTYP .NE. LPERSH) GO TO 258
C
C ... INSURE THAT INSIDER NOT INCLUDED ON FORCE
C
89 IF(IVAL(JTYPE,IPLARF) .NE. KINSDR) GO TO 256
90 CALL DELIST(KP,JCANTS,IFORC)
91 1001 FORMAT('0*** INSIDER INCLUDED IN FORCE*,I2,* ERRONEOUSLY*,//
+ * PERSON-',I2,' DELETED FROM FORCE CONTENTS*)
92 WRITE(7,1001) I,IRECNO
93 GO TO 258
C
94 256 IPTR=IRFF1(IPLARF)
95 ITEM(IPTR+JACSON)=IFORC
96 258 IPLARF=NEXX(ILIST,KP)
97 GO TO 255
98 260 CONTINUE
C
C ... INPUT OF FIXED PLEX RECORDS COMPLETE
C SEE NRECS0

FLEX PREPROCESSOR -MAIN PROGRAM

1SN
C
100 30 DO 40 I=1,NDTYPE
100 NRECS0(I)=LSTREC(I)
101 40 CONTINUE
C
C ... GET REPLACEABLE INITIAL DATA
C
102 IF(ENDFL1) GO TO 60
103 50 CALL INPLEX(4,INTYP)
104 IF(INTYP.NE.NULL) GO TO 50
C
C ... GET REPLACEABLE RESPONSE FORCE DATA
C
105 60 IF(ENDFL2) GO TO 80
106 70 CALL INPLEX(5,INTYP)
107 IF(INTYP.NE.NULL) GO TO 70
C
C ... REVIEW TUSED TO IDENTIFY UNSATISFIED REFERENCES
C PROCESS LISTS SEPARATELY
C
108 80 DO 90 I=1,MAXLIS
109 L1SREF=I1SED(I)
110 IF(L1SREF.EQ.NULL) GO TO 90
111 CALL PARSPFL1SREF,INTYP,IRN,IFN)
112 IF(INTYP.EQ.LLIST) GO TO 90
C
C ... LIST REFERENCED BUT NOT INITIALIZED
C RETURN LIST NODE TO AVAILABLE SPACE LIST
C
113 IRECRF=INVALID(L1SREF)
114 IPTR=IPFF1(L1SREF)
115 ITEM(IPT2)=NULL
116 CALL DFLFFC(IRECRF)
117 CALL EFF1S,3,LLIST,I,I)
118 90 CONTINUE
C

ISN

PLFX PREPROCESSOR -MAIN PROGRAM

C ... LISTS CLEANED UP, NOW DO OTHER RECORDS
C
119 N1=LLIST+1
120 DD 120 IDTYP=N1,NOTYPE
121 N2=LSTREC(IDTYP)
122 IF(N2.EQ.0) GO TO 120
123 JJ=IUSPTP(IDTYP,0,LLIST,NRECS,NULL)
C
C ... IF AVAILABLE SPACE LIST EXISTS, CLEAN IT UP
C
124 92 IF(IAVAIL(IDTYP).EQ.NULL) GO TO 95
125 IRECTP = NEWREF(IDTYP,IAVAIL(IDTYP),0)
126 IAVAIL(IDTYP) = IVAL(JNXT,IRECTP)
127 IRECTP = IREF(1,IRECTP)
128 CALL NUIFY(ITEM(IRECTP),NFLDS(IDTYP))
129 GO TO 92
130 95 DD 110 T=1,N2
131 MVAL=IUSED(JNXT)
132 IF(MVAL.EQ.0) GO TO 110
133 CALL DELREC(NEWREF(IDTYP,T,0))
134 IF(MVAL.EQ.NULL) GO TO 100
C
C ... REFERENCED BUT NOT INITIALIZED
C
135 CALL ERR(15,9,ICTYP,T,IDIUM)
135 GO TO 110
C
C ... RECORD SKIPPED IN SEQUENCE
C
137 100 CALL EP?(16,9,IDTYP,T,IDIUM)
138 110 CONTINUE
139 120 CONTINUE
C
C OUTPUT PLFX STRUCTURE IN EXTERNAL FORM
C
140 DD 150 IDTYP=LLIST,NOTYPE

FLEX PREPROCESSOR - MAIN PROGRAM

```

151      LAST=LAST(LETYP)
152      *ALL AURPLX(LETYP,1, LAST,9,1,1)
153      CONTINUF
C      OUTPUT MAP OF SITE
C      160      CALL MAPSIT
C      OUTPUT BINARY COMMON BLOCKS
C      165      CALL OUTBLK(ITEM1),ITEM(20001),OPTNAM(1),20000,NSIAT=-20000,
C                  +      NPARSQ
C
166      STOP
167      END

```

PLEX PREPROCESSOR - AUTLIS SUBROUTINE

1 SUBROUTINE AUTLIS(LISREF,LUNO)

C C ... OUTLIS IS USED TO PRINT A LIST OF SYMBOLIC
 C C REFERENCES.

C C INPUT PARAMETERS

C LISREF ... A LIST REFERENCE FOR LIST T BE PRINTED
 C LUNO ... LU# FOR OUTPUT

2 COMMON /PAKS/
 3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
 4 REAL*8 DTPNAM,FLDNAM,FORMOT
 5 COMMON IDATA(50),IMAX(50),IMIN(50),IUSED(10100),MAXLIS
 6 DIMENSION XMIN(50),XMAX(50),DATA(50)
 7 EQUIVALENCE (XMAX(1),IMAX(1)), (XMIN(1),IMIN(1))
 8 EQUIVALENCE (DATA(1),IDATA(1))

C C ... IF NOT A LIST RETURN

C I=IFIRST(LISREF,LPTR)
 10 IF (LPTR.EQ.NULL) RETURN
 11 CALL PARSRF(LISREF,IRF,ILIS,1DUM)
 12 MPTR=0
 13 MFLDS=0

C C ... BUILD OUTPUT LIST IN IDATA

C 14 10 IF (I.LT.1000000000) GO TO 20
 15 MFLDS=MFLDS+1
 16 CALL PARSRF(I,1DTYP,IRECNO,IFLD)
 17 MPTR=MPTR+1
 18 DATA(MPTR)=DTPABR(1DTYP)
 19 MPTR=MPTR+1
 20 IDATA(MPTR)=IRECNO
 21 IF (MFLDS.LT.14) GO TO 20

C C ... LINE COMPLETE, OUTPUT IT

PLEX PREPROCESSOR - AUTLIS SUBROUTINE

ISN

C

22 WRITE(LUNO,1000) ILIS, (IDATA(I),I=1,28)

23 MFLDS=0

24 MPTR=0

25 20 I=NEXT(LISREP,LPTR)

26 IF(I.NE.NULL) GO TO 10

C

C ... END OF LIST PRINT LINE

C

27 IF(MFLDS.EQ.0) RETURN

28 WRITE(LUNO,1000) ILIS, (IDATA(I),I=1,MPTR)

29 1000 FORMAT(' ',14,2X,14(A2,I4,2X))

30 RETURN

31 END

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

ISN

```
1      SUBROUTINE AUTPLX(IDTYP,MREC1,MREC2,LUNO,MODE1,MODE2)
C ... A ROUTINE TO OUTPUT PLEX RECORDS ON LOGICAL UNIT SPECIFIED
C BY CALLING PROGRAM.
C
C ... PARAMETERS:
C       IDTYP ... RECORD DATATYPE
C       IREC1 ... 1ST RECORD NUMBER TO BE PRINTED
C       IREC2 ... LAST RECORD NUMBER TO BE PRINTED.
C       MODE1 ... 0 PRINT RECORD WITHOUT HEADINGS
C                  1 PRINT RECORD WITH HEADINGS
C       MODE2 ... 0 DO NOT PRINT REFERENCED LISTS
C                  1 PRINT REFERENCED LISTS
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS3/
9      COMMON IDATA(50),IMAX(50),IMIN(50),IUSED(10100),MAXLIS
10     DIMENSION XMIN(50),XMAX(50),DATA(50)
11     EQUIVALENCE (XMAX(1),IMAX(1)),(XMIN(1),IMIN(1))
12     EQUIVALENCE (DATA(1),IDATA(1))
13     REAL*8 BUF(50),BLANK,FMT(12)
14     DATA BLANK /8H          /
C
C ... INSURE ARGUMENTS IN RANGE
C
15     IREF=NULL
16     IF (IDTYP.LE.LLIST .OR. IDTYP.GT.NDTYPE) RETURN
17     IF (LSTREC(IDTYP).LE.0) RETURN
18     IREC1=MREC1
19     IREC2=MREC2
20     CALL FTNCMD('SET MODECHECK=OFF',17)
21     IRCMAX=NRECS(IDTYP)-NGENRC
```

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

```
1 SN
22      IF(IREC1.LE.0) IREC1=1
23      IF(IREC1.GT.IRCMAX) IREC1=IRCMAX
24      IF(IREC2.GT.IRCMAX) IREC2=IRCMAX
25      IF(IREC2.LT.IREC1) IREC2=IREC1
26      IF(IRECIO(IDTYP).NE.0) GO TO 300
C
C ... INITIALIZATION FOR IRECIO=0
C
27      DO 5 I=1,12
28      5 FMT(I)=FORMOT(I, IDTYP)
29      IF(FMT(1).EQ.BLANK) RETURN
30      NBASE=IREF1(NEWREF(IDTYP, NRECS(IDTYP), 0))
31      NFLD=NFLDS(IDTYP)
C
C ... COUNT NUMBER OF FIELDS TO BE OUTPUT
C     AND MOVE FLDNAMES TO BUF
C
32      NFPTK=IFDNPT(IDTYP)
33      NFLD1=1
34      DO 6 I=1,NFLD
35      IF(ITEM(NBASE+I).GT.3) GO TO 6
36      BUF(NFLD1)=FLDNAM(I+NFPTR)
37      NFLD1=NFLD1+1
38      6 CONTINUE
39      MLIST=0
40      IF(MODE1.EQ.0) GO TO 7
C
C ... PRINT HEADING
C
41      IF(LUNO.EQ.6) WRITE(LUNO,1003) DTPNAM(IDTYP)
42      IF(LUNO.NE.6) WRITE(LUNO,1002) DTPNAM(IDTYP)
43      1002 FORMAT(' ',/,A8)
44      1003 FORMAT(' '/'NEW ',A8,' RECORD:')
45      NN=NFLD1-1
46      WRITE(LUNO,1001) BLANK,(BUF(I),I=1,NN)
C
```

ISN

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

```
C ... START RECORD PROCESSING LOOP.....  
C  
47      7 DO 100 IREC=IREC1,IREC2  
48          IDATA(1)=IREC  
49          NPTR=1  
50          CALL NZERO(IUSED,50)  
51          IPTR=IREF1(NEWREF(IDTYP,IREC,0))  
52          IF(IPTR.EQ.NBASE) GO TO 100  
C  
C ... CHECK FOR NULL RECORD  
C  
53          DO 109 I=2,NFLD  
54          IF(ITEM(IPTR+I).NE.NULL) GO TO 8  
55 109          CONTINUE  
C  
C ... NULL RECORD UNLESS WEAPON OR EQUIPMENT  
C  
56          IF(IDTYP.NE.LWEAP.AND.IDTYP.NE.LEQUIP) GO TO 100  
C  
C ... SPECIAL CHECK FOR WEAPON OR EQUIPMENT  
C  
57          MTYP=ITEM(IPTR+1)  
58          IF(MTYP.LE.0) GO TO 100  
59          IF(IDTYP.EQ.LWEAP.AND.MTYP.GT.NWEPTP) GO TO 100  
60          IF(IDTYP.EQ.LEQUIP.AND.MTYP.GT.NEQPTP) GO TO 100  
C  
C ... IF IRECI0<>0 CALL OUTSET  
C  
61 8          IF(IRECI0(IDTYP).EQ.0) GO TO 9  
62          CALL OUTSET(IDTYP,NBASE,NFLD,NFLD1,IPTR,FMT)  
63          IF(IRTNCD.EQ.IFAIL) GO TO 100  
C  
C ... BUILD RECORD IN IDATA  
C  
64 9          DO 50 I=1,NFLD  
65          ITYP=ITEM(NBASE+I)
```

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

ISN
66 IF (ITYP.GT.3) GO TO 50
67 GO TO (10,20,30), ITYP
C
C ... INTEGER
C
68 10 NPTR=NPTR+1
69 IDATA(NPTR)=ITEM(IPTR+I)
70 IF (IDATA(NPTR).NE.NULL) GO TO 50
71 IDATA(NPTR)=0
72 IUSED(I)=1
73 GO TO 50
C
C ... REAL
C
74 20 NPTR=NPTR+1
75 DATA(NPTR)=DITEM(IPTR+I)
76 IF (DATA(NPTR).NE.NULL) GO TO 50
77 DATA(NPTR)=0.0
78 IUSED(I)=1
79 GO TO 50
C
C ... REFERENCE
C
80 30 IRECRF=ITEM(IPTR+I)
81 IF (IRECRF.NE.NULL) GO TO 35
82 NPTR=NPTR+2
83 IDATA(NPTR)=0
84 IUSED(I)=1
85 GO TO 50
86 35 CALL PARSRF(IRECRF, IDT, IRNO, IFN)
87 NPTR=NPTR+1
88 DATA(NPTR)=DTPAER(IDT)
89 NPTR=NPTR+1
90 IDATA(NPTR)=IRNO
91 IF (IDT.NE.LLIST) GO TO 50
C

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

ISN

```
C ...      SAVE LISREF
C
92          MLIST=MLIST+1
93          IUSED(MLIST+50)=IRECRF
94          50      CONTINUE
C
C ...      PRINT A RECORD
C
95          WRITE(99,FMT) (IDATA(I),I=1,NPTR)
96          READ(99,1000) (BUF(I),I=1,NFLD1)
97          1000 FORMAT(A6,42A8)
98          1001 FORMAT(A6,14A8,,2X,14A8,,4X,14A8)
C
C ... BLANK OUT NULL FIELDS
C
99          J=0
100         DO 70 I=1,NFLD
101             IF(ITEM(NBASE+I) .GT. 3) GO TO 70
102             J=J+1
103             IF(IUSED(I).EQ.0) GO TO 70
104             BUF(J+1)=BLANK
105         70      CONTINUE
106         WRITE(LUNO,1001) (BUF(I),I=1,NFLD1)
107         100 CONTINUE
C ... END RECORD PROCESSING LOOP..... .
C
C ... IF REQUIRED, PRINT REFERENCED LISTS
C
108         IF(MODE2.EQ.0.OR.MLIST.EQ.0) GO TO 250
109         WRITE(LUNO,1002) DTPNAM(LLIST)
110         DO 200 I=1,MLIST
111             CALL AUTLIS(IUSED(I+50),LUNO)
112         200 CONTINUE
113         250 IF(IIREF.NE.NULL) CALL DELREC(IIREF)
114         RETURN
C
```

PLEX PREPROCESSOR - AUTPLX SUBROUTINE

TSN

```
C ... INITIALIZATION FOR IRECIO=1
C
115 300 MLIST=0
116  IIREF=NEWREF(IDTYP,NRECS(IDTYP),0)
117  IIREF=ICOPY(IIREF)
118  NBASE=IREF1(IIREF)
119      NFLDS(IDTYP)
120  IF(MODE1.EQ.0) GO TO 7
C
C ... PRINT HEADING
C
121  N1=IFDNPT(IDTYP)+1
122  N2=N1+NFLDS(IDTYP)-1
123  IF (LUNO.NE.6) WRITE(LUNO,1002) DTPNAM(IDTYP)
124  IF (LUNO.EQ.6) WRITE(LUNO,1003) DTPNAM(IDTYP)
125  WRITE(LUNO,1001) BLANK,(FLDNAM(I),I=N1,N2)
126  GO TO 7
127  END
```

PLEX PREPROCESSOR - DSTAIR SUBROUTINE

1 SUBROUTINE DSTAIR(ISTAIR,X0,Y0,XSCALE,YSCALE)

C ROUTINE TO DRAW IN A REPRESENTATION OF A FLIGHT OF STAIRS ON
C A MAP.

C INPUT PARAMETERS:

C ISTAIR REFERENCE TO A STAIRS RECORD

C X0 X COORDINATE OF WEST SIDE OF BUILDING IN WHICH STAIRS
C ARE LOCATED

C Y0 Y COORDINATE OF SOUTH SIDE OF BUILDING IN WHICH STAIRS
C ARE LOCATED.

C XSCALE SCALING FACTOR TO APPLY IN THE X DIRECTION

C YSCALE SCALING FACTOR TO APPLY IN THE Y DIRECTION

C

2 COMMON /PARS1/
3 COMMON //
4 LOGICAL*1 MAP

C

5 LOGICAL*1 BAR/*|*/,DASH/-*/
6 DATA STRWID/1./

C

C DETERMINE THE DIRECTION OF THE STAIRS

C

7 NBR = VAL(JSTAIR,ISTAIR)
8 XA = VAL(JXCO,ISTAIR)
9 XB = VAL(JXCO,NBR)
10 YA = VAL(JYCO,ISTAIR)
11 YB = VAL(JYCO,NBR)
12 IDIREC = 1
13 X = ABS(XA - XB)
14 IF (X .LT. .01) IDIREC = 2

C

C BRANCH ACCORDING TO ORIENTATION OF THE STAIRS

C

15 IF (IDIREC .EQ. 2) GO TO 20

PLEX PREPROCESSOR - DSTAIR SUBROUTINE

ISN

```
C  STAIRS ORIENTED PARALLEL TO THE X AXIS
16      IF (XB .LT. XA) GO TO 10
17          X1 = XA
18          X2 = XB
19          GO TO 12
20      10      X1 = XB
21          X2 = XA
22      12      MX = (X1-X0)*YSCALE + 1.0
23          NX = (X2-X0)*KSCALE + 1.0
24          MY = (YA-Y0-STRWID/2.)*YSCALE + 1.0
25          NY = (YA-Y0+STRWID/2.)*YSCALE + 1.0
26          DO 15 IY=MY,NY
27              DO 15 IX=MX,NX
28          15      MAP(IX,IY) = BAR
29          RETURN
C
C  STAIRS ORIENTED PARALLEL TO THE Y AXIS
30      20      IF (YB .LT. YA) GO TO 25
31          Y1 = YA
32          Y2 = YB
33          GO TO 30
34      25      Y1 = YB
35          Y2 = YA
36      30      MY = (Y1-Y0)*YSCALE + 1.0
37          NY = (Y2-Y0)*YSCALE + 1.0
38          MX = (XA-X0-STRWID/2.)*YSCALE + 1.0
39          NX = (XA-X0+STRWID/2.)*YSCALE + 1.0
40          DO 35 IY=MY,NY
41              DO 35 IX=MX,NX
42          35      MAP(IX,IY) = DASH
43          RETURN
44          END
```

ISN

PLEX PREPROCESSOR - IBLMAP FUNCTION

```
1      FUNCTION IBLMAP(LX,LY,NC)
C      FUNCTION TO SEE IF A MAP HAS BLANKS IN IT BEGINNING AT (LX,LY)
C      AND CONTINUING FOR NC SPACES.
C
2      COMMON /PARS/
3      EQUIVALENCE (PNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON //
6      LOGICAL*1 MAP
7      LOGICAL*1 L
8      INTEGER I/* */,BL/* */
9      EQUIVALENCE (L,I)
10     IBLMAP = 0
11     LC=LX+NC
12     IF (LC .GT. 120) RETURN
13     IF (LY.LT.1 .OR. LY.GT.72) RETURN
14     DO 10 K=1,NC
15     L = MAP(LX+K-1,LY)
16    10 IF (I .NE. BL) RETURN
17     IBLMAP = 1
18     RETURN
19     END
```

PLEX PREPROCESSOR - INBLK SUBROUTINE

1 SUBROUTINE INBLK(STATV,STATV2,PARS,DATAV,NSTATV,NSTV2,NPARS,
+ NDATAV)

C ROUTINE TO READ LONG BINARY RECORDS PASSED FROM THE INPUT
C PREPROCESSORS.

C INPUT PARAMETERS:

C STATV THE FIRST PORTION OF COMMON /STATEV/
C STATV2 THE SECOND PORTION OF COMMON /STATEV/
C PARS A VECTOR EQUIVALENCED TO /PARS/
C DATAV A VECTOR EQUIVALENCED TO /DATAV/
C NSTATV THE SIZE OF STATV
C NSTV2 THE SIZE OF STATV2
C NPARS THE SIZE OF PARS
C NDATAV THE SIZE OF DATAV

C

2 DIMENSION STATV(NSTATV),STATV2(NSTV2),PARS(NPARS),DATAV(NDATAV)

C

3 READ (4) DATAV
4 READ (5) STATV
5 READ (5) STATV2
6 READ (5) PARS
7 RETURN
8 END

ISN

PLEX PREPROCESSOR - INCHK SUBROUTINE

1

C SUBROUTINE INCHK(IPLARF,N)
C THIS SUBROUTINE IS CALLED BY PLEXPP TO VERIFY THAT AN
C INSIDER HAS BEEN PROPERLY DEFINED BY A USER. IN PARTICULAR
C THE SUBROUTINE CHECKS THAT:

C 1. ALLEGIANCE IS AFE. IF NOT, TYPE IS CHANGED TO
C COMBATANT; ERROR MESSAGE WRITTEN; AND INSIDER COUNT IS
C DECREMENTED. SUBROUTINE RETURNS WITHOUT PERFORMING
C FURTHER CHECKS.

C 2. FOLLOWING FIELDS SHOULD BE NULL: LEADER,
C SUBORDINATES, COMMUNICATION NETS, AND FORCE MEMBERSHIP.
C IF THEY ARE NON-NULL, AN ERROR MESSAGE IS WRITTEN AND
C FIELD IS SET TO NULL.

2

C COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/

11

C
12 DIMENSION NNN(4)
13 REAL*8 TTT(4)
14 DATA TTT /'LEADER','SUBORDS','COMNETS',' FORCE'/
15 1001 FORMAT('0*** PERSON-',I2,'AN INSIDER, IS SFE MEMBER'//
+ ' TYPE CHANGED TO COMBATANT')
16 1002 FORMAT('0*** PERSON-',I2,'AN INSIDER, HAD NON-NULL VALUE'//
+ ' FOR ',A8,'...SET TO NULL')
17 NNN(1) = JLEADR
18 NNN(2) = JSUBS
18 NNN(3) = JCMNTS

PLEX PREPROCESSOR - INCHK SUBROUTINE

ISN

19 NNN(4) = JACSON

20 IPTB = IREF1(IPLARF)

C

21 CALL PARSRF(IPLARF, ID, IREN, IDUM)

22 IF (IVAL(JALLEG, IPLARF) .EQ. KAFE) GO TO 10

23 NN = NN - 1

24 ITEM(IPTR+JTYPE) = KCOMBT

25 WRITE(7,1001) IREN

26 RETURN

C

C ... CHECK FOR NULL FIELDS

C

27 10 DO 20 I=1,4

28 J=NNN(I)

29 IF (IVAL(J,IPLARF) .EQ. NULL) GO TO 20

30 WRITE(7,1002) IREN, TTT(I)

31 ITEM(IPTR+J) = NULL

32 20 CONTINUE

33 RETURN

34 END

ISN

PLEX PREPROCESSOR - INDESC SUBROUTINE

1 SUBROUTINE INDESC(LUNO)
C ... THIS ROUTINE READS RECORD DESCRIPTIONS FROM FILE ASSOCIATED
C WITH LUNO AND STORES DATA IN PLEX STRUCRE ARRAYS: DTPNAM,
C DTPABR,NFLDS,IDLTOFF,IFDNPT. IT ALSO INITIALIZES THE GENERIC
C RECORDS FOR EACH DATA TYPE.
C ... INPUT PARAMETERS
C LUNO ... UNIT FROM WHICH DATA IS TO BE READ
C
C ... THE ARRAY IRECIO IS SET FOR EACH DATATYPE. A VALUE OF
C 0 INDICATES A STANDARD TYPE WITH FIXED FIELDS THAT IS INPUT
C VIA INPLEX. A VALUE OF 1 INDICATES A SPECIAL DATA TYPE
C THAT IS CREATED BY THE MODEL AND HAS VARIABLE OUTPUT CHARACTER-
C ISTICS.
C ... THE PROGRAM STORES MIN, MAX AND DEFAULT VALUES FOR EACH
C FIELD IN EACH RECORD TYPE. IT ALSO STORES THE FIELD TYPE
C AS FOLLOWS*
C 1 = INTEGER FIELD
C 2 = REAL FIELD
C 3 = RECORD REFERENCE FIELD
C 4 = VARIABLE FORMAT FIELD (RECORD WILL HAVE IRECIO=1)
C 5 = VACANT FIELD
C ... DURING PROCESSING OF A DATATYPE DESCRIPTION FOLLOWING ARRAYS
C ARE USED TO STORE DATA UNTIL MOVED TO GENERIC RECORD:
C IDATA ... DEFAULT VALUES
C IMAX ... MAX VALUES
C IMIN ... MIN VALUES
C IUSED ... KEEP TRACK OF FIELDS NOT DEFINED
C
2 REAL*8 BUF(13),BLANK
3 LOGICAL EQUC
4 LOGICAL*1 CHAR1,CHAR2,CHAR3,CHAR4
5 COMMON /STATEV/
6 DIMENSION ITEM(41900),DITEM(41900)
7 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
8 COMMON /PARS/

PLEX PREPROCESSOR - INDESC SUBROUTINE

```

1 ISN
2
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS3/
6      COMMON IDATA(50),IMAX(50),IMIN(50),IUSED(10100),MAXLIS
7      DIMENSION XMIN(50),XMAX(50),DATA(50)
8      EQUIVALENCE (XMAX(1),IMAX(1)), (XMIN(1),IMIN(1))
9      EQUIVALENCE (DATA(1),IDATA(1))
10     1000 FORMAT(12A8,A4)
11     1001 FORMAT(A1)
12     1002 FORMAT(A8,11X,A2)
13     1003 FORMAT(3X,A1,2X,A1,37X,2A1)
14     1004 FORMAT(40X,A1,33X,A6,4X,A6,4X,A6)
15     1005 FORMAT(39X,2I1,33X,F6.0,4X,F6.0,4X,F6.0)
16     1006 FORMAT(39X,2I1,33X,I6,4X,I6,4X,I6)
17     DATA BLANK /8H          /
18
19     #PROC=0
20
21     CALL NULIFY(IDATA,50)
22     CALL NULIFY(IMIN,50)
23     CALL NULIFY(IMAX,50)
24     CALL NZERO(IUSED,50)
25     CALL FTNCMD("SET MODECHECK=OFF",17)
26
27     C
28     C ... READ A RECORD AND STORE IN INTERNAL BUFFER 99
29
30     10 READ(LUNO,1000) EUF
31     WRITE(99,1000) BUF
32     READ(99,1001) CHAR1
33     IF(EQUC('*' ,CHAR1)) GO TO 10
34     IF(EQUC(' ' ,CHAR1)) GO TO 20
35
36     C
37     C ... PROCESS HEADER
38
39     15 IF(NPROC.EQ.0) GO TO 15
40
41     C
42     C ... CONVERT IDATA,IMIN AND IMAX TO GENERIC RECORDS
43

```

PLEX PREPROCESSOR - INDESC SUBROUTINE

```

1 SN
36      NFLDS=NFLDS(IDTYP)
37      NREC=NRECS(IDTYP)
38      NBASE=IREF1(NEWREF(IDTYP,NREC,0))
39      MINBAS=IREF1(NEWREF(IDTYP,NREC-1,0))
40      MAXBAS=IREF1(NEWREF(IDTYP,NREC-2,0))
41      MDFBAS=IREF1(NEWREF(IDTYP,NREC-3,0))
42      DO 12 I=1,NFLD
43      IF (IUSED(I).EQ.0) IUSED(I)=5
44      IF (IUSED(I).EQ.4) IRECIO(IDTYP)=1
45      ITEM(NBASE+I)=IUSED(I)
46      ITEM(MINBAS+I)=IMIN(I)
47      ITEM(MAXBAS+I)=IMAX(I)
48      ITEM(MDFBAS+I)=IDATA(I)
49      12 CONTINUE
C
C ... SET UP FOR NEW DATA TYPE
C
50      15 NDTYPE=NDTYPE+1
51      IDTYP=NDTYPE
52      NPROC=1
53      READ(99,1002) DTPNAM(IDTYP),DTPABR(IDTYP)
54      NFLDS(IDTYP)=0
55      IDTOFF(IDTYP)=IDTOFF(IDTYP-1) + NRECS(IDTYP-1)*NFLDS(IDTYP-1)
56      IFDNPT(IDTYP)=IFDNPT(IDTYP-1) + NFLDS(IDTYP-1)
57      IRCOFF(IDTYP)=IRCOFF(IDTYP-1)+NRECS(IDTYP-1)
58      CALL NZERO(IUSED,50)
59      CALL NULIFY(IDATA,50)
60      CALL NULIFY(IMIN,50)
61      CALL NULIFY(IMAX,50)
62      16 GO TO 10
C
C ... PROCESS FIELD DESCRIPTION RECORDS
C
C ... GET ID NUMBER FIRST
63      20 READ(99,1004) CHAR1
64      READ(99,1005) I1,I2

```

PLEX PREPROCESSOR - INDESC SUBROUTINE

ISN
65 N=I1
66 IF (.NOT.EQUC(' ',CHAR1)) N=10*I1 + I2
C
C ... DETERMINE AND BRANCH ON RECORD TYPE
C
67 READ(99,1003) CHAR1,CHAR2,CHAR3,CHAR4
68 IF(EQUC(' ',CHAR1).AND.EQUC(' ',CHAR2)) GO TO 100
69 IF(EQUC(' ',CHAR1)) GO TO 10
70 IF(EQUC(' ',CHAR4)) GO TO 30
C
C ... TYPE IS REFERENCE
C
71 25 ITYP=3
72 IUSED(N)=ITYP
73 GO TO 55
74 30 IF(EQUC('R',CHAR3)) GO TO 40
75 IF(EQUC('I',CHAR3)) GO TO 50
C
C ... TYPE NOT DESCRIBED
C
76 35 ITYP=4
77 IUSED(N)=ITYP
78 GO TO 55
C
C ... TYPE IS REAL
C
79 40 READ(99,1004) CHAR1,BUF(1),BUF(2),BUF(3)
80 READ(99,1005) I1,I2,X1,X2,X3
81 IUSED(N)=2
82 IF(BUF(1).NE.BLANK) XMIN(N)=X1
83 IF(BUF(2).NE.BLANK) XMAX(N)=X2
84 IF(BUF(3).NE.BLANK) DATA(N)=X3
85 45 GO TO 55
C
C ... TYPE IS INTEGER
C

PLEX PREPROCESSOR - INDESC SUBROUTINE

```

1SN
86      50 READ(99,1004) CHAR1,BUF(1),BUF(2),BUF(3)
87      READ(99,1006) I1,I2,N1,N2,N3
88      IUSED(N)=1
89      IF(BUF(1).NE.BLANK) IMIN(N)=N1
90      IF(BUF(2).NE.BLANK) IMAX(N)=N2
91      IF(BUF(3).NE.BLANK) IDATA(N)=N3
C
C ... UPDATE NFLDS COUNTER
C
92      55 IF(N.GT.NFLDS(IDTYP)) NFLDS(IDTYP)=N
93      IOFF = IFDNPT(IDTYP) + N
94      READ(99,1200) FLDNAM(IOFF)
95      1200 FORMAT(3X,A8)
96      GO TO 10
C
C ... LAST RECORD DESCRIPTION PROCESSED
C
97      100 CALL FTNCMD('SET MODECHECK=ON',16)
C
C ... CONVERT IDATA,IMIN AND IMAX TO GENERIC RECORDS
C
98      NFLD=NFLDS(IDTYP)
99      NREC=NRECS(IDTYP)
100     NBASE=IREF1(NEWREF(IDTYP,NREC,0))
101     MINBAS=IREF1(NEWREF(IDTYP,NREC-1,0))
102     MAXBAS=IREF1(NEWREF(IDTYP,NREC-2,0))
103     MDFBAS=IREF1(NEWREF(IDTYP,NREC-3,0))
104     DO 112 I=1,NFLD
105     IF(IUSED(I).EQ.0) IUSED(I)=5
106     IF(IUSED(I).EQ.4) IRECIO(IDTYP)=1
107     ITEM(NBASE+I)=IUSED(I)
108     ITEM(MINBAS+I)=IMIN(I)
109     ITEM(MAXBAS+I)=IMAX(I)
110     ITEM(MDFBAS+I)=IDATA(I)
111     112 CONTINUE
112     RETURN
113     END

```

PLEX PREPROCESSOR - INITPP SUBROUTINE

1 SUBROUTINE INITPP
C
C ROUTINE TO INITIALIZE VARIABLES FOR THE PLEX PREPROCESSOR.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS3/
C
9 DATA BLANK/' '/
10 CALL NULIFY(ITEM,NSTATE)
11 CALL NULIFY(IAVAIL,NDTYPE)
12 CALL INITVL(FORMOT,24*NDTYPE,BLANK)
13 CALL NZERO(IDTOFF,NDTYPE)
14 CALL NZERO(IRCOFF,NDTYPE)
15 CALL NZERO(IFDNPT,NDTYPE)
16 CALL NZERO(LSTREC,NDTYPE)
17 CALL NZERO(IRECIO,NDTYPE)
18 CALL INITVL(DTPNAM,2*NDTYPE,BLANK)
19 CALL INITVL(FLDNAM,2*NFDXRC,BLANK)
20 CALL INITVL(DTPABR,NDTYPE,BLANK)
21 IRTNCD = IOK
22 ISIDE = 1
23 IPLAYR = 1
24 TMIN = 0.0
25 JFORCE=JACSON
26 RETURN
27 END

ISN

PLEX PREPROCESSOR - INPLEX SUBROUTINE

1 SUBROUTINE INPLEX(LUNO, IDTYP)
C ... INPLEX IS A SUBROUTINE TO INPUT PLEX STRUCTURES IN EXTERNAL
C FORMAT, PERFORM APPROPRIATE DATA CHECKS AND STORE INTERNALLY
C IN /STATEV/
C ... INPUT PARAMETERS:
C LGNO ... LOGICAL UNIT NUMBER FROM WHICH TO GET DATA
C ... OUTPUT PARAMETERS:
C IDTYP ... DATA TYPE OF PLEXRECORDS JUST READ
C OR NULL IF EOF REACHED.
C
C ASSUMPTIONS AND INPUT FORMATS ARE COVERED IN WP #XXXX.XX
C
C ... DATA USED ASSUMED TO BE IN COMMON /PARS/
C DTPNAM(I) IS 8-CHARACTER NAME FOR DATATYPE I (LEFT ADJ, BLANK-FILL)
C
C DTPABR(I) IS 2-CHAR ABBREVIATION FOR DATATYPE I
C
C NGENRC IS NUMBER OF GENERIC RECORDS USED TO DESCRIBE DATATYPE
C RECNO=NRECS(I) SPECIFIES FIELD TYPE 1=INTEGER, 2=REAL, 3=REF
C 4=SPECIAL 5=VACANT
C RECNO=NRECS(I)-1 SPECIFIES MINIMUM ACCEPTABLE VALUE (OR NULL)
C RECNO=NRECS(I)-2 SPECIFIES MAXIMUM ACCEPTABLE VALUE (OR NULL)
C RECNO=NRECS(I)-3 SPECIFIES DEFAULT VALUE (OR NULL)
C
C ... IUSED IS USED TO KEEP TRACK OF RECORDS REFERRED TO AND RECORDS
C ACTUALLY INITIALIZED. (JJ=IUSPTR(IDTYP,IRECNO))
C
C IDTYP RECORD STATUS IUSED(JJ)
C =LIST NOT USED NULL
C REFERENCED IFLDRF FOR FIELD CONTAINING LISTRF
C DEFINED LISTRF
C
C <>LIST NOT USED NULL
C REFERENCED 0
C DEFINED 1

PLEX PREPROCESSOR - IMPLEX SUBROUTINE

ISN

C

```
2      LOGICAL EQUC
3      LOGICAL*1 CHAR
4      REAL*8 DNAME,FMT(12),BLANK,BUF(50)
5      DIMENSION BADNAM(2)
6      EQUIVALENCE(DNAME,BADNAM(1))
7      COMMON /STATEV/
8      DIMENSION ITEM(41900),DITEM(41900)
9      EQUIVALENCE(DTMIN,ITEM(1),DITEM(1))
10     COMMON /PARS/
11     EQUIVALENCE(FNULL,NULL),(IFAIL,FAIL)
12     REAL*8 DTPNAM,FLDNAM,FORMAT
13     COMMON /PARS1/
14     COMMON /PARS2/
15     COMMON /PARS3/
16     COMMON IDATA(50),IMAX(50),IMIN(50),IUSED(10100),MAXLIS
17     DIMENSION XMIN(50),XMAX(50),DATA(50)
18     EQUIVALENCE(XMAX(1),IMAX(1)),(XMIN(1),IMIN(1))
19     EQUIVALENCE(DATA(1),IDATA(1))
20     DATA BLANK/8H      /
```

C

C ... READ A RECORD

C

```
21     IIREF=NULL
22     10 MLIST=0
23     READ(1000) DNAME,FMT
24     WRITE(99,1000) DNAME
25     READ(99,1002) CHAR
26     1002 FORMAT(A1)
27     1000 FORMAT(A8,4X,12A8)
```

C

C ... SKIP BLANK RECORD

C

```
28     IF(EQUC(' ',CHAR)) GO TO 450
29     IF(EQUC('* ',CHAR)) GO TO 10
```

C

JSN

PLEX PREPROCESSOR - INPLEX SUBROUTINE

```
C ... PROCESS FILE HEADER RECORD
C
30      DO 20 I=1,NDTYPE
31      IF(DNAME.NE.DTPNAM(I)) GO TO 20
32      IDTYP=I
C
C ... IF ACTION OR RELATION, SPECIAL PROCESSING
C
33      IF(IDTYP.EQ.LACTN) GO TO 350
34      IF(IDTYP.EQ.LRELN) GO TO 310
35      IF(IRECIO(IDTYP).NE.0) GO TO 22
36      GO TO 30
37      20 CONTINUE
C
C ... INVALID DATATYPE NAME
C
38      22 CALL ERR(8,9,BADNAM(1),BADNAM(2),I)
C
C ... SKIP REMAINDER OF SUBSECTION
C
39      25 READ(LUNO,1000) DNAME
40      IF(DNAME.NE.BLANK) GO TO 25
41      GO TO 10
C
C ... BRANCH ON IDTYP = LLIST
C
42      30 IF(IDTYP.EQ.LLIST) GO TO 200
43      NREC=NRECS(IDTYP)
44      NBASE=IKEF1(NEWREF(IDTYP,NREC,0))
C
C ... PROCESS INPUT RECORD
C
C ... SAVE I/O FORMAT
C
45      DO 35 I=1,12
46      FORMOT(I, IDTYP)=FMT(I)
```

PLEX PREPROCESSOR - INPLEX SUBROUTINE

1SN
47 35 CONTINUE
C
C ... READ RECORD ID TITLE AND FIELD TITLES
C
48 1003 FORMAT(A6,14A8,2X,14A8,4X,14A8)
49 1004 FORMAT(A6,42A8)
50 NFLD=NFLDS(IDTYP)
C
C ... COMPUTE FIELDS TO BE INPUT (TYPE<>5)
C
51 MFLD=NFLD
52 DO 37 I=1,NFLD
53 IF(ITEM(NBASE+I).EQ.5) MFLD=MFLD-1
54 37 CONTINUE
C
C ... READ AND STORE FIELD TITLES
C
55 NFLD1=MFLD+1
56 N1=IFDNPT(IDTYP)
57 KK=MPREAD(NFLD1,BUF,LUNO)
58 IF(KK.NE.0) GO TO 450
59 J=1
60 DO 38 I=1,NFLD
61 IF(ITEM(NBASE+I).EQ.5) GO TO 38
62 J=J+1
63 FLDNAM(N1+I)=BUF(J)
64 38 CONTINUE
C
C ... COMPUTE NUMBER OF DATA IN INPUT RECORD
C FIELD TYPE =1 OR 2 HAVE 1 DATUM PER FIELD
C FIELD TYPE = 3 HAVE 2 DATA PER FIELD
C FIELD TYPE=5 IS NOT INPUT
C
65 NDATA=0
66 DO 40 I=1,NFLD
67 IF(ITEM(NBASE+I).EQ.5) GO TO 40

PLEX PREPROCESSOR - INPLEX SUBROUTINE

15N

68 ITYP=ITEM(NBASE+I)/3
69 NDATA=NDATA+1+ITYP
70 40 CONTINUE
C
C ... SET UP MIN AND MAX POINTERS
C
71 MINBAS=IREF1(NEWREF(IDTYP,NREC-1,0))
72 MAXBAS=IREF1(NEWREF(IDTYP,NREC-2,0))
73 MDFBAS=IREF1(NEWREF(IDTYP,NREC-3,C))
C
C ... READ INPUT RECORD
C
74 42 KK=MREAD(NFLD1,BUF,LUNO)
75 IF(KK.NE.0) GO TO 450
76 WRITE(99,1004) (BUF(I),I=1,NFLD1)
77 43 READ(99,FMT) ID,(DATA(I),I=1,NDATA)
78 IF(ID.EQ.0) GO TO 400
C
C ... CHECK FOR ROOM FOR THIS ID
79 IF(ID.GT.NRECS(IDTYP)-NGENRC) GO TO 44
80 IRECNO=ID
81 JJ=IUSPTR(IDTYP,ID,LLIST,NRECS,NULL)
82 IF(ID.GT.LSTREC(IDTYP)) GO TO 45
83 IF(IUSED(JJ).LE.0) GO TO 50
C
C ... WRITE ERR (DUPLICATE RECORD, IGNORED)
C
84 44 CALL ERR(9,9,IDLTYPE,ID,I)
85 GO TO 42
C
C ... UPDATE LSTREC(IDTYP)
C
86 45 IRECNO=NEWREC(IDTYP)
87 IF(IRECNO.EQ.ID) GO TO 50
88 IF(IRECNO.NE.NULL) GO TO 45
C

PLEX PREPROCESSOR - INPLEX SUBROUTINE

ISN

```
C ... ERROR (NO ROOM IN THE INN, IGNORED
C
89      CALL ERR(3,9, IDTYP, ID, I)
90      GO TO 42
C
C ... START FIELD DATA PROCESSING LOOP
91      50 IUSED(JJ)=1
92      IPTR=IREF1(NEWREF(IDTYP,IRECNO,0))
93      NPTR=1
94      J=1
95      DO 100 I=1,NFLD
96          ITYP=ITEM(NBASE+I)
97          IF (ITYP.GT.3) GO TO 100
98          II=NEWREF(IDTYP,IRECNO,I)
99          J=J+1
100         GO TO (60,70,80), ITYP
C
C ... PROCESS INTEGER FIELD
C
101        60      IF (IDATA(NPTR).NE.0) GO TO 62
102      IF (BUF(J).NE.BLANK) GO TO 68
103      IDATA(NPTR)=ITEM(MDFEAS+I)
104      GO TO 68
C
C ... CHECK RANGE
C
105        62      MIN=ITEM(MINBAS+I)
106      IF (MIN.EQ.NULL) GO TO 64
107      IF (MIN.LE.IDATA(NPTR)) GO TO 64
C
C ... ERROR, VALUE TOO LOW
C
108        64      CALL ERR(10,9,DATA(NPTR),IDTYP,II)
109      MAX=ITEM(MAXBAS+I)
110      IF (MAX.EQ.NULL) GO TO 68
111      IF (MAX.GE.IDATA(NPTR)) GO TO 68
```

ISN

PLEX PREPROCESSOR - INPLEX SUBROUTINE

C
C ... ERROR, VALUE TOO HIGH
C
112 68 CALL ERR(10,9,DATA(NPTR),IDTYP,II)
113 ITEM(IPTR+I)=DATA(NPTR)
114 NPTR=NPTR+1
115 GO TO 100
C
C ... PROCESS REAL FIELD
C
116 70 IF (DATA(NPTR).NE.0.0) GO TO 72
117 IF (BUF(J).NE.BLANK) GO TO 72
118 DATA(NPTR)=DITEM(MDFBAS+I)
119 GO TO 78
C
C ... CHECK RANGE
C
120 72 ZMIN=DITEM(MINBAS+I)
121 IF (ZMIN.EQ.FNULL) GO TO 74
122 IF (ZMIN.LE.DATA(NPTR)) GO TO 74
C
C ... ERROR, VALUE TOO LOW
C
123 74 CALL ERR(10,9,DATA(NPTR),IDTYP,II)
124 ZMAX=DITEM(MAXBAS+I)
125 IF (ZMAX.GE.DATA(NPTR)) GO TO 78
126 IF (ZMAX.EQ.FNULL) GO TO 78
C
C ... ERROR, VALUE TOO HIGH
C
127 78 CALL ERR(10,9,DATA(NPTR),IDTYP,II)
128 DITEM(IPTR+I)=DATA(NPTR)
129 NPTR=NPTR+1
130 GO TO 100
C
C ... PROCESS REFERENCE FIELD

PLEX PREPROCESSOR - INPLEX SUBROUTINE

ISN
C
131 80 IF(BUF(J).NE.BLANK) GO TO 90
132 ITEM(IPTR+I)=NULL
133 NPTK=NPTK+2
134 GO TO 100
C
C ... NON NULL REF VALUE, VALIDATE AND CONVERT TO IRECRF
C
135 90 REF=DATA(NPTR)
136 NUM=1DATA(NPTR+1)
137 NPTK=NPTK+2
C
C ... IF REFERENCE TO LIST, UPDATE MAXLIS
C
138 IF(REF.NE.DTPABR(LLIST)) GO TO 95
139 IF(NUM.GT.MAXLIS. AND .NUM.LE.NRECS(LLIST)) MAXLIS=NUM
140 95 IRECRF=IVALRF(REF,NUM,II)
141 ITEM(IPTR+I)=IRECRF
142 IF(IRECRF.NE.NULL) GO TO 100
C
C ... FIELD REFERENCE ERROR PRINT DTPNAM(IDTYP),ID,REF,NUM
C
143 CALL ERH(11,9,II,REF,NUM)
144 100 CONTINUE
145 105 GO TO 42
C
C ... PROCESS LIST INPUT
C
C ASSUME ALL ARE REC REFERENCES, IGNORE FIELD TITLES
146 200 READ(LUNO,1002) DNAME
C
C ... READ INPUT RECORD
C
147 205 READ(LUNO,1003) (BUF(I),I=1,15)
148 WRITE(99,1004) (BUF(I),I=1,15)
C

ISN

PLEX PREPROCESSOR - INPLEX SUBROUTINE

```
C ... COMPUTE FIELDS IN RECORD
C
149      NFLD=14
150      DO 210 I=1,15
151      IF(BUF(I).NE.BLANK) GO TO 210
152      NFLD=I-2
153      GO TO 220
154      210 CONTINUE
C
C ... IF BLANK RECORD GO TO 400
C
155      220 IF(NFLD.LE.0) GO TO 400
156      NDATA=2*NFLD
157      READ(99,FMT) ID,(IDATA(I),I=1,NDATA)
C
C ... CHECK FOR CONTINUATION
C
158      IF(ID.EQ.MLIST) GO TO 230
C
C ... CHECK FOR PREVIOUSLY ENTERED
C
159      IF(IUSED(ID).EQ.NULL) GO TO 230
160      CALL PARSRF(IUSED(ID),IRT,Irn,IFN)
161      IF(IRT.NE.LLIST) GO TO 230
C
C ... THIS LIST WAS PREVIOUSLY DEFINED
C
162      CALL ERR(9,9,LLIST,ID,I)
163      GO TO 205
C
C ... CONVERT TO LIST OF RECORD REFERENCES
C
164      230 MLIST=ID
165      IF(ID.GT.MAXLIS) MAXLIS=ID
166      NPTR=1
167      NREF=0
```

PLEX PREPROCESSOR - INPLEX SUBROUTINE

TSN

168 DO 250 I=1,NFLD
169 REF=DATA(NPTR)
170 NUM=IDATA(NPTR+1)
171 NPTR=NPTR+2
172 IDUM=IVALRF(REF,NUM,NEWREF(LLIST,ID,0))
173 IF(IDUM.NE.NULL) GO TO 240

C

C ... ERROR PRINT DATATYPE,RECNO,BUF(I+1)

C

174 II=NEWREF(LLIST,ID,0)
175 CALL ERR(11,9,II,REF,NUM)
176 GO TO 250

C

C ... VALID RECREF, ADD TO LIST

C

177 240 NREF=NREF+1
178 IDATA(NREF)=IDUM
179 250 CONTINUE

C

C ... NOW ADD IDATA(I), I=1,NREF TO LIST(ID)

C

180 255 IF(NREF.EQ.0) GO TO 205

C

C ... SEE IF LIST HAS BEEN PREVIOUSLY REFERENCED

C

181 IONE=1
182 LISREF=IUSED(ID)
183 IF(LISREF.NE.NULL) GO TO 260

C

C ... NOT PREVIOUSLY REFERENCED

C

184 LISTNO=NEWREC(LLIST)
185 IF(LISTNO.NE.NULL) GO TO 270

C

C ... NO ROOM IN THE INN

C

PLEX PREPROCESSOR - INPLEX SUBROUTINE

186 CALL ERR(12,9,LLIST, ID, 1)
187 GO TO 205
C
C ... PREVIOUSLY REFERENCED, HAS IT BEEN INITIALIZED
C
188 260 CALL PARSRF(LISREF, IDT, IRN, IFN)
189 IF(IDT.EQ.LLIST) GO TO 280
190 LISREF=IVAL1(LISREF)
191 CALL PARSRF(LISREF, IDT, LISTNO, IFN)
192 GO TO 275
C
C ... HAS NOT BEEN INITIALIZED, UPDATE IUSED AND
C MAKE IDATA(1) THE HEAD OF THE LIST
193 270 LISREF=NEWREF(LLIST, LISTNO, 0)
194 275 IUSED(ID)=LISREF
195 LISTND(JVAL, LISTNO)=IDATA(1)
196 IONE=2
197 IF(NREF.LT.2) GO TO 205
C
C ... ADD OTHER ELEMENTS TO LIST
C
198 280 DO 300 I=IONE,NREF
199 IDUM=IQUEUE(IDATA(I), LISREF)
200 300 CONTINUE
201 GO TO 205
C
C ... SET UP TO PROCESS RELATION RECORDS
C
202 310 READ(LUNO,1002) DNAME
203 315 READ(LUNO,1003) (BUF(I), I=1,14)
204 WRITE(99,1004) (BUF(I), I=1,14)
205 READ(99,FMT) ID,LRNTYP
206 IF(ID.EQ.0) GO TO 400
207 NREC=NRECS(IDTYP)
208 IIREF=NEWREF(IDTYP, NRECS(IDTYP), 0)
209 IIREF=ICOPY(IIREF)

PLEX PREPROCESSOR - INPLEX SUBROUTINE

ISN
210 NBASE=IREF1(LIREF)
211 J=JPART1-1
212 DO 320 I=1,2
213 320 ITEM(NBASE+J+I)=MPELTP(I,LRNNTYP)
214 GO TO 372
C
C ... SET UP TO PROCESS ACTION RECORDS
C
C ... SKIP TITLES
C
215 350 READ(LUNO,1002) DNAME
C
C ... READ FIRST INPUT RECORD TO GET ACTION TYPE
C
216 355 READ(LUNO,1003) (BUF(I),I=1,14)
217 WRITE(99,1004) (BUF(I),I=1,14)
218 READ(99,FMT) ID,LACTYP
219 IF(ID.EQ.0) GO TO 400
220 IF(LACTYP.GE.1 .AND. LACTYP.LE.KCAPNG) GO TO 360
C
C ... ERROR MESSAGE, BAD ACTION TYPE
C
221 GO TO 355
C
C ... SET UP TO USE NORMAL RECORD READ
C
222 360 NREC=NRECS(IDTYP)
223 LIREF=NEWREF(IDTYP,NRECS(IDTYP),0)
224 LIREF=ICOPY(LIREF)
225 NBASE=IREF1(LIREF)
226 J=JPART1-1
C
C ... INSERT CURRENT TYPE IN VARIABLE TYPE FIELDS IN GENRC
C
227 DO 370 I=1,3
228 ITEM(NBASE+J+I)=MACTTP(I,LACTYP)

PLEX PREPROCESSOR - INPLEX SUBROUTINE

ISN

229 370 CONTINUE
C
C ... COMPUTE FIELDS TO BE INPUT
C
230 372 NFLD=NFLDS(IDTYP)
231 NDATA=0
232 MFLD=0
233 DO 375 I=1,NFLD
234 IF(ITEM(NBASE+I).EQ.5) GO TO 375
235 MFLD=MFLD+1
236 ITYP=ITEM(NBASE+I)/3
237 NDATA=NDATA+1+ITYP
238 375 CONTINUE
C
C ... SET UP MIN AND MAX POINTERS
C
239 MINBAS=IREF1(NEWREF(IDTYP,NREC-1,0))
240 MAXBAS=IREF1(NEWREF(IDTYP,NREC-2,0))
241 MDFBAS=IREF1(NEWREF(IDTYP,NREC-3,0))
242 NFLD1=MFLD+1
C
C ... USE MAINLINE CODE FOR REST OF INPUT
C
243 GO TO 43
C
C ... NORMAL TERMINATION OF INPUT OF SINGLE RECORD TYPE
C
244 400 IF(IIREF.NE.NULL) CALL DELREC(IIREF)
245 RETURN
C
C ... END OF FILE TERMINATION, SET IDTYP=NULL
C
246 450 IDTYP=NULL
247 GO TO 400
248 END

PLEX PREPROCESSOR - IUSPTR FUNCTION

ISN

```
1      FUNCTION IUSPTR(IDTYP,IRECNO,LLIST,NRECS,NULL)
C ... THE PURPOSE OF IUSPTR IS TO COMPUTE A POINTER IN ARRAY IUSED
C FOR KEEPING TRACK OF RECORDS REQUESTED AND/OR INITIALIZED IN
C INPLEX. MAKES USE OF THE FACT THAT THIS DATA IS NOT REQUIRED
C FOR IDTYP<=LLIST
C
2      DIMENSION NRECS(1)
3      IUSPTR=NULL
C
C ... FOR IDTYP < LLIST, IUSED IS NOT USED
C
4      IF(IDTYP.LT.LLIST) RETURN
5      N1=LLIST
6      N2=IDTYP-1
7      IUSE=0
8      IF(N2.LT.N1) GO TO 20
9      DO 10 I=N1,N2
10     IUSE=IUSE+NRECS(I)
11     10 CONTINUE
12     20 IUSPTR=IUSE+IRECNO
13     RETURN
14     END
```

PLEX PREPROCESSOR - IVALRF FUNCTION

ISN

1

FUNCTION IVALRF(REF,NUM,IFLDRF)
C ... IVALRF IS CALLED BY INPLEX TO CONVERT SYMBOLIC RECORD
C REFERENCES OF THE FORM "PN 2" OR "# 3" TO A NUMERIC
C RECORD REFERENCE AND UPDATE THE IUSED ARRAY. IF THE
C SYMBOLIC REFERENCE IS IN ERROR OR THE REFERENCE IS AN
C UNALLOWED DUPLICATE A NULL VALUE IS RETURNED. OTHERWISE
C A VALID RECORD REFERENCE IS RETURNED.
C
C ... THE PROGRAM QUERIES AND UPDATES THE ARRAY IUSED.
C PROCESSING DEPENDS ON WHETHER REFERENCE IS TO A RECORD
C OR TO A LIST. (JJ=IUSPTR(IDTYP,IRECNO ...)
C RECORD: IF IUSED(JJ) IS NULL, IT IS SET TO 0 TO INDICATE
C INITIALIZATION EXPECTED.
C
C LIST: IF IUSED(JJ) IS NULL, IT IS SET TO THE FIELD REFERENCE
C THAT REFERS TO THE LIST TO INDICATE INITIALIZATION
C EXPECTED.
C
C IF IUSED(JJ) IS ALREADY A LIST REFERENCE A WARNING
C MESSAGE IS ISSUED INDICATING REFERENCE TO PREVIOUSLY
C DEFINED LIST.
C
C IF IUSED(JJ) IS A RECORD REFERENCE A MULTIPLE LIST
C REFERENCE IS DIAGNOSED, AND A NULL VALUE IS RETURNED.
C
C ... INPUT PARAMETERS:
C REC ... REAL WORD CONTAINING 2-CHAR ABBREVIATION OF
C DATATYPE.
C NUM ... RECORD NUMBER IN INTEGER FORMAT
C IFLDRF ... FIELD REFERENCE TO FIELD WHERE RECORD REFERENCE
C IS TO BE STORED. (USED IF REF="#").
C
2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)

PLEX PREPROCESSOR - IVALRF FUNCTION

```
1SN
4      REAL*8 DTPNAM, FLDNAM, FORMOT
5      COMMON /PARS3/
6      COMMON IDATA(50), IMAX(50), IMIN(50), IUSED(10100), MAXLIS
7      DIMENSION XMIN(50), XMAX(50), DATA(50)
8      EQUIVALENCE (XMAX(1), IMAX(1)), (XMIN(1), IMIN(1))
9      EQUIVALENCE (DATA(1), IDATA(1))

C
C ... SEARCH ABBREVIATION LIST TO FIND IDTYP
C
10     IVALRF=NULL
11     DO 10 I=LLIST,NDTYPE
12     IF (REP.NE.DTPABR(I)) GO TO 10
13     IDTYP=I
14     IF (IDTYP.EQ.LLIST) GO TO 40
15     GO TO 20
16     10 CONTINUE

C
C ... INVALID DATA TYPE ABBREVIATION
C
17     RETURN

C
C ... RECORD REFERENCE, CHECK FOR NUM IN RANGE
C
18     20 IF (NUM.GT.0 .AND. NUM.LE.NRECS(IDTYP)-NGENRC) GO TO 30
C
C ... NUM OUT OF RANGE, ERR MESSAGE
C
19     RETURN

C
C ... PROCESS RECORD REFERENCE
C
20     30 JJ=IUSPTR(IDTYP,NUM,LLIST,NRECS,NULL)
21     IF (IUSED(JJ).EQ.NULL) IUSED(JJ)=0
22     IVALRF=NEWREF(IDTYP,NUM,0)
23     RETURN
```

PLEX PREPROCESSOR - IVALRF FUNCTION

1SN

```
C ... PROCESS LIST REFERENCE, GET RECORD FROM IUSED
C
24      40 IJ=IUSPTR(IDTYP,NUM,LLIST,NRECS,NULL)
25      IUSE=IUSED(JJ)
26      IF(IUSE.NE.NULL) GO TO 60
C
C ... NEW LIST REFERENCE
C
27      LISTNO=NEWREC(LLIST)
28      IF(LISTNO.NE.NULL) GO TO 50
C
C ... NO MORE LIST SPACE LEFT
C
29      RETURN
C
C ... SET IUSED(JJ)=IFLDRF, RETURN LISTRF
C
30      50 IUSED(JJ)=IFLDRF
31      IVALRF=NEWREF(LLIST,LISTNO,0)
32      RETURN
C
C ... THIS IS NOT NEW LIST REFERENCE
C
33      60 CALL PARSRF(IUSE,IRT,IRN,IFN)
34      IF(IRT.EQ.LLIST) GO TO 70
C
C ... PREVIOUSLY REFERENCED, WRITE ERROR MESSAGE
C
35      CALL ERR(13,10,NUM,IUSE,IFLDRF)
36      RETURN
C
C ... PREVIOUSLY DEFINED LIST, WRITE WARNING
C
37      70 CALL ERR(14,10,NUM,IFLDRF,NULL)
38      IVALRF=IUSE
39      RETURN
```

PLEX PREPROCESSOR - TVALRF FUNCTION

ISBN
40

END

ISN

PLEX PREPROCESSOR - KONTNT SUBROUTINE

```
1      SUBROUTINE KONTNT(IRECRF)
C ... THE PURPOSE OF THIS SUBROUTINE IS TO UPDATE THE CONTENTS
C FIELDS OF REGIONS, TO INCLUDE THE ENTITY REFERENCED BY
C IRECRF.
C
C ... IRECRF IS RECORD REFERENCE TO PERSON, VEHICLE, SENSOR
C OR ACTIVATED DELAY
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900), DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(FAIL,PAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS1/
9      COMMON /PARS3/
C
C      CHECK FOR PROPER DATA TYPE
C
10     CALL PARSRF(IRECRF, IDTYP, IRECNO, IDUM)
11     IF (IDTYP .LT. LVEHIC .OR. IDTYP .GT. LACTD) GO TO 100
C
C      GET LOCATION OF IRECRF
C
12     ILOC=IVAL(JLOCN,IRECRF)
13     IF (ILOC .EQ. NULL) GO TO 100
14     IPLAC=IPLACE(ILOC)
15     ICONLT=IVAL(JCONTS,IPLAC)
C
C ... SEE IF IRECRF ON CONTENTS LIST OF IPLAC
C
16     ION=LSEARCH(ICONLT,IRECRF,"NULL,NULL,NULL)
17     IF (ION .NE. NULL) GO TO 100
C
C ... ADD IRECRF TO CONTENTS LIST
```

PLEX PREPROCESSOR - KONTNT SUBROUTINE

ISN

C

```
18      IPTR=IREF1(IPLAC)
19      ICONLT=ISTACK(IRECRP,ICONLT)
20      ITEM(IPTR+JCONTS)=ICONLT
21      100 RETURN
22      END
```

PLEX PREPROCESSOR - LABL SUBROUTINE

1 ISN

1 SUBROUTINE LABL(IRECRF,LX,LY)

C

C ROUTINE TO PUT A LABEL ON A MAP OF THE SITE. THE LABEL CONSISTS
C OF A TWO LETTER ABBREVIATION PLUS AN INDEX. THE LABEL IS CENTERED
C ABOUT A POINT ON THE MAP, (LX,LY).

C

C INPUT PARAMETERS:

C IRECRF A REFERENCE TO THE RECORD WHOSE IDENTIFICATION
C CONSTITUTES THE LABEL.

C LX THE MAP INDEX OF THE X COORDINATE OF THE CENTER OF THE
C LABEL

C LY THE MAP INDEX OF THE Y COORDINATE OF THE CENTER OF THE
C LABEL

C

2 COMMON /PARS/

3 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)

4 REAL*8 DTPNAM,FLDNAM,FORMOT

5 COMMON //

6 LOGICAL*1 MAP

C

7 LOGICAL*1 DIGIT(10)/'1','2','3','4','5','6','7','8','9','0'/,
+ ABBREV(4,30),BL/* /*,LAB(6)

8 DIMENSION IDIGS(4),INCX(13),INCY(

9 EQUIVALENCE (ABBREV(1),DTPABR(1))

10 DATA INCX/0,4,-4,-4,8,0,-8,0,4,8
1 INCY/1,-1,-1,1,1,-2,0,2,1,- ,4,-4/

C

11 CALL PARSRF(IRECRF, IDTYP,IRECNO, IDUM)

12 LAB(1) = ABBREV(1, IDTYP)

13 LAB(2) = ABBREV(2, IDTYP)

C

C BREAK APART RECORD INDEX INTO DIGITS AND STORE IN VECTOR

C

14 1 = 1

15 5 IF (IRECNO .LE. 9) GO TO 10

PLEX PREPROCESSOR - LABL SUBROUTINE

ISN

16 IDIG = IRECHO - 10*(IRECNO/10)

17 IRECNO = IRECNO/10

18 IDIGS(I) = IDIG

19 I = I + 1

20 GO TO 5

21 10 IDIGS(I) = IRECNO

C

C PUT INDEX INTO LABEL

C

22 J = 1

23 15 K = IDIGS(I)

24 IF (K .EQ. 0) K = 10

25 LAB(2+J) = DIGIT(K)

26 IF (I .LE. 1) GO TO 30

27 I = I - 1

28 J = J + 1

29 GO TO 15

C FIND A BLANK AREA OF THE MAP NEAR LX,LY.

30 30 J2 = J + 2

31 LXC = LX - 2

32 LYC = LY

33 DO 100 K=1,13

34 IF (IBLMAP(LXC,LYC,J2) .EQ. 0) GO TO 50

35 IF (LXC .LT. 1) GO TO 50

36 IF (LXC .GT. 120) GO TO 50

37 IF (LYC .LT. 1) GO TO 50

38 IF (LYC .GT. 72) GO TO 50

39 GO TO 200

40 50 LXC = LXC + INCX(K)

41 LYC = LYC + INCY(K)

42 100 CONTINUE

43 RETURN

44 200 CONTINUE

45 DO 220 K=1,J2

46 220 MAP(LXC+K-1,LYC)=LAB(K)

47 RETURN

48 END

ISN

PLEX PREPROCESSOR -MAPRDG SUBROUTINE

1

SUBROUTINE MAPRDG(IBLDG,XSCALE,YSCALE)

C

C ROUTINE TO DRAW A FLOOR PLAN OF EACH OF THE FLOORS OF A BUILDING
C IN THE SITE, INCLUDING ANY PERSONS WHO MAY OCCUPY ANY OF THE ROOMS.

C INPUT PARAMETERS

C IBLDG REFERENCE TO THE BUILDING TO BE MAPPED

C XSCALE NUMBER OF MAP UNITS PER METER IN THE X DIRECTION

C YSCALE NUMBER OF MAP UNITS PER METER IN THE Y DIRECTION

C

2

COMMON /PAPC/

3

FOUTVALENCE (ENULL,NULL), (FAIL,FAIL)

4

REAL*8 DTPNAM,FLENAM,FORMAT

5

COMMON /PAPS1/

6

COMMON /PAPS3/

7

COMMON //

8

LOGICAL*1 MAP

C

9

LOGICAL*1 DASH//--/,PAR//**/,BLANK// * /

10

LOGICAL EDUC

C

11

CALL PAPSREF(IBLDG,ICTYP,IBDGND,IDIUM)

12

WRITE (9,1010) IBDGND

13

1010 FORMAT ('1 MAP OF BUILDING ',I2)

C

C DETERMINE COORDINATES AND DIMENSIONS OF BUILDING

C

14

XLENB = VAL(JXLEN,IBLDG)

15

YLENB = VAL(JYLEN,IBLDG)

16

X0 = VAL(JXCO,IBLDG) - XLENB / 2.0

17

Y0 = VAL(JYCO,IBLDG) - YLENB / 2.0

18

NXR = XLENB*XSCALE + 1.0

19

NYR = YLENB*YSCALE + 1.0

20

NRROWS = NYR

C

PLEX PREPROCESSOR -MAPBDG SUBROUTINE

ISN

```
C ITERATE OVER FLOORS OF BUILDING
C
21      IFLRS = IVAL(IFLPS,IPLDG)
22      IFLR = IFIPST(IFLRS,KF)
23      IFLRNO = 1
24 10      IF (IFLR .EQ. NULL) GO TO 130
C
C WRITE HEADING TO IDENTIFY THE FLOOR
C
25          WRITE (9,1020) IFLRNO,IBDGNO
25 1020      FORMAT ('- FLOOR ',I2,' OF BUILDING ',I2/'0')
C
C PUT IN OUTLINE OF FLOOR
C
27      DO 20 IX=1,NXB
28          MAP(IX,1) = DASH
29 20      MAP(IX,NYB) = DASH
30      DO 25 IY=1,NYB
31          MAP(1,IY) = BAR
32 25      MAP(NXB,IY) = BAR
C
C ITERATE OVER CONTAINED ROOMS, HALLS
C
33      IRROOMS = IVAL(IRROOMS,IFLR)
34      IFLAG = 1
35      IRROMM = IFIPST(IRROOMS,KR)
36 30      IF (IRROMM .EQ. NULL) GO TO 80
C
C DRAW IN WALLS OF ROOM
C
37          XLEN = VAL(IXLEN,IRROOM)
38          XLEN2 = XLEN / 2.0
39          X = VAL(JXCO,IRROOM) - X0
40          MX = (X - XLEN2)*XSCALE + 1.0
41          NX = (X + XLEN2)*XSCALE + 1.0
42          YLEN = VAL(JYLEN,IRROOM)
```

PLFX PREPROCESSOR -MAPRDG SUBROUTINE

```

1SN
43          YLEN2 = YLEN / 2.0
44          Y = VAL(JYCO,IRDM) - YO
45          MY = (Y - YLEN2)*YSCALE + 1.0
46          NY = (Y + YLEN2)*YSCALE + 1.0
47          DO 35 IX=MX,NX
48              IF (EQUC(MAP(IX,MY),BLANK)) MAP(IX,MY) = DASH
49          35      IF (EQUC(MAP(IX,NY),BLANK)) MAP(IX,NY) = DASH
50          DO 40 TY=MY,NY
51              IF (EQUC(MAP(MX,IY),BLANK)) MAP(MX,IY) = BAR
52          40      IF (EQUC(MAP(NX,IY),BLANK)) MAP(NX,IY) = BAR
53
54      C      PUT ROOM INDEX IN RECm
55
56      C      LX = (MX + NX) / 2
57      C      LY = (MY + NY) / 2
58      CALL LABL(IRRECm,LX,LY)
59
60      C      ITERATE OVER PORTALS OF ROOMS, HALLS
61
62      C      NBRs = TVAL(JNBRs,IRDM)
63      C      IPORT = TFIRST(NBRs,KP)
64      45      IF (IPORT .EQ. NULL) GO TO 70
65      CALL PARSF(IPORT,ITDTYP,IDUM,IDUM)
66      IF (ITDTYP .NE. LDOOR .AND. ITDTYP .NE. LWIND) GO TO 60
67
68      C      DRAW TN DOORS & WINDOWS
69
70      C      IDIREC = 1
71      C      IPORT1 = TVAL(IPORT,IPORT)
72      X = AFSEVAL(JYCO,IPORT1) - VAL(JYCO,IPORT1)
73      IF (X .LT. .01) IDIREC = 2
74      CALL MAPORT(IPORT, IDIREC, X0, Y0, XSCALE, YSCALE)
75
76      C      ITERATE OVER ITEMS LOCATED AT WINDOWS AND DOORS
77
78      C      ICOUNTS = TVAL(JCOUNTS,IPORT)

```

PLEX PREPROCESSOR -MAPPDG SUBROUTINE

1SN
57
69 E5 ICNT = IFIRST(ICNTS,KC)
 C IF (ICNT .EQ. NULL) GO TO 50
 C C OUTPUT LABELS DESCRIBING VEHICLE OR PERSON CONTENTS
 C
69 CALL PARSRF(ICNT, IDTYP, IDUM, IDUM)
70 IF (IDTYP .LT. LVEHIC .OR. IDTYP .GT. LACID) GO TO 55
71 CALL CCERPS(IVAL(JLOCN,ICNT),X,Y,Z)
72 LX = (X-X0)*XSCALE + 1.0
73 LY = (Y-Y0)*YSCALE + 1.0
74 CALL LABL(ICNT,LX,LY)
 C END LOOP OVER PORTAL CONTENTS
75 ICNT = NEXT(ICNTS,KC)
75 GO TO 55
 C END LOOP OVER PORTALS
77 E0 IPORT = NEXT(NBRS,KP)
78 GO TO 45
 C
 C ITERATE OVER CONTENTS OF ROOMS
 C
79 70 ICNTS = IVAL(ICNTS,IROOM)
80 ICNT = IFIRST(ICNTS,KC)
81 75 IF (ICNT .EQ. NULL) GO TO 78
82 CALL PARSRF(ICNT, IDTYP, IDUM, IDUM)
83 IF (IDTYP .LT. LVEHIC .OR. IDTYP .GT. LACID) GO TO 77
 C
 C OUTPUT LABELS DESCRIBING VEHICLE OR PERSON CONTENTS
 C
84 CALL CCERPS(IVAL(JLOCN,ICNT),X,Y,Z)
85 LX = (X-X0)*XSCALE + 1.0
86 LY = (Y-Y0)*YSCALE + 1.0
87 CALL LABL(ICNT,LX,LY)
 C END LOOP OVER ROOM CONTENTS
88 77 ICNT = NEXT(ICNTS,KC)
89 GO TO 75
 C END LOOP OVER ROOMS OR HALLS

PLEX PREPROCESSOR -MAPBOD SUBROUTINE

```

15N
50  78      IRROOM = NEXT(IRROOMS,KR)
51      GO TO 30
52      C CHECK WHETHER SHOULD BEGIN LIST OF HALLS OR WHETHER HAVE FINISHED IT.
53  80      IF (IFLAG .GT. 1) GO TO 90
54      IFLAG = 2
55      IRROOMS = IVAL(JHALLS,IFLR)
56      IRCEM = IFIRST(IRROOMS,KR)
57      GO TO 30
C
C   ITERATE OVER STAIRS CONTAINED IN FLOOR
C
58  90      ISTARS = IVAL(ISTARS,IFLR)
59      ISTAIR = IFIRST(ISTAPS,KS)
60      IF (ISTAIR .EQ. NULL) GO TO 120
C
C   DRAW IN STAIRS
C
61 100      CALL DSTAIR(ISTAIR,X0,Y0,XSCALE,YSCALE)
C
C   ITERATE OVER CONTENTS OF STAIRS
C
62 101      ICNTS = IVAL(ICONT,ISTAIR)
63      ICONT = IFIRST(ICNTS,KC)
64 100      IF (ICONT .EQ. NULL) GO TO 110
65      CALL PARSPF(ICONT,IDLV,IDUM,IDUM)
66      IF (IDLV .LT. LVEHIS .OR. IDLV .GT. LACTD) GO TO 100
C
C   OUTPUT LABELS DESCRIBING VEHICLE OR PERSON CONTENTS
C
67 106      CALL CCARDS(IVAL(JLOCN,ICONT),X,Y,Z)
68      LX = (X-X0)*XSCALE + 1.0
69      LY = (Y-Y0)*YSCALE + 1.0
70      CALL LABEL(ICONT,LX,LY)
C   END LOOP OVER STAIRS CONTENTS
71 110      ICONT = NEXT(ICNTS,KC)
72      GO TO 100

```

```

PLIX PROCESSES -MAPING SUBROUTINE

114
115      C END LOOP OVER STAIRS
116      IStAIR = NEXT(IStAIR,KS)
117      GO TO 95
118
119      C JUMP MAP OF FLOOR PRODUCED
120      CALL CUTMAP(MAP,PROJ)
121
122      C FIND END OVER FLOORS OF BUILDING
123      IFLD = NEXT(IFLP,KF)
124      IFLEN = IFLEN + 1
125      GO TO 10
126
127      C
128      130      RETURN
129      END
130

```

PLEX PREPROCESSOR - MAPORT SUBROUTINE

```

1      SUBROUTINE MAPORT(IPORT, IDIREC, X0, Y0, XSCALE, YSCALE)
C
C      ROUTINE TO FILL IN THE PORTAL AS PART OF A SITE MAP.
C
C      INPUT PARAMETERS:
C          IPORT      REFERENCE TO PORTAL TO BE FILLED IN
C          IDIREC     A FLAG INDICATING THE DIRECTION OF THE PORTAL:
C                      =1 IF PORTAL IS ORIENTED HORIZONTALLY; =2 IF PORTAL IS
C                      ORIENTED VERTICALLY; OTHERWISE IDIREC EQUALS 100 * SLOPE
C                      OF THE PORTAL, WHERE "SLOPE" = DELTY / DELTX.
C          X0         X COORDINATE CORRESPONDING TO THE LEFT SIDE OF THE MAP
C          Y0         Y COORDINATE CORRESPONDING TO THE BOTTOM OF THE MAP
C          XSCALE    NUMBER OF MAP UNITS PER METER IN THE X DIRECTION
C          YSCALE    NUMBER OF MAP UNITS PER METER IN THE Y DIRECTION
C
2      COMMON /PARS1/
3      COMMON /PARS3/
4      COMMON //
5      LOGICAL*1 MAP
C
6      LOGICAL*1 DR/'D'/, WI/'W'/, LAB
7      CALL PARSRF(IPORT, IDTYP, IRECNO, IDUM)
8      LAB = DR
9      IF (IDTYP .EQ. LWIND) LAB = WI
10     NBR = IVAL(JPORT, IPORT)
11     IF (IDIREC .EQ. 2) GO TO 70
12     IF (IDIREC .GT. 2) GO TO 80
C
C      PORTAL IS PARALLEL TO X AXIS.
C
13     X = VAL(JYCO, IPORT) - VAL(JHOPIZ, IPORT)/2.
14     Y = (VAL(JYCO, IPORT) + VAL(JYCO, NBR)) / 2.
15     MX = (X-X0)*XSCALE + 1.0
16     NX = (X - X0 + VAL(JHORIZ, IPORT))*XSCALE + 1.0
17     IY = (Y-Y0)*YSCALE + 1.0

```

PLEX PREPROCESSOR - MAPORT SUBROUTINE

ISN

18 DO 65 IX=MX,NX
19 65 MAP(IX,IY) = LAB
20 RETURN

C

C PORTAL IS PARALLEL TO Y AXIS

C

21 70 X = (VAL(JXCO,IPORT) + VAL(JXCO,NBR)) / 2.
22 Y = VAL(JYCO,IPORT) - VAL(JHORIZ,IPORT)/2.
23 MY = (Y-Y0) * YSCALE + 1.0
24 NY = (Y - Y0 + VAL(JHORIZ,IPORT))*YSCALE + 1.0
25 IX = (X-X0)*XSCALE + 1.0
26 DO 75 IY=MY,NY
27 75 MAP(IX,IY) = LAB
28 RETURN

C

C PORTAL ORIENTED OBLIQUELY

C

29 80 SLOPE = FLOAT(IDIREC) / 100.
30 X = (VAL(JXCO,IPORT) + VAL(JXCO,NBR)) / 2.
31 Y = (VAL(JYCO,IPORT) + VAL(JYCO,NBR)) / 2.
32 THETA = ATAN(SLOPE)
33 HLFLEN = VAL(JHORIZ,IPORT) / 2.
34 DX = HLFLEN * COS(THETA)
35 MX = (X - DX - X0)*XSCALE + 1.0
36 NX = (X + DX - X0)*XSCALE + 1.0
37 MY = (Y - Y0 - DX*SLOPE)*YSCALE + 1.0
38 DELTX = 0.
39 DO 90 IX=MX,NX
40 IY = MY + IFIX(DELTX*SLOPE*YSCALE)
41 DELTX = DELTX + 1.0
42 90 MAP(IX,IY) = LAB
43 RETURN
44 END

ISN

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

```
1      SUBROUTINE MAPSIT
C
C      ROUTINE TO DRAW A MAP OF THE SITE, ITS BUILDINGS AND THEIR ROOMS,
C      PORTALS AND CONTENTS, ITS YARDS AND THEIR PERSON OR VEHICLE
C      CONTENTS, AND ITS BARRIERS.
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS3/
7      COMMON //
8      LOGICAL*1 MAP
C
9      LOGICAL*1 DASH/*-*/,BAR/*|*/,STAR/***/
10     DATA BL/*      */ ,NXDIM/30/,NYDIM/72/,GRDFLR/2.8/
C
C      BLANK OUT MAP
C
11    CALL INITVL(MAP,NXDIM*NYDIM,BL)
C
C      GET SITE DIMENSIONS AND DETERMINE SCALE (NUMBER OF MAP UNITS / METER)
C
12    ISITE = NEWREF(LSITE,1,0)
13    X = VAL(JXLEN,ISITE)
14    Y = VAL(JYLEN,ISITE)
15    DMAX = X
16    IF (Y .GT. DMAX) DMAX = Y
17    XSCALE = 119.9 / DMAX
C      MAKE YSCALE LESS THAN XSCALE SINCE A LINE PRINTER PUTS MORE
C      SPACE BETWEEN LINES THAN BETWEEN CHARACTERS ON A SINGLE LINE.
18    YSCALE = 0.6 * XSCALE
C
C      WRITE HEADER
C
```

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

ISN

```
19      WRITE (9,1000)
20 1000 FORMAT (*
C
C   DRAW OUTLINE OF SITE
C
21      NX = K*XSCALE + 1.0
22      NY = Y*YSCALE + 1.0
23      DO 10 IX=1,NX
24          MAP(IX,1) = DASH
25 10      MAP(IX,NY) = DASH
26      DO 20 IY=1,NY
27          MAP(1,IY) = BAR
28 20      MAP(NX,IY) = BAR
29      NROWS = NY
30      GO TO 120
C
C   ITERATE OVER BUILDINGS OF SITE
C
31 25      IBLDGS = IVAL(JBLDGS,ISITE)
32      IBLDG = IFIRST(IBLDGS,KB)
33          BDGMAX = 0.
34 30      IF (IBLDG .EQ. NULL) GO TO 170
C
C   DRAW OUTLINES OF BUILDINGS
C
35      XLEN = VAL(JXLEN,IBLDG)
36      YLEN = VAL(JYLEN,IBLDG)
37      X = VAL(JXCO,IBLDG)
38      Y = VAL(JYCO,IBLDG)
39      MX = (X - XLEN/2.)*XSCALE + 1.0
40      MY = (Y - YLEN/2.)*YSCALE + 1.0
41      NX = (X + XLEN/2.)*XSCALE + 1.0
42      NY = (Y + YLEN/2.)*YSCALE + 1.0
43      DO 40 IX=MX,NX
44          MAP(IX,MY) = DASH
45 40      MAP(IX,NY) = DASH
```

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

1SN

46 DO 45 IY=MY,NY
47 MAP(NX,IY) = BAR
48 45 MAP(NX,IY) = BAR
C
C PUT IN BUILDING INDEX
C
49 LX = (MX + NX) / 2
50 LY = (MY + NY) / 2
51 CALL LABL(IBLDG,LX,LY)
C
C ITERATE OVER CONTAINED WALLS
C
52 IWALLS = IVAL(JWALLS,IBLDG)
53 IWALL = IFIRST(IWALLS,KW)
54 50 IF (IWALL .EQ. NULL) GO TO 110
C DETERMINE DIRECTION OF WALL
55 IDIREC = 1
56 IF (VAL(JXCO1,IWALL) .EQ. VAL(JXCO2,IWALL)) IDIREC = 2
C
C ITERATE OVER CONTAINED PORTALS
C
57 IPORTS = IVAL(JPORTS,IWALL)
58 IPORT = IFIRST(IPORTS,KP)
59 60 IF (IPORT .EQ. NULL) GO TO 100
C
C DRAW IN PORTALS OF GROUND FLOOR
C
60 IF (VAL(JZCO,IPORT) .GT. GRDFLR) GO TO 85
61 CALL MAPORT(IPORT, IDIREC, 0., 0., XSCALE, YSCALE)
C END LOOP OVER PORTALS
62 85 IPORT = NEXT(IPORTS,KP)
63 GO TO 60
C END LOOP OVER WALLS
64 100 IWALL = NEXT(IWALLS,KW)
65 GO TO 50
66 110 SIZ = VAL(JXLEN,IBLDG)

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

ISN

67 IF (SIZ .GT. BDGMAX) BDGMAX = SIZ
68 SIZ = VAL(JYLEN,IBLDG)
69 IF (SIZ .GT. BDGMAX) BDGMAX = SIZ
C END LOOP OVER BUILDINGS
70 IBLDG = NEXT(IBLDGS,KB)
71 GO TO 30
C
C ITERATE OVER BARRIERS OF SITE
C
72 120 IBARS = IVAL(JCBARS,ISITE)
73 IBAR = IFIRST(IBARS,KB)
74 121 IF (IBAR .EQ. NULL) GO TO 25
C
C DRAW IN BARRIERS
C
75 X1 = VAL(JXCO1,IBAR)
76 Y1 = VAL(JYCO1,IBAR)
77 X2 = VAL(JXCO2,IBAR)
78 Y2 = VAL(JYCO2,IBAR)
79 IF (X1 .EQ. X2) GO TO 135
80 SLOPE = (Y2 - Y1) / (X2 - X1)
81 IDIREC = 1
82 IF (SLOPE .GT. 0.1) IDIREC = 100. * SLOPE
83 IF (X1 .GT. X2) GO TO 123
84 MX = X1*XSCALE + 1.0
85 NX = X2*XSCALE + 1.0
86 DX = 0.
87 IF (NX-MX .GT. 0) DX = (X2 - X1) / FLOAT(NX - MX)
88 Y = Y1
89 GO TO 127
90 123 MX = X2*XSCALE + 1.0
91 NX = X1*XSCALE + 1.0
92 Y = Y2
C BARRIER IS HORIZONTAL OR OBLIQUE
93 127 X = 0.0
94 DO 130 IX=MX,NX

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

ISN

95 IY = (Y + X*SLOPE) * YSCALE + 1.0
 96 X = X + DX
 97 130 MAP(IX,IY) = STAR
 98 GO TO 145
 C BARRIER IS VERTICAL
 99 135 IDIREC = 2
 100 IX = X1*XSCALE + 1.0
 101 IF (Y1 .GT. Y2) GO TO 137
 102 MY = Y1*YSCALE + 1.0
 103 NY = Y2*YSCALE + 1.0
 104 GO TO 138
 105 137 NY = Y2*YSCALE + 1.0
 106 NY = Y1*YSCALE + 1.0
 107 138 DO 140 IY=MY,NY
 108 140 MAP(IX,IY) = STAR
 C
 C ITERATE OVER PORTALS OF BARRIER
 C
 109 145 IPORTS = IVAL(JPORTS,IBAR)
 110 IPORT = IFIRST(IPORTS,KP)
 111 150 IF (IPORT .EQ. NUL) GO TO 160
 C
 C DRAW IN PORTALS
 C
 112 CALL MAPORT(IPORT, IDIREC, 0., 0., XSCALE, YSCALE)
 C END LOOP OVER PORTALS
 113 IPORT = NEXT(IPORTS,KP)
 114 GO TO 150
 C END LOOP OVER BARRIERS
 115 160 IBAR = NEXT(IBARS,KB)
 116 GO TO 121
 C
 C ITERATE OVER YARDS OF SITE
 C
 117 170 IYARDS = IVAL(JYARDS,ISITE)
 118 IYARD = IFIRST(IYARDS,KY)

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

ISN

119 175 IF (IYARD .EQ. NULL) GO TO 200
C
C PUT IN YARD INDEX
C
120 LX = VAL(JXCO,IYARD)*XSCALE + 1.0
121 LY = VAL(JYCO,IYARD)*YSCALE + 1.0
122 CALL LABL(IYARD,LX,LY)
C
C ITERATE OVER CONTENTS OF YARDS
C
123 ICONTS = TVAL(JCONTS,IYARD)
124 ICONT = IFIRST(ICONTS,KC)
125 180 IF (ICONT .EQ. NULL) GO TO 190
C
C PUT IN INDEX OF PERSONS OR VEHICLES LOCATED IN YARDS
C
126 LOC = TVAL(JLOCN,ICONT)
127 CALL COORDS(LOC,X,Y,Z)
128 LX = X*XSCALE + 1.0
129 LY = Y*YSCALE + 1.0
130 CALL LABL(ICONT,LX,LY)
C END LOOP OVER CONTENTS OF YARD
131 ICONT = NEXT(ICONTS,KC)
132 GO TO 180
C END LOOP OVER YARDS
133 190 IYARD = NEXT(IYARDS,KY)
134 GO TO 175
C
C DUMP MAP PRODUCED
C
135 200 CALL OUTMAP(MAP,NROWS)
C
C DETERMINE SCALE OF MAXIMUM-SIZED BUILDING
C
136 XSCALB = 119.9 / BDGMAX
137 YSCALB = 0.6 * XSCALB

ISN

PLEX PREPROCESSOR - MAPSIT SUBROUTINE

C
C ITERATE OVER BUILDINGS OF SITE, MAPPING EACH
C
138 IBLDG = IFIRST(IBLDGS,KB)
139 210 IF (IBLDG .EQ. NULL) RETURN
140 CALL MAPBDG(IBLDG,XSCALE,YSCALB)
141 220 IBLDG = NEXT(IBLDGS,KB)
142 GO TO 210
143 END

PLEX PREPROCESSOR - MPREAD FUNCTION

ISN

```
1      FUNCTION MPREAD(NFLD,BUF,LUNO)
C ... MPREAD IS CALLED BY INPLEX TO READ RECORD DATA AND ACCOUNTS
C     FOR THE FACT THAT A LOGICAL RECORD MAY INCLUDE MULTIPLE
C     PHYSICAL (INPUT) RECORDS.  THE FIRST RECORD WILL HAVE ID
C     AND 14 DATA FIELDS OF 8 CHARACTERS.  2ND AND 3RD RECORD
C     WILL HAVE 14 DATA FIELDS OF EIGHT CHARACTERS.
C ... IF FIRST PHYSICAL RECORD IS BLANK, ROUTINE RETURNS
C     WITHOUT READING ANY MORE RECORDS.
C
C ... INPUT PARAMETERS:
C     NFLD ... NUMBER OF FIELDS TO BE READ
C     BUF ... ADDRESS OF BUFFER INTO WHICH DATA IS READ
C     LUNO ... LU# FOR DATA SOURCE
C
C     REAL*8 BUF(1),BLANK
C     DATA BLANK /8H           /
2     1001 FORMAT(A6,14A8)
3     1002 FORMAT(2X,14A8)
4     1003 FORMAT(4X,14A8)
5
6     MPREAD=-1
7     IF(NFLD.LE.0. OR .NFLD.GT.43 ) RETURN
8
9     MPREAD=0
10    READ(LUNO,1001) (BUF(I),I=1,15)
11    IF(NFLD.LT.16) GO TO 15
12    DO 5 I=1,15
13    IF(BUF(I).NE.BLANK) GO TO 10
14    5 CONTINUE
15    RETURN
16    10 READ(LUNO,1002) (BUF(I),I=16,29)
17    IF(NFLD.LT.30) GO TO 15
18    READ(LUNO,1003) (BUF(I),I=30,43)
19    15 MPREAD=0
20    RETURN
21    END
```

PLEX PREPROCESSOR - OUTBLK SUBROUTINE

1 SUBROUTINE OUTBLK(STATV,STATV2,PARS,NSTATV,NSTV2,NPARS)

C ROUTINE TO OUTPUT LONG BINARY COMMON BLOCK SECTION TO THE
C MAIN SIMULATION.

C INPUT PARAMETERS:

C STATV THE FIRST PORTION OF COMMON /STATEV/
C STATV2 THE SECOND PORTION OF COMMON /STATEV/
C PARS A VECTOR EQUIVALENCED TO /PARS/
C NSTATV THE SIZE OF STATV
C NSTV2 THE SIZE OF STATV2
C NPARS THE SIZE OF PARS

C

2 DIMENSION STATV(NSTATV),STATV2(NSTV2),PARS(NPARS)

C

3 WRITE (6) STATV
4 WRITE (6) STATV2
5 WRITE (6) PARS
6 RETURN
7 END

PLEX PREPROCESSOR - OUTMAP SUBROUTINE

1 SUBROUTINE OUTMAP(MAP,NROWS)

C

C ROUTINE TO WRITE A MAP PRODUCED AND TO REINITIALIZE THE MAP'S

C STORAGE TO BLANKS.

C

C INPUT PARAMETERS:

C MAP A MAP OF THE SITE TO BE OUTPUT

C NROWS NUMBER OF ROWS OF MAP TO OUTPUT

C

2 DIMENSION MAP(30,72)

3 DATA BL/* */,NXDIM/30/

C

C WRITE OUT MAP, LINE BY LINE

C

4 WRITE (9,100)

5 100 FORMAT ('-')

6 IROW = NROWS

7 10 WRITE (9,101) (MAP(ICOL,IROW),ICOL=1,NXDIM)

8 101 FORMAT ('9',30A4)

9 IROW = IROW - 1

10 IF (IROW .GE. 1) GO TO 10

C

C REINITIALIZE MAP STORAGE

C

11 CALL INITVL(MAP,NXDIM*NROWS,BL)

12 RETURN

13 END

7.3 Data Preprocessor

This subsection contains listings for the Data Preprocessor. Detailed contents of the common blocks for this program are displayed in section 7.1. The program has no subprograms, utility or otherwise. A detailed discussion of the logic is in section 2.2, in volume I of this manual.

DATA PREPROCESSOR - MAIN PROGRAM

ISBN

C BINARY FORMATTER. READS A FORMATTED FILE OF DATA VARIABLE
C HEADERS FOLLOWED BY DATA FOR EACH VARIABLE, COLLECTS THE DATA INTO
C COMMON BLOCK RECORDS, AND OUTPUTS THE RECORDS IN BINARY FORM. EACH
C VARIABLE CONSISTS OF THE FOLLOWING FIELDS:

NAME	COL 1-8	VARIABLE NAME
DIM 1	9-11	1ST DIMENSION OF VARIABLE IN PROGRAM
DIM 2	12-14	2ND DIMENSION OF VARIABLE IN PROGRAM
DIM 3	15-17	3RD DIMENSION OF VARIABLE IN PROGRAM
STAR	19	*** IF DATA FOR VARIABLE IS ALL ZEROS
RED. DIM 1	20-22	(OPTIONAL) REDUCED DIMENSION APPLYING TO DATA
REPL 1	23	(OPTIONAL) DATA GIVEN ALONG FIRST SECTION OF DIMENSION IS TO BE REPLICATED ALONG LATER SECTIONS IF FIELD CONTAINS "##"
RED DIM 2	24-25	(OPTIONAL) REDUCED DIMENSION 2
REPL 2	26	(OPTIONAL) REPLICATION SYMBOL (#)
RED DIM 3	27-28	(OPTIONAL) REDUCED DIMENSION 3
REPL 3	29	(OPTIONAL) REPLICATION FACTOR 3
FTN FORMAT	31-78	FORTRAN FORMAT (IN PARENTHESES)

C THE DATA VALUES FOR A VARIABLE FOLLOW THE HEADER CARD. FOR
C ARRAYS THE NUMBER OF DATA VALUES EXPECTED IS THE PRODUCT OF THE
C NON-ZERO DIMENSIONS GIVEN ON THE HEADER CARD. IF REDUCED DIMENSIONS
C ARE SPECIFIED THEY ARE USED INSTEAD OF THE ACTUAL DIMENSIONS IN
C COMPUTING THE NUMBER OF EXPECTED DATA VALUES. FOR SCALARS, VARIABLES
C WITH NO DIMENSIONS, A SINGLE VALUE IS EXPECTED. IF A ** IS GIVEN
C IN THE STAR FIELD, ZERO DATA VALUES ARE EXPECTED & THE NEXT HEADER
C CARD SHOULD FOLLOW IMMEDIATELY. THE FORMAT THAT THE DATA VALUES
C APPEAR IN IS UP TO THE USER. THE FORMAT IS INDICATED ON THE HEADER
C CARD & MUST BE A VALID FORTRAN FORMAT SPECIFICATION.

C EACH BLOCK OF DATA VARIABLES IS TERMINATED BY A RECORD WITH
C "ENDBLK" IN COLS 1-6. THE FILE ITSELF IS TERMINATED BY A RECORD WITH
C "ENDFIL" IN COLS 1-6. ALL VARIABLES WITHIN A BLOCK MUST BE IN PROPER
C SEQUENCE & IN 1:1 CORRESPONDENCE WITH THE VARIABLES OF THE ASSOCIATED

DATA PREPROCESSOR - MAIN PROGRAM

ISN

```

C COMMON BLOCK USED IN THE MAIN PROGRAM. SEE THE USER'S MANUAL FOR A
C LIST OF THE INPUT VARIABLES EXPECTED, THEIR ORDER, & THEIR
C DIMENSIONS.
C      THE DATA SET REFERENCE NUMBERS USED ARE AS FOLLOWS:
C      5   INPUT  SANDIA DATA FILE
C      6   OUTPUT OF BINARY DATA BLOCKS
C      7   OUTPUT  ECHO OF DATA FILE HEADER CARDS
C
C
1      REAL VEC(5000),V(1000)
2      REAL*8 VAR,ENDBLK,ENDFIL
3      INTEGER FMT(12)
4      DATA STAR1/1H*,ENDBLK/6HENDBLK/,ENDFIL/6HENDFIL/
5      DATA SHARP/1H*/
6      CALL FTNCMD('SET MODECHECK=OFF',17)
C
7      10     JJ0 = 0
8      11     READ (5,100) VAR,I,J,K,STAR,I2,S1,J2,S2,K2,S3,FMT
9      WRITE (7,101) JJ0,VAR,I,J,K,STAR,I2,S1,J2,S2,K2,S3,FMT
10     100    FORMAT (A8,3I3,1X,A1,I3,2(A1,I2),A1,1X,12A4)
11     101    FORMAT (I6,2X,A8,3I3,1X,A1,I3,2(A1,I2),A1,1X,12A4)
12     IF (VAR .EQ. ENDBLK) GO TO 44
13     IF (VAR .EQ. ENDFIL) GO TO 50
14     JSIZE = 1
15     IF (I .NE. 0) JSIZE = JSIZE*I
16     IF (J .NE. 0) JSIZE = JSIZE*j
17     IF (K .NE. 0) JSIZE = JSIZE*k
18     IF (JSIZE .GT. 5000) STOP 100
19     JJ2 = JJ0 + JSIZE
20     IF (JJ2 .GT. 5000) STOP 200
21     JJ1 = JJ0 + 1
22     14     IF (STAR .EQ. STAR1) GO TO 40
C ***   IF ALL ZEROS, THEN SKIP THIS VARIABLE
23     IF (I .NE. 0) GO TO 31
C ***   IS THIS A SCALAR
24     READ (5,FMT) VEC(JJ1)

```

DATA PREPROCESSOR - MAIN PROGRAM

1 SN
25 GO TO 40
C
26 31 IF (J .NE. 0) GO TO 32
C *** IS IT A VECTOR ?
27 IF (I2 .EQ. 0) I2 = I
28 JJ3 = JJ0 + I2
29 READ (5,FMT) (VEC (M) ,M=JJ1,JJ3)
30 GO TO 40
C
31 32 IF (K .NE. 0) GO TO 36
C *** IS IT A 2-D ARRAY
32 IF (I2 .NE. 0) GO TO 33
33 READ (5,FMT) (VEC (M) ,M=JJ1,JJ2)
34 GO TO 40
35 33 IJ=I2*I2
36 READ (5,FMT) (V (M) ,M=1,IJ)
37 DO 34 I1=1,I2
38 DO 34 J1=1,J2
39 J11 = J1-1
40 M1 = JJ0+I1+I*J11
41 M2 = I1+I2*I11
42 34 VEC (M1) = V (M2)
43 GO TO 40
C
44 36 IF (I2 .NE. 0) GO TO 38
45 READ (5,FMT) (VEC (M) ,M=JJ1,JJ2)
46 GO TO 40
47 38 IJK = I2*I2*K2
48 READ (5,FMT) (V (M) ,M=1,IJK)
49 DO 39 I1=1,I2
50 DO 39 J1=1,J2
51 J11 = J1-1
52 DO 39 K1=1,K2
53 K11 = K1-1
54 M1=JJ0+I1+I*J11+I*K11
55 M2=I1+I2*I11+I2*K11

DATA PREPROCESSOR - MAIN PROGRAM

ISN
56 39 VEC(M1) = V(M2)
C
C
57 40 IREP = 1
58 IF (S1 .EQ. SHARP) IREP = IREP*I
59 IF (S2 .EQ. SHARP) IREP = IREP*I
60 IF (S3 .EQ. SHARP) IREP = IREP*K
61 IF (IREP .EQ. 1) GO TO 43
62 J2SIZE = JSIZE/IREP
63 L1 = JJ0
64 DO 42 ICT=1,IREP
65 L1 = L1 + J2SIZE
66 DO 42 L2=1,J2SIZE
67 42 VEC(L1+L2)=VEC(JJ0+L2)
68 43 JJ0 = JJ0 + JSIZE
69 GO TO 11
C *** WRITE BLOCK INTO SEQUENTIAL FILE
70 44 WRITE(6) (VEC(I), I=1, JJ2)
71 GO TO 10
72 50 STOP
73 END

7.4 Fixed Site Neutralization Model

This subsection contains listings for the Fixed Site Neutralization Model. Detailed contents of the common blocks for this program and its subprograms are displayed in section 7.1. Subprogram listings follow the main program listings. Utility routines, used to process the plex structure and associated lists,¹ are an exception: they are located in section 7.5, because they are also used by the *WPS Preprocessor*.

The reader is referred to section 2.3, in volume 1 of this manual, for the detailed logic descriptions which supplement these listings.

¹That is, the processing described in section 3.3, in volume 1 of this manual.

FSN MODEL - MAIN PROGRAM

ISN

C PROGRAM TO SIMULATE A GROUP OF ADVERSARIES ATTEMPTING TO STEAL
C OR SABOTAGE NUCLEAR MATERAIL AT A NUCLEAR REACTOR SITE AND THEIR
C OPPOSITION BY THE SITE'S GUARDS AND THEIR REINFORCEMENTS. THE
C SIMULATION UPDATES THE STATUS OF THE PLAYERS AT EACH OF A SERIES OF
C SMALL FIXED TIME STEPS UNTIL TERMINATION CONDITIONS ARE REACHED.
C AMONG THE ENTITIES WHICH THE MODEL REPRESENTS ARE THE FOLLOWING:
C THE NUCLEAR FACILITY SITE, INCLUDING ITS OUT-OF-DOORS REGIONS AND
C BARRIERS; ITS BUILDINGS WITH THEIR ROOMS, PORTALS, AND CONTENTS;
C THE GUARDS AND ADVERSARIES WITH THEIR WEAPONS AND EQUIPMENT; VEHICLES
C USED BY PERSONS; SENSORS; AND REMOTE EFFECTORS (ACTIVATED DELAYS).
C THE MODELLING OF PERSONS INCLUDES REPRESENTATION OF THEIR PLANS AND
C ACTIONS, THEIR PERCEPTIONS, AND THE COMMUNICATION OF MESSAGES AMONGST
C THEM.

C THE MODEL AND PROGRAM WERE DEVELOPED FOR SANDIA LABS BY VECTOR
C RESEARCH, INC.

C THIS IS THE MAIN ROUTINE OF THE SIMULATION PROPER. THERE ARE ALSO
C TWO ASSOCIATED PREPROCESSORS FOR READING AND CHECKING THE INPUT.
C THEY EACH GENERATE LONG BINARY RECORDS WHICH ARE READ BY THE MAIN
C SIMULATION.

C THE FOLLOWING IS A SUMMARY OF THE DATA SETS USED BY THE MAIN
C SIMULATION:

C INPUT FILES:

C 4 BINARY DATA FILE
C 5 BINARY PLEX STRUCTURE

C OUTPUT FILES:

C 6 REPORTS OF CHANGES TO STATE VARIABLES
C 7 ERRORS
C 8 PLEX STRUCTURE AT END OF SIMULATION

1 COMMON /STATEV/
2 DIMENSION ITEM(41900),DITEM(41900)
3 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
4 COMMON /PARS/

FSN MODEL - MAIN PROGRAM

```
ISN
5      EQUIVALENCE (NULL,NULL), (FAIL,FAIL)
6      REAL*8 DTPNAM,FLDNAM,FORMOT
7      COMMON /PARS3/
8      COMMON /DATAV/
9      DIMENSION ACTTIM(25)
10     EQUIVALENCE (ACTTIM(1),ACTRAT(1))

C
11    DIMENSION SIDE(2)
12    REAL*8 PROCES(2,10)
13    DATA SIDE/'SPE','AFE'/
14    DATA PROCES/'INITIALI','ZATIONS', 'PERSON D','ECISIONS',
+      'DO MAIN ','ACTIVITY', 'DETECT O','BJECTS',
+      'PROCESS ','OBSVNS', 'PROCESS ','MESSAGES',
+      'NEW TIME','SLICE', 'CHECK FO','R END',
+      'LEADER P','LANNING','BACKUPS ','ARRIVE'/

C
C READ DATA VARIABLES AND INITIAL VALUES OF STATE VARIABLES
C
15    CALL INBLK(ITEM(1),ITEM(20001),DTPNAM(1),ACTRAT(1),20000,
+      NSTATE-20000,NPARSQ,NDATAV)
16    IFLAG=0
C
C INITIALIZATIONS AT THE BEGINNING OF THE SIMULATION
C
17    WRITE (6,100) PROCES(1,1),PROCES(2,1)
18    100 FORMAT (* '/**** ',2A8)
19    CALL INIT
C
C CHECK FOR NEW PLANNING BY LEADERS
C
20    10    WRITE (6,100) PROCES(1,9),PROCES(2,9)
21    CALL LDRPLN
C
C FOR EACH PLAYER, DECIDE NEXT ACTIVITY
C
22    WRITE (6,100) PROCES(1,2),PROCES(2,2)
```

FSN MODEL - MAIN PROGRAM

15N
23 CALL DECIDE
C
C PERFORM MOVEMENT OR FIRING ACTIVITY FOR EACH PLAYER, AS APPROPRIATE
C
24 WRITE (6,100) PROCES(1,3),PROCES(2,3)
25 CALL ACTTY
C
C DETERMINE THE DETECTIONS MADE BY EACH PERSON & SENSOR FOR THE TIME
C STEP
C
26 WRITE (6,100) PROCES(1,4),PROCES(2,4)
27 CALL OBS
C
C UPDATE EACH PLAYER'S OWN PERCEPTIONS BASED ON HIS OWN OBSERVATIONS.
C ADD MESSAGES TO COMMUNICATION NETS TO REPRESENT THE PLAYERS'
C COMMUNICATING THEIR OBSERVATIONS TO OTHERS.
C
28 WRITE (6,100) PROCES(1,5),PROCES(2,5)
29 CALL COMOBS
C
C UPDATE EACH PLAYER'S PERCEPTIONS BASED ON THE COMMUNICATIONS WHICH
C REACH HIM THIS TIME SLICE.
C
30 WRITE (6,100) PROCES(1,6),PROCES(2,6)
31 CALL COMPER
C
C UPDATE THE TIME OF THE SIMULATION
C
32 WRITE(6,101)
33 101 FORMAT(1H1)
34 WRITE (6,100) PROCES(1,7),PROCES(2,7)
35 TMIN = TMIN + DTMIN
36 CALL CHGVAR(1,TMIN)
C
C CHECK FOR THE ARRIVAL OF GUARD REINFORCEMENTS
C

FSN MODEL - MAIN PROGRAM

```
TSN
37      WRITE(6,102)
38      102 FORMAT(' /**** ',32HCHECK FOR RESPONSE FORCE ARRIVAL)
39      IF (IFLAG .EQ. 1) GO TO 13
40      IF (TMIN .GE. TMRESP) GO TO 12
41      WRITE(6,103)
42      103 FORMAT('      RESPONSE FORCE DOES NOT ARRIVE')
43      GO TO 15
44      12 IFLAG=1
45      CALL ARRIVL
46      WRITE(6,104)
47      104 FORMAT('      RESPONSE FORCE DOES ARRIVE')
48      GO TO 15
49      13 WRITE(6,105)
50      105 FORMAT('      RESPONSE FORCE HAS ARRIVED')

C
C      CHECK FOR TERMINATION
C
51      15 WRITE (6,100) PROCES(1,8),PROCES(2,8)
52      IEEND = ITERM(IDUM)

C
C      IF HAVEN'T REACHED THE END OF THE SIMULATION, GO PROCESS ANOTHER
C      TIME SLICE.
C
53      IF (IEEND .GT. 0) GO TO 18
54      WRITE(6,106)
55      106 FORMAT('      SIMULATION CONTINUES')
56      GO TO 10

C
C      MAKE NOTE OF WHEN THE SIMULATION ENDED & WHO WON.
C
57      16 WRITE (7,108) TMIN,SIDE(IEEND)
58      108 FORMAT ('      SIMULATION ENDS AT TIME=',F8.2/A6,' WAS THE WINNER.')
C
C      COLLECT GARBAGE BEFORE OUTPUT OF MAIN STATE VARIABLES
C
59      CALL COLECT
```

FSN MODEL - MAIN PROGRAM

ISN

C
C OUTPUT THE VALUES OF THE MAIN STATE VARIABLES AT THE END OF THE
C SIMULATION.
C
60 DO 20 IDTYP=LLIST,LFORCE
61 CALL OUTPLX(IDTYP,1,LSTREC(IDTYP),8,1,1)
62 20 CONTINUE
C
63 STOP
64 END

FSN MODEL - ACTTY SUBROUTINE

ISN

1

SUBROUTINE ACTTY

C

C

C -- THE PURPOSES OF THIS SUBROUTINE ARE TO:

C 1. UPDATE ALL PLAYERS WITH FIRE OR MOVE
C ACTIVITIES WHICH HAVE RESULTED FROM
C SUBROUTINE DECIDE.

C

C 2. FOR PLAYER FIRING:

- C - FIND THE WEAPON SELECTED AND RATE OF FIRE.
- C - CALCULATE A PROBABILITY OF HIT, AND
C MONTE CARLO SAMPLE FOR HIT OR MISS.
- C - IF A HIT OCCURS, DETERMINE THE DEGREE OF
C DAMAGE.
- C - ASSESS THE FIRE RESULTS AND DECIDE WHEN
C TO TERMINATE FIRING ACTIVITY.

C

C 3. FOR PLAYER MOVING:

C

- C - MOVE EACH PLAYER AS FAR ALONG
C PATH AS POSSIBLE
- C - IF TWO PLAYERS OF OPPOSITE ALLEGIANCE
C INTERSECT PATHS, A CAPTURE SITUATION
C MUST BE RESOLVED

C

C 4. OBSERVATION WILL BE MADE BY EACH PLAYER
C FIRING AND MOVING.

C

2

COMMON /STATEV/

3

DIMENSION ITEM(41900), DITEM(41900)

4

EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

5

COMMON /PARS/

6

EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)

7

REAL*8 DTPNAM, FLDNAM, FORMOT

FSN MODEL - ACTTY SUBROUTINE

1 SN

```

8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /DATAV/
12     DIMENSION ACTTIM(25)
13     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14     COMMON /RECREF/
15     DIMENSION RECRFQ(140)
16     EQUIVALENCE (RECRFQ(1),IGOALS)
17     COMMON /PARS4/
18     COMMON /NEW/
19     COMMON /NEW2/
20     DIMENSION INSTAT(2,5)
21     EQUIVALENCE (IOBSV(41),INSTAT(1,1))

```

C

```

22    NINSDR=5
23    KINSDR=4

```

C

C

```

C   -- SELECT PLAYER IN FIRING ACTIVITY
C

```

```

24    LASMNT=NULL
25    LNODIN(1)=NULL
26    LNODIN(2)=NULL
27    CALL CHGVAR(11,NULL)
28    CALL CHGVRI(13,NULL,1)
29    CALL CHGVRI(13,NULL,2)
30    JL=JGARDS
31    2 IPLALT=IVAL(JI,ISITE)
32          IPLARF=IFIRST(IPLALT,KA)
33    3           IF(IPLARF.EQ.NULL) GO TO 80

```

C

```

C   -- TEST ACTIVITY OF PLAYER
C       IF NOT FIRING NEXT PLAYER
C

```

34

```
IACTLT=IVAL(JACTIV,IPLARF)
```

FSN MODEL - ACTTY SUBROUTINE

1SN
35 IACTRF=IFIRST(IACTLT,KB)
36 IF (IVAL(JTYPE,IACTRF) .NE. KFIRNG) GO TO 70
C
C -- TEST IF PLAYER IS INSIDER
C
37 IF (IVAL(JTYPE,IPLARF) .NE. KNSDR) GO TO 300
C
C -- SET TRIGGER FOR INSIDER EXPOSURE
C
38 DO 298 IK=1,NNSDR
39 IF (INSTAT(1,IK) .EQ. IPLARF) GO TO 299
40 298 CONTINUE
41 GO TO 300
C
C -- TRIGER VALUE IS 1
C
42 299 INSTAT(2,IK)=1
C
C
C -- FIND PHYSICAL STATUS OF PLAYER
C
43 300 ISHOT=KPWHOL
44 IF (IVAL(JPSTAT,IPLARF) .EQ. KWOUND) ISHOT=KPWDWN
C
C -- FIND TARGET REFERENCE, TARGET TYPE, AND PHYSICAL OBJECT TYPE
C
45 DREDU=0.0
46 CALL PARSRF(IPLARF, IDYP, IPLAYR, IDUM)
47 CALL CHGVAR(3,IPLAYR)
48 ISIDE=IVAL(JALLEG,IPLARF)
49 CALL CHGVAR(4,ISIDE)
50 IRF=NULL
51 ITARGT=IVAL(JPAR1,IACTRF)
C
C -- FIND WEAPON REFERENCE AND TYPE
C

FSN MODEL - ACTTY SUBROUTINE

```

ISN
52      IWPNRF=IVAL(JPAR3,IACTR)
53      IWPN=IVAL(JTYPE,IWPNRF)
54      CALL PARSRF(ITARGT, IDYP, IRNO, IDUM)
55      ITYPE=IVAL(JTYPE,ITARGT)
56          IF (IDYP .EQ. LPERSN) ITYPE=1
57          IF (IDYP .EQ. LVEHIC) GO TO 4
58          IF (IDYP .EQ. LPERSN) GO TO 5

C
C -- SENSOR TYPE TARGET
C
59      ITART=KTGSEN
60      IOBJJ=IPHSOB(5)+ITYPE
61      GO TO 6

C
C -- VEHICLE TYPE TARGET
C
62      4      I = IVAL(JTYPE,ITARGT)
63      ITART=IARVTP(I)
64      IOBJJ=IPHSOB(3)+ITYPE
65      CO TO 6

C
C -- PERSON TYPE TARGET
C
66      5      ITART=KTGPSN
67      IOBJJ=IPHSOB(4)+ITYPE
C
C -- REDUCE VULNERABILITY FOR EQUIPMENT CARRIED
C
68      IIELT=IVAL(JWEAPS,ITARGT)
69      IIIGRF=IFIRST(IIELT,KKZ)
70      603     IF (IIIGRF .EQ. NULL) GO TO 6
71      CALL PARSRF(IIIGRF, IDDD, IRRR, IDUM)
72      IF (IDDD .NE. LEQUIP) GO TO 605
73      IF (IVAL(JTYPE,IIIGRF) .EQ. KVEST)
*          DRDNU=DREDU+DEGVUL(IWPN,KSHBUL)
74      IF (IVAL(JTYPE,IIIGRF) .EQ. KGMASK)

```

FSN MODEL - ACTTY SUBROUTINE

```
ISM
      *      DREDU=DREDU+DEGVUL(IWPN,KSHGAS)
75     605 IIGRF=NEXT(IIELT,KKZ)
76     GO TO 603
C
C
C
C      -- DETERMINE IF AREA WEAPON USED
C
77     6 IF(IFIRTP(IWPN) .EQ. 1) GO TO 7
C
C      -- AREA FIRE WEAPON
C      CALCULATE PROBABILITY OF HIT FUNCTION
C
78     SLOP=-1.0/(RNGLET(IWPN)*2.)
79     B=1.0
C
C      -- CALCULATE PROBABILITY OF HIT
C
80     7 RNGE=DIST(IPLARF,ITARGT)
C
C      -- INTERPOLATE STANDARD DEVIATION ARRAY
C
81     RSTEP=(RNGMAX(IWPN)-RNGMIN(IWPN))/FLOAT(NRNGCL-1)
82     SIGL=FCNLIN(PSIG(1,ITART,IWPN),NRNGCL,RNGE,RSTEP)
83     SIGR=FCNLIN(PSIG(NRNGCL+1,ITART,IWPN),NRNGCL,RNGE,RSTEP)
C
C      -- ADJUST SIGMAS BY SKILL
C
84     SIGL=SIGL*SKILL(KFIRSK,ISHOT,IPLAYR)
85     SIGR=SIGR*SKILL(KFIRSK,ISHOT,IPLAYR)
C
C      -- GENERATE RANDOM NUMBERS
C
86     10 CALL GAUSS(IRN(3),SIGL,0.,RNLAT)
87     CALL GAUSS(IRN(4),SIGR,0.,RNRAD)
C
```

FSN MODEL - ACTTY SUBROUTINE

ISN
C -- TEST IF AREA FIRE WEAPON USED
C
88 IF(IFIRTP(IWPN) .EQ. 2) GO TO 40
C
C-- DETERMINE IF HIT OR MISS
C
89 EXPOS=1.0
C
C -- IF TARGET TYPE IS PERSON FIND POSTURE AND COVER
C
90 IF(ITART .NE. KTGPSN) GO TO 15
91 CALL DIREC(ITARGT,IPLARF,THET,PE)
92 TEM=COVR(ITARGT,THET)
93 EXPOS=VAL(JPOSTR,ITARGT)-TEM
C
C -- SET AREA OF PERSON
C
94 IF (EXPOS .GE. 0.8) GO TO 11
95 IF(EXPOS .GE. 0.5) GO TO 12
96 IF(EXPOS .GE. 0.2) GO TO 13
97 X1=0.3
98 X2=-1.0
99 Y1=0.3
100 Y2=-1.0
101 GO TO 14
102 11 X1=0.3
103 X2=0.5
104 Y1=1.5
105 Y2=0.3
106 GO TO 14
107 12 X1=0.3
108 X2=0.5
109 Y1=1.0
110 Y2=0.3
111 GO TO 14
112 13 X1=0.5

FSN MODEL - ACTTY SUBROUTINE

ISN
113 X2=-1.
114 Y1=0.
115 Y2=-1.0
C
C -- TEST FOR HIT
C
116 14 IPHS=0
117 IF(ABS(RNLAT) .LE. X1 .AND. ABS(RNRAD) .LE. Y1) GO TO 19
118 IF(ABS(RNLAT) .GT. X2) GO TO 50
119 IF(RNRAD .GT. ((Y1/2.)-Y2)) GO TO 50
120 IF(RNRAD .LT. -(Y1/2.)) GO TO 50
121 GO TO 19
C -- FIND AREA OF EXPOSURE
C USE PHYSICAL OBJECT TYPE
C
122 15 CCAR=AREA(IOBJJ)*EXPOS
C
C -- ASSUME AREA IS A CIRCLE,
C DETERMINE IF (RNLAT,RNRAD) LIES
C WITHIN CIRCLE
C
123 AHT=PI*SQRT(RNLAT**2+RNRAD**2)
124 IF(CCAR .LT. AHT) GO TO 50
C
C -- SET RESULTS OF HIT TO MISS
C
125 18 IPHS=0
C
C -- HAVE RECORDED A HIT
C DETERMINE RESULTS OF HIT
C
126 19 RNN=URAND(IRN(5))
C
C -- TEST IF AIM TO WOUND
C
127 IF(IAIMPT(ISIDE) .EQ. 1) GO TO 22

FSN MODEL - ACTTY SUBROUTINE

1SN
128 IF (IAIMPT (ISIDE) .EQ. 2) GO TO 21
C
C -- AIM TO KILL
C
129 PP1=PWOUND (ITART,IWEN)
130 PP2=PK (ITART,IWPN)*(1.-DREDU)
131 GO TO 23
C
C -- AIM TO WOUND
C
132 21 PP1=PWOUND (ITART,IWEN)
133 PP2=PK (ITART,IWPN)*0.25
134 GO TO 23
C
C -- AIM TO WARN
C
135 22 PP1=0.0
136 PP2=0.0
137 23 IF (RNN .LT. PP1) IPHS=KWOUND
138 IF (RNN .LT. PP2) IPHS=KDEAD
C
C -- TEST RESULTS OF HIT
C
139 IF (IPHS .EQ. 0) GO TO 50
C
C -- ADJUST IPHS TO REFLECT PHYSICAL STATUS OF
C TARGET TYPE
C
140 IF (ITART .EQ. KTGPSN) GO TO 30
141 IF (ITART .EQ. KTGSEN) GO TO 20
C
C -- VEHICLE TYPE TARGET
C
142 IF (IPHS .EQ. KWOUND) IPHS=KIMMOB
143 IF (IPHS .EQ. KDEAD) IPHS=KDESTR
144 GO TO 25

PSN MODEL - ACTTY SUBROUTINE

ISN

```
C
C -- SENSOR TARGET
C
145    20 IF(IPHS .EQ. KWOUND) IPHS=KDESTR
146        IF(IPHS .EQ. Kdead) IPHS=KDESTR
C
C -- ADJUST PHYSICAL STATUS
C
147    25 IPTR=IREF1(ITARGT)
148        ITEM(IPTR+JPSTAT)=IPHS
149        GO TO 50
C
C ADJUST PHYSICAL STATUS OF PERSON
C
150    30 ICOND=IVAL(JPSTAT,ITARGT)
151        IF(ICOND .EQ. Kdead) GO TO 50
152        IF(IPHS .EQ. KWOUND .AND. ICOND .EQ. KWOUND) IPHS=Kdead
153        IF(IPHS .EQ. ICOND) GO TO 50
C
C ADJUST FORCES
C
154    IIT=KSFE
155    IF(ISIDE .EQ. KSFE) IIT=KAFE
156    FORCES(ICOND,IIT)=FORCES(ICOND,IIT)-1.0
157    FORCES(IPHS,IIT)=FORCES(IPHS,IIT)+1.0
158    CALL CHGVR2(5,FORCES(ICOND,IIT),ICOND,IIT)
159    CALL CHGVR2(5,FORCES(IPHS,IIT),IPHS,IIT)
160    IPTR=IREF1(ITARGT)
161    ITEM(IPTR+JPSTAT)=IPHS
162    IF(IPHS .EQ. Kdead) CALL DEAD(ITARGT)
163    GO TO 50
C
C -- AREA EFFECTS WEAPON
C FIND LOCATION OF CHARGE
C
```

FSN MODEL - ACTTY SUBROUTINE

164 40 IILOC=IVAL(JLOCN,ITARGT)
 165 CALL COORDS(IILOC,X,Y,Z)
 166 X=X+RNLAT
 167 Y=Y+RNRAD
 C
 C -- SEARCH PLAYER LIST
 168 JL=JGA^DS
 169 41 IIPLT=IVAL(JL,ISITE)
 170 IIPL=IFIRST(IIPLT,KZ)
 171 42 IF(IIPL .EQ. NULL) GO TO 46
 C
 C-- FIND RANGE FROM CHARGE
 C
 172 IILOC=IVAL(JLOCN,ITART)
 173 CALL COORDS(IILOC,X1,Y1,Z1)
 174 DD=SQRT((X1-X)**2+(Y1-Y)**2)
 175 PHH=SLOP*DD+B
 176 IF(PHH .LE. 0.) GO TO 44
 C
 C -- SAMPLE FOR HIT
 C
 177 RNN=URAND(IRN(6))
 178 IF(RNN .LE. PHH) GO TO 18
 179 44 IIPLT=NEXT(IIPLT,KZ)
 180 GO TO 42
 C
 C -- NEXT PLAYER LIST
 C
 181 46 IF(JL .EQ. JADVRS) GO TO 55
 182 JL=JADVRS
 183 GO TO 41
 C
 C -- ASSESSMENT OF RESULTS
 C
 184 50 CALL FASMT(IPLARF,ITARGT,IPHS,IEF)

C

PSN MODEL - ACTTY SUBROUTINE

ISN
C -- DETERMINE IF AREA FIRE WEAPON
C
185 IF(IFIRTP(IWPN) .EQ. 2) GO TO 44
C
C -- UPDATE PLAYER TIME AND AMMUNITION
C
186 55 IPTRPL=IREF1(IPLARF)
187 IACS=NACTSU(IPLARF,NULL)
188 TEM=ACTTIM(IACS)
189 IF(IACTTP(IACS) .EQ. 1) TEM=1./ACTRAT(IACS)
190 TEM=TEM*SKILL(KFRTSK,ISHOT,IPLAYER)
191 DITEM(IPTRPL+JTREQ)=DITEM(IPTRPL+JTREQ)-TEM
192 IF(DITEM(IPTRPL+JTREQ) .LT. 0.0) DITEM(IPTRPL+JTREQ)=0.0
193 IPTR=IREF1(IWPNR)
194 ITEM(IPTR+JAMT)=ITEM(IPTR+JAMT)-1
C
C -- DECIDE TO FIRE AGAIN
C
195 ICODE=NDRFIR(IPLARF,ITARGT,IACTR,IRF)
C
196 IF(ICODE .EQ. 1) GO TO 10
C
C --- PLAYER DECIDES NOT TO FIRE AGAIN
C LET PLAYER OBSERVE WHILE FIRING
C
197 TT=DTSEC-VAL(JTREQ,IPLARF)
198 DD=DIREC(IPLARF,ITARGT,THET,PHE)
199 CALL OBSERV(IPLARF,TT,KFIRNG,THET)
C
C -- ADJUST SITUATIONS OF TARGET AND PLAYER
C
200 60 IPTEMP=IPLARF
201 63 CALL PARSRF(IPTEMP, IDY, IRNO, IDUM)
202 IF(IDY .NE. LPERSN) GO TO 65
203 IPTR=IREF1(IPTEMP)
204 ITEM(IPTR+JSITN)=KSITN4

FSN MODEL - ACTTY SUBROUTINE

ISN

205 65 IF(IPTEMP .EQ. ITARGT) GO TO 70
206 IPTEMP=ITARGT
207 GO TO 63

C

C -- NEXT PLAYEL

C

208 70 IPLARF=NEXT(IPLALT,KA)
209 GO TO 3

C

C -- TEST IF LIST OF PLAYERS COMPLETE

C

210 80 IF(JL .EQ. JADVRS) GO TO 90
211 JL=JADVRS
212 GO TO 2

C

C -- FIRING UPDATE COMPLETE

C UPDATE PLAYERS MOVING

C

213 90 JL=JGARDS

214 105 IPLALT=IVAL(JL,ISITE)

C

215 IPLARF=IFIRST(IPLALT,KN)

216 108 IF(IPLARF .EQ. NULL) GO TO 200
217 IF(IVAL(JPSTAT,IPLARF) .LE. KCAPTR) GO TO 190

C

C -- FIND ACTIVITY REFERENCE

C

218 IACTLT=IVAL(JACTIV,IPLARF)
219 IF(IACTLT .EQ. NULL) GO TO 190
220 IACTRFL=IFIRST(IACTLT,KD)
221 CALL PARSRF(IPLARF, IDYP, IPLAYR, IDUM)
222 CALL CHGVAR(3, IPLAYR)
223 ISIDE=IVAL(JALLEG,IPLARF)

C

C -- TEST FOR MOVING ACTIVITY, IF NOT - NEXT PLAYER

C

FSN MODEL - ACTTY SUBROUTINE

ISN
224 IF(IVAL(JTYPE,IACTR) .NE. KMOVNG) GO TO 190
C
C -- IDENTIFY LIST OF NODES(PATH)
C
225 IPTRPL=IREF1(IPLARF)
C
C -- FIND PLAYER POSITION
C
226 ILOC=IVAL(JLOCN,IPLARF)
227 IPTR=IREF1(ILOC)
228 ISOU=IVAL(JSOURC,ILOC)
229 IPPRF=IVAL(JPLACE,ILOC)
C
C -- FIND FIRST NODE ON PATH
C
230 120 LSRM=IVAL(JPAR2,IACTR)
231 ISIN=IFIRST(LSRM,KR)
232 IF(ISIN .EQ. NULL) GO TO 185
C
C -- TEST IF PLAYER AT NODE ON PATH
C
233 IF(IVAL(JSINK,ILOC) .NE. ISOU) GO TO 130
234 DITEM(IPTR+JFRAC)=0.0
235 IF(ISOU .NE. ISIN) GO TO 130
C
C -- REMOVE NODE FROM LIST
C
236 CALL DELIST(KR,JPAR2,IACTR)
237 GO TO 120
C
C -- FIND SUBACTIVITY FOR MOVING
C
238 130 IACTS=NACTSU(IPLARF,NULL)
C
C -- FIND TRAVEL TIME FOR THIS PLAYER
C FROM ISOU TO ISIN

FSN MODEL - ACTTY SUBROUTINE

TSN

C

239 ISUM=NULL

240 RTIM=TRAVEL(IPLARF,ISOU,ISIN,IACTS,ISUM)

C

C TEST IF TRAVELING THROUGH PORTAL

C

241 IF (ISUM .NE. NULL) CALL POROPN(IPLARF,ISOU,ISIN,ISUM)

C

C -- DETERMINE TIME NEEDED FOR THIS LINK

C

242 RTIM=RTIM*(1.0-VAL(JFRAC,ILOC))

C

C -- DETERMINE IF TIME IS AVAILABLE TO

C TRAVEL OVER LINK

C

243 TA=VAL(JTREQ,IPLARF)

244 TO=RTIM

245 IF ((TA-RTIM) .LT. 0.) TO=TA

C

C -- DETERMINE DIRECTION FOR OBSERVATION

C

246 CALL DIREC(ISOU,ISIN,THET,PHE)

C

C -- LET PLAYER OBSERVE WHILE MOVING

C

247 135 IF (TO .LE. 0.0) GC TO 136

248 CALL OBSERV(IPLARF,TO,KMOVNG,THET)

C

C -- DETERMINE IF LINK IS PASSED

C

249 136 IOLDRF=ICOPY(ILOC)

250 IF (TO .EQ. TA) GO TO 150

C

C -- LINK PASSED

C UPDATE PLAYER DESCRIPTION

FSN MODEL - ACTTY SUBROUTINE

ISN
251 140 DITEM(IPTRP+JTREQ)=DITEM(IPTRP+JTREQ)-RTIM
252 ITEM(IPTR+JSOURC)=ISIN
253 ITEM(IPTR+JSINK)=ISIN
254 DITEM(IPTR+JFRAC)=0.0
255 IT=IPLACE(ILOC)
256 CALL CHGFLD(JPLACE,ILOC,IT,1,1DUM)
C
C -- REMOVE ISIN FROM LSRM
C
257 CALL DELIST(KR,JPAR2,IACTR)
258 GO TO 160
C
C -- DID NOT PASS LINK
C
259 150 DITEM(IPTRP+JTREQ)=0.0
C
C -- UPDATE LOCATION ON LINK
C
260 ITEM(IPTR+JSINK)=ISIN
261 DITEM(IPTR+JFRAC)=DITEM(IPTR+JFRAC)+
1 (TA/RTIM)*(1.-DITEM(IPTR+JFRAC))
262 IT=IPLACE(ILOC)
263 CALL CHGFLD(JPLACE,ILOC,IT,1,1DUM)
264 GO TO 170
C
C -- PORTAL STATUS
C
265 160 CALL PORTST(IPLARF,IOLDR)
C
C -- SENSOR DETECTION
C
266 170 CALL SNSACT(IPLARF,IOLDR)
C
C -- CHECK FOR CAPTURE SITUATION
C
267 CALL CAPDET(IPLARF,IOLDR)

FSN MODEL - ACTTY SUBROUTINE

ISN

```
C
C   -- DETERMINE IF MOVEMENT ACTIVITY COMPLETE
C
268      IF (VAL (JTREQ, IPLARF) .LE. 0.0) GO TO 188
C
C   -- PLAYER STILL HAS TIME TO MOVE
C
269      ISOU=ISIN
270      GO TO 120
C
C   -- PLAYER COMPLETED MOVEMENT
C
271      185  CONTINUE
C
C   -- UPDATE CONTENTS OF REGIONS
C
272      188 CALL CONTNT (IPLARF, IPPRF)
C
C   -- NEXT PLAYER
C
273      190 IPLARF=NEXT (IPLALT, KN)
274      GO TO 108
C
275      200 IF (JL .EQ. JADVRS) GO TO 220
276      JL=JADVRS
277      GO TO 105
C
278      220 CONTINUE
279      RETURN
280      END
```

PSN MODEL - ADDVAL SUBROUTINE

ISN

```
1      SUBROUTINE ADDVAL(IREF,P,IARRAY,PARRAY,LCUR,ILOW,PMIN,LMAX)
C ... THE PURPOSE OF THIS ROUTINE IS TO MAINTAIN TWO PARALLEL
C ARRAYS, WHERE IARRAY CONTAINS INFORMATION, PARRAY CONTAINS
C AN ASSOCIATED PRIORITY. DATA IS ADDED TO THE ARRAYS IF THEY
C ARE NOT FULL. IF THE ARRAYS ARE FULL, DATA IS ONLY ADDED IF
C ITS PRIORITY IS HIGHER THAN THE LOWEST PRIORITY IN PARRAY.
C IN THIS CASE, IT REPLACES THE LOW PRIORITY ITEM.
C
C INPUT PARAMETERS:
C     IREF    ... DATUM TO BE ADDED
C     P       ... PRIORITY OF DATUM
C     IARRAY  ... DATA ARRAY
C     PARRAY  ... PARALLEL PRIORITY ARRAY
C     LCUR    ... NUMBER OF ENTRIES CURRENTLY IN ARRAY
C     ILOW    ... INDEX FOR LOW PRIORITY ENTRY
C     PMIN    ... PRIORITY OF LOW PRIORITY ENTRY
C     LMAX    ... MAXIMUM NUMBER OF ENTRIES ALLOWED IN ARRAYS
C
C OUTPUT PARAMETERS: (UPDATED BY SUBROUTINE)
C     IARRAY, PARRAY, LCUR, ILOW, PMIN
C
2      DIMENSION IARRAY(1),PARRAY(1)
3      IF(LCUR.EQ.LMAX) GO TO 10
C
C ... LCUR < LMAX, UNCONDITIONAL ADD
C
4      LCUR=LCUR+1
5      PARRAY(LCUR)=P
6      IARRAY(LCUR)=IREF
7      IF(P.GE.PMIN) RETURN
C
C ... UPDATE ILOW, PMIN
C
8      PMIN=P
9      ILOW=LCUR
```

FSN MODEL - ADDVAL SUBROUTINE

```
1SN
10      RETURN
C
C ... ARRAY FULL, ADD ONLY IF P>PMIN
C
11  10  IF (P.LE.PMIN) RETURN
12      IARRAY (ILOW)=IREF
13      PARRAY (ILOW)=P
14      PMIN=1000.0
C
C ... FIND NEW ILOW, PMIN
C
15      DO 20 I=1,LMAX
16          IF (PARRAY (I).GT.PMIN) GO TO 20
17          PMIN=PARRAY (I)
18          ILOW=I
19  20  CONTINUE
20      RETURN
21      END
```

FSN MODEL - ARRIVAL SUBROUTINE

ISN

1

SUBROUTINE ARRIVAL

C

C THE PURPOSE OF THIS SUBROUTINE IS TO LINK THE
C RESPONSE FORCE PLAYERS TO THE CURRENT ACTIVE PLAYERS
C LIST. AT THIS POINT THE RESPONSE FORCE HAS ARRIVED.
C THE CONTENTS OF ALL REGIONS IS UPDATED ALONG WITH
C ADDING TO THE FORCES ARRAY.

C

C

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900), DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
7 REAL*8 DTPNAM, FLDNAM, FORMOT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /DATAV/
11 DIMENSION ACTTIM(25)
12 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
13 COMMON /RECREF/
14 DIMENSION RECRFQ(140)
15 EQUIVALENCE (RECRFQ(1),IGOALS)

C

C

C -- ADD RESPONSE FORCE TO GUARDS

C

16 TMRESP=TFIN
17 JL=JGARDS
18 ISIDE=KSFE
19 IPTR=IREF1(ISITE)
20 IF (IRSITE .EQ. NULL) GO TO 70
21 10 IPLRST=IVAL(JL,IRSITE)
22 IPLARF=IFIRST(IPLRST,KA)

FSN MODEL - ARRIVAL SUBROUTINE

ISN C
23 20 IF(IPLARF .EQ. NULL) GO TO 50
24 CALL PARSRF(IPLARF, IDY, IPLAYR, IDUM)
C
C -- UPDATE CONTENTS
C
25 ILOC=IVAL(JLOCN, IPLARF)
26 IPLAC=IPLACE(ILOC)
27 IF(IPLAC .EQ. NULL) GO TO 30
28 IPPT=IREF1(IPLAC)
29 LISTRR=IVAL(JCONTS, IPLAC)
30 ITEM(IPPT+JCONTS)=ISTACK(IPLARF, LISTRR)
C
C -- UPDATE FORCES
C
31 30 IF(JL .EQ. JVEHS) GO TO 40
32 IPHS=IVAL(JPSTAT, IPLARF)
33 FORCES(IPHS, ISIDE)=FORCES(IPHS, ISIDE) + 1.0
C
C -- ADD PLAYER TO ISITE PLAYERS LIST
C
34 40 LISTRF=IVAL(JL, ISITE)
35 ITEM(IPTR+JL)=IQUEUF(IPLARF, LISTRF)
C
C -- NEXT PLAYER
C
36 IPLARF=NEXT(IPLRST, KA)
37 GO TO 20
C
C -- END OF LIST
C
38 50 IF(JL .GE. JADVRS) GO TO 60
39 JL=JADVRS
40 ISIDE=KAFE
41 GO TO 10
C

PSN MODEL - ARRIVAL SUBROUTINE

ISN

42 60 IF(JL .GE. JVEHS) GO TO 70
43 JL=JVEHS
44 GO TO 10

C

45 70 IRSITE=NULL
46 RETURN
47 END

FSN MODEL - BRECH SUBROUTINE

1 ISN

SUBROUTINE BRECH(IPLARP,IPORT,IWPNRF)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. DETERMINE IF A PLAYER HAS A WEAPON
C CAPABLE OF BREECHING THE PORTAL

C

C -- INPUT PARAMETERS

C

C IPLARP -- PLAYER REFERENCE

C

C IPORT -- PORTAL REFERENCE

C

C -- OUTPUT VARIABLES

C

C IWPNRF -- WEAPON REFERENCE

C

C

C -- PROCEDURE

C 1. DETERMINE PENETRABILITY OF PORTAL
C 2. SEARCH EQUIPMENT TO DETERMINE IF WEAPON EXISTS
C TO DESTROY PORTAL

C

2 COMMON /PARS/

3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)

4

REAL*8 DTPNAM,FLDNAM,FORMOT

5

COMMON /PARS1/

6

COMMON /PARS3/

7

COMMON /DATAV/

8

DIMENSION ACTTIM(25)

9

EQUIVALENCE (ACTTIM(1),ACTRAT(1))

C

C

C -- FIND PENETRABILITY AND TARGET TYPE

C

10

IPEN=IVAL(JPEN,IPORT)

FSN MODEL - BRECH SUBROUTINE

1 ISN
2 ITAR=KTGPO3
3 IF (IPEN .GE. 4) ITAR=KTGPO6
4 IF (IPEN .GE. 7) ITAR=KTGPO7
5 C
6 C -- SEARCH EQUIPMENT LIST
7 C
8 ITT=99
9 IWPNRF=NULL
10 IEQLT=IVAL (JWEAPS,IPLARF)
11 IWPN=IFIRST (IEQLT,KA)
12 10 IF (IWPN .EQ. NULL) GO TO 50
13 C
14 C -- IGNORE EQUIPMENT
15 C
16 CALL PARSRF (IWPN, IDY, IR, ID)
17 IF (IDY .NE. LWEAP) GO TO 30
18 C
19 C -- TEST AMMUNITION SUPPLY
20 C
21 IF (IVAL (JAMT, IWPN) .LE. 0) GO TO 30
22 ITYP=IVAL (JTYPE, IWPN)
23 C
24 C -- FIND WEAPON RANKING FOR TARGET TYPE
25 C
26 IF (IRNKWP (ITYP, ITAR) .LE. 0) GO TO 30
27 IF (IRNKWP (ITYP, ITAR) .GE. ITT) GO TO 30
28 ITT=IRNKWP (ITYP, ITAR)
29 IWPNRF=IWPN
30 C
31 30 IWPN=NEXT (IEQLT, KA)
32 GO TO 10
33 C
34 50 CONTINUE
35 RETURN
36 END

* ISN FSN MODEL - CAPDET SUBROUTINE

1 SUBROUTINE CAPDET(IPLARF,ILOCRF)
C
C -- THE PURPOSE OF THIS SUBROUTINE IS TO:
C 1. DETERMINE IF ANY TWO PLAYERS HAVE
C OCCUPIED THE SAME LOCATION
C 2. AND RESOLVE ANY CAPTURES THAT OCCUR
C
C -- INPUT PARAMETERS
C
C IPLARF -- PLAYER REFERENCE
C ILOCRF -- OLD LOCATION REFERENCE
C
C -- OUTPUT
C
C THIS SUBROUTINE ADJUSTS THE ACTIVITY RECORDS
C OF ANY PLAYERS IN CAPTURE.
C ALSO, A LIST OF ALL NODES PASSED
C IS KEPT TO CHECK FOR CAPTURES
C
C -- PROCEDURE
C 1. SEARCH LIST OF PLAYERS OF OPPOSITE
C ALLEGIENCE
C
C - TO FIND ANY PLAYER FIRING OR OBSERVING
C THAT IPLARF CROSSED OVER
C
C 2. SEARCH NOD OF IMPACT LIST FOR
C INTERSECTION OF PLAYERS MOVING
C
C 3. ADD NEW NODE TO LIST OF IMPACT
C NODES

2 COMMON /STATEV/

FSN MODEL - CAPDET SUBROUTINE

```
1SN
 3      DIMENSION ITEM(41900),DITEM(41900)
 4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
 5      COMMON /PARS/
 6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
 7      REAL*8 DTPNAM,FLDNAM,FORMOT
 8      COMMON /PARS1/
 9      COMMON /PARS2/
10      COMMON /PARS3/
11      COMMON /DATAV/
12      DIMENSION ACTTIM(25)
13      EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14      COMMON /RECREF/
15      DIMENSION RECRFQ(140)
16      EQUIVALENCE (RECRFQ(1),IGOALS)
17      COMMON /PARS4/
18      DIMENSION INSTAT(2,5)
19      EQUIVALENCE (IOBSV(41),INSTAT(1,1))

C
20      NINSDR=5
21      KINSDR=4

C
C
C      -- TEST IF PLAYER IS A INSIDER
C
22      IF (IVAL(JTYPE,1PI,1F) .EQ. KINSDR) GO TO 250
C
C      -- DESCRIBE IPLA      MOVEMENT AS
C          (A,B,P)
C
23      ILOC=IVAL(JLOCN,IPLARF)
24      IF (IVAL(JSOURC,ILOCRF) .EQ. IVAL(JSINK,ILOCRF)) GO TO 10
C
C      -- MOVEMENT CASE 1
C
25      ICASE=1
26      ISIN=IVAL(JSOURC,ILOCRF)
```

FSN MODEL - CAPDET SUBROUTINE

ISN
27 ISOU=IVAL(JSOURCE, ILOC)
28 PRO=1.-VAL(JFRAC, ILOCRF)
29 GO TO 40
C
30 10 IF (IVAL(JSOURCE, ILOC) .EQ. IVAL(JSINK, ILOC)) GO TO 20
C
C -- MOVEMENT CASE 2
C
31 ICASE=2
32 ISOU=IVAL(JSOURCE, ILOC)
33 ISIN=IVAL(JSINK, ILOC)
34 PRO=VAL(JFRAC, ILOC)
35 GO TO 40
C
C -- MOVEMENT CASE 3
C
36 20 ISOU=IVAL(JSOURCE, ILOCRF)
37 ISIN=IVAL(JSOURCE, ILOC)
38 PRO=1.0
39 ICASE=3
C
C -- SEARCH LIST OF PLAYERS OF OPPOSITE ALLEGIANCE
C
40 40 JL=JGARDS
41 IF (IVAL(JALLEG, IPLARF) .EQ. KSFE) JL=JADVRS
42 INODE=ISOU
43 IPLT=IVAL(JL, ISITE)
C
44 IPRF=IFIRST(IPLT, KA)
45 50 IF (IPRF .EQ. NULL) GO TO 100
C
C -- IGNORE INSIDERS
C
46 IF (IVAL(JTYPE, IPRF) .EQ. KNSDR) GO TO 70
C
C IGNORE DEAD OR CAPTURED PLAYERS

394

FSN MODEL - CAPDET SUBROUTINE

ISN

C

47 IF (IVAL (JPSTAT, IPRF) .LE. KCAPTR) GO TO 70

C

48 IARF=IVAL (JACTIV, IP1)

49 IF (IARF .EQ. NULL) GO TO 70

50 IF (IVAL (JTYPE, IARF) .EQ. KMOVNG) GO TO 70

51 IPLOC=IVAL (JLOCN, IPRF)

C

C -- TEST FOR SAME LINK

C

52 I1=IVAL (JSOURC, IPLOC)

53 I2=IVAL (JSINK, IPLOC)

54 IF (I1 .EQ. I2) GO TO 55

55 TPRO=PRO

56 IF (I1 .EQ. ISOU .AND. I2 .EQ. ISIN) GO TO 60

57 TPRO=1.-PRO

58 IF (I1 .EQ. ISIN .AND. I2 .EQ. ISOU) GO TO 60

59 GO TO 70

C

C TEST IF SINK IS I2

C

60 55 IF (ISIN .NE. I2) GO TO 70

61 INODE=ISIN

62 GO TO 180

C

C -- SAME LINK FOR BOTH PLAYERS

C

C FIND IF ON SAME PART OF LINK

C

63 60 IF (VAL (JFRAC, IPLOC) .GE. TPRO) GO TO 180

C

C -- NEXT PLAYER

C

64 70 IPRF=NEXT (IPLT, KA)

65 GO TO 50

C

FSN MODEL - CAPDET SUBROUTINE

ISN

```
C -- TEST MOVEMENT CASE FOR NODE PASSED
C
66    100 IF(ICASE .EQ. 2) GO TO 300
C
C -- NODE PASSED
C
67    INODE=IVAL(JSOURC,ILOC)
C
C -- DETERMINE IF ANY PLAYER
C     OF OPPSITE ALLEGIANCE OCCUPIED
C     THIS NODE
C
68    IAL=1
69    IF(IVAL(JALLEG,IPLARP) .EQ. KSPE) IAL=2
C
C -- FIND PLAYER TIME
C
70    TTIM=(DTSEC-VAL(JTREQ,IPLARP))/60.+TMIN
71    ILIS=IFIRST(LNODIN(IAL),KB)
72    140    IF(ILIS .EQ. NULL) GO TO 140
73    IF(IVAL(JID,ILIS) .NE. INODE) GO TO 120
74    IPRF=IVAL(JCONTS,ILIS)
C
C -- CHECK TIME FOR LOCATION
C     CURRENT RULE IF WITHIN TIME STEP,
C     CAPTURE EXISTS
C
75    IF((VAL(JPAR1,ILIS)-TTIM) .LE. DTSEC) GO TO 150
C
C -- NEXT NODE OF IMPACT
C
76    120 ILIS=NEXT(LNODIN(IAL),KB)
77    GO TO 110
C
C -- ADD NODE TO LIST
C
```

FSN MODEL - CAPDET SUBROUTINE

1SN
78 140 II=IVAL(JALLEG,IPLARF)
79 IRF = NEWRC5(LINODE,inode,TTIM,NULL,IPLARF,NULL)
C
C -- ADD RECORD TO LIST
C
80 IALL=IVAL(JALLEG,IPLARF)
81 CALL CHGLST(IRF,1,9,IALL)
82 LNODIN(IALL)=ISTACK(IRF,LNODIN(IALL))
83 GO TO 300
C
C -- CAPTURE EXISTS
C RESOLVE BETWEEN IPLARF AND IPRF
C BOTH PLAYERS MOVING
C
C
C -- RULE
C FIRST PLAYER TO REACH NODE CAPTURES
C SECOND PLAYER TO REACH NODE
C
84 150 IF(VAL(JPAR1,ILIS) .LT. TTIM) GO TO 155
85 IWIN=IPLARF
86 ILOS=IPRF
87 GO TO 160
C
88 155 IWIN=IPRF
89 ILOS=IPLARF
C
90 160 IP1=NEWREF(1,1,JTMIN)
91 P2=TMIN+ACTTIM(KCPTRG)/60.
92 IFF = NEWRC3(LRELN,KGEF,IP1,P2)
93 CALL CREAT(KCAPNG,ILOS,NULL,NULL,IFF,IWIN,IRF)
C
C -- ADD CAPTURE TO ILOS ACTIVITIES
C
94 ISLT=IVAL(JACTIV,ILOS)
95 ISLT=ISTACK(IRF,ISLT)

PSN MODEL - CAPDET SUBROUTINE

ISN
96 IPTR=IREF1(ILOS)
97 ITEM(IPTR+JACTIV)=ISLT
C
C -- CALL ERROR FOR RECORD
C
98 CALL ERR(32,29,IWIN,ILOS,INODE)
99 GO TO 200
C
C -- CAPTURE EXISTS
C IPLARF MOVING, IPRF STATIONARY
C
C RULE
C IF IPRF OBSERVING IN DIRECTION OF IPLARF
C IPRF WINS
C
100 IACLT=IVAL(JACTIV,IPRF)
101 IARF=IFIRST(IACLT,KS)
102 IF (IVAL(JTYPE,IARF) .EQ. KFIRNG) GO TO 185
103 IF (IVAL(JTYPE,IARF) .EQ. KDETNG) GO TO 188
C
C -- ACTIVITY IS CAPTURING, DETERMINE
C IF BEING CAPTURED
104 IF (IVAL(JPAR1,IARF) .EQ. IPRF) GO TO 100
C
C -- IPRF IS CAPTURING ANOTHER PLAYER
C
105 IWIN=IPLARF
106 ILOS=IPRF
107 GO TO 160
C
C -- FIRING ACTIVITY
C
108 IT=IVAL(JPAR1,IARF)
109 CALL DIREC(IPLARF,IT,THET,PHE)
110 GO TO 190
C

FSN MODEL - CAPDET SUBROUTINE

188 ISN
C -- OBSERVING ACTIVITY
C
111 188 THET=VAL(JPAR1,IARF)
C
112 190 IF(THET .EQ. NULL) GO TO 193
113 IDD=ISOU
C IF(ICASE .EQ. 1) IDD=ISIN
114 CALL DIREC(IPRF,IDD,TT,PH)
115 IF(TT .GT. (THET-PI/4.) .AND. TT .LT. (THET+PI/4.))
* GO TO 195
C
C -- IPLARF WINS IPRF LOSES
C
116 193 IWIN=IPLARF
117 ILOS=IPRF
118 GO TO 160
C
119 195 IWIN=IPRF
120 ILOS=IPLARF
121 GO TO 160
C
C -- DETERMINE IF LOSING PLAYER IN CAPTURE
C PRIOR TO THIS CAPTURE
C
122 200 IACLT=IVAL(JACTIV,ILOS)
123 IACRF=IFIRST(IACLT,KC)
124 210 IF(IACRF .EQ. NULL) GO TO 225
125 IF(IVAL(JTYPE,IACRF) .NE. KCAPNG) GO TO 220
C
C -- TEST IF BEING CAPTURED
C
126 IF(IVAL(JPAR1,IACRF) .EQ. ILOS) GO TO 220
127 IPP=IVAL(JPAR1,IACRF)
128 ITT=IREF1(IPP)
129 ITEM(ITT+JACTIV)=NULL
130 DITEM(ITT+JTREQ)=DTSEC

FSN MODEL - CAPDET SUBROUTINE

ISN
131 CALL DELIST(KC,JACTIV,ILOS)
132 GO TO 300
C
133 220 IACRF=NEXT(IACLT,KC)
134 GO TO 210
C
C REMOVE ALL IMPACT NODES FROM LIST FOR IWIN ILOS
C
135 225 DO 230 IAL=1,2
136 IPLS=NULL
137 ILST=IFIRST(LNODIN(IAL),KP)
138 IF(ILST .EQ. NULL) GO TO 229
139 IF(IVAL(JCONTS,ILST) .EQ. IWIN) GO TO 227
140 IF(IVAL(JCONTS,ILST) .EQ. ILOS) GO TO 227
141 IPLS=ISTACK(ILST,IPLS)
C
142 227 ILST=NEXT(LNODIN(IAL),KP)
143 GO TO 226
C
144 229 LNODIN(IAL)=IPLS
145 230 CONTINUE
C
C
C UPDATE LOCATION FOR PLAYERS
C
146 IPER=IWIN
147 232 ILOC=IVAL(JLOCN,IPER)
148 IPTR=IREF1(ILOC)
149 ITEM(IPTR+JSOURC)=INODE
150 ITEM(IPTR+JSINK)=INODE
151 DITEM(IPTR+JFRAC)=0.0
152 ITEM(IPTR+JPLACE)=INODE
153 IPTR=IREF1(IPER)
154 DITEM(IPTR+JTREQ)=0.0
155 IF(IPER .EQ. ILOS) GO TO 300
156 IPER=ILOS

FSN MODEL - CAPDET SUBROUTINE

ISN
157 GO TO 232
C
C -- INSIDER
C CAN NOT CAPTURE OR BE CAPTURED
C
C TEST IF IN SAME REGION AS AN AFE MEMBER
C
158 250 ILOC=IVAL(JLOCN,IPLARF)
159 IINSL=IVAL(JPLACE,ILOC)
160 IPLT=IVAL(JADVRS,ISITE)
C
161 IPRF=IFIRST(IPLT,KA)
162 255 IF(IPRF .EQ. NULL) GO TO 300
C
163 ILOC=IVAL(JLOCN,IPRF)
164 IPLAC=IVAL(JPLACE,ILOC)
165 IF(IPLAC .EQ. IINSL) GO TO 260
166 IPRF=NEXT(IPLT,KA)
167 GO TO 255
C
C -- SET TRIGGER
C
168 260 DO 265 IK=1,NINSDR
169 IF(INSTAT(1,IK) .EQ. IPLARF) GO TO 270
170 265 CONTINUE
171 GO TO 300
C
C -- TRIGGER VALUE IS 3
C
172 270 INSTAT(2,IK)=3
C
C
173 300 CONTINUE
174 RETURN
175 END

FSN MODEL - CHKOUT SUBROUTINE

1 SUBROUTINE CHKOUT(IOBJ,LOC,LEADER,IFORCE)
C
C ROUTINE CALLED BY A LEADER TO DISPATCH ONE OR MORE PERSONS
C UNDER HIS COMMAND TO INVESTIGATE AN ALARM OR TO LOOK FOR A MISSING
C PERSON.
C THE CURRENT VERSION OF THE ROUTINE JUST SCHEDULES A MOVE TO
C THE LOCATION OF INTEREST BY EACH OF THE PERSONS DESIGNATED.
C
C INPUT PARAMETERS:
C IOBJ OBJECT TO LOOK FOR
C LOC PLACE TO MOVE TO
C LEADER REFERENCE TO LEADER GIVING ORDERS
C IFORCE REFERENCE TO LIST OF PERSONS WHO ARE TO PERFORM
C THE ACTION
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMAT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
C
C ITERATE THROUGH PLAYERS WHO ARE TO INVESTIGATE
11 IPERSN = IFIRST(IFORCE,KP)
12 10 IF (IPERSN .EQ. NULL) RETURN
C
C CREATE INSTRUCTIONS FOR HIM TO MOVE
13 IACT = NEWRC6(LACTN,KMOVNG,LOC,NULL,NULL,NULL,IPERSN)
14 IMSG = NEWRC5(LMESS,LEADER,IPERSN,JPLANS,TMIN,IACT)
C SEND MESSAGE TO THE PLAYER
15 CALL COMMO(LEADER,IPLRSN,IMSG)
C GET NEXT PLAYER

FSN MODEL - CHKOUT SUBROUTINE

15
16 IPERSN = NEXT(IFORCE,KP)
17 GO TO 10
18 END

FSN MODEL - COMMO SUBROUTINE

ISN

1 SUBROUTINE COMMO(IPERSN,IADRES,IMESS)

C ROUTINE TO TRANSMIT AN ALREADY SPECIFIED MESSAGE TO AN INDIVIDUAL.

C INPUT PARAMETERS:

C IPERSN A REFERENCE TO THE PERSON SENDING THE MESSAGE.

C IADRES A REFERENCE TO THE PERSON TO WHOM THE MESSAGE IS

C ADDRESSED.

C IMESS THE MESSAGE TO BE COMMUNICATED.

2 COMMON /STATEV/

3 DIMENSION ITEM(41900), DITEM(41900)

4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

5 COMMON /PARS/

6 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)

7 REAL*8 DTPNAM,FLDNAM,FORMOT

8 COMMON /PARS1/

C C IF IPERSN EQUAL IADRES, UPDATE PLAYER RECORD DIRECTLY

9 IF (IPERSN .NE. IADRES) GO TO 5

10 MCONT=IVAL(JCONT,IMESS)

11 IATTR = IVAL(JATTR,IMESS)

12 IF (IATTR .NE. JPLANS) GO TO 50

C C .. IF CONTENTS FUNCTION RECORD, CONVERT BY CAL TO IFNCVL

13 CALL PARSRF(MCONT, ID, IN, IF)

14 IF (ID .NE. LFUNC) GO TO 2

15 MCONT = IFNCVL(MCONT)

16 IF (MCONT. EQ. 0) GO TO 50

17 2 CALL CHGFLD(IATTR,IPERSN,MCONT,1,NULL)

18 GO TO 50

C C SELECT A MEANS BY WHICH TO COMMUNICATE WITH THE ADDRESSEE.

C C THE CURRENT ROUTINE CHOOSES THE FIRST COMMO NET ENCOUNTERED WHICH

C C IS AVAILABLE TO THE SENDER AND ON WHICH THE ADDRESSEE IS A RECIPIENT.

FSN MODEL - COMMO SUBROUTINE

1SN
19 5 ICMNTS = IVAL(JCMNTS,IPERSN)
20 ICOMNT = IFIRST(ICMNTS,KC)
21 10 IF (ICOMNT .EQ. NULL) GO TO 50
C
C IS THE ADDRESSEE EITHER UNSPECIFIED OR ON THIS COMNET ?
22 IF (IADRES .EQ. NULL) GO TO 20
23 IRCVRS = IVAL(JRCVRS,ICOMNT)
24 IRCVR = IFIRST(IRCVRS,KR)
25 15 IF (IRCVR .EQ. NULL) GO TO 30
26 IF (IRCVR .EQ. IADRES) GO TO 20
27 IRCVR = NEXT(IRCVRS,KR)
28 GO TO 15
C
C ADD MESSAGE TO THIS COMNET. FIRST UPDATE TIME OF THE MESSAGE.
29 20 I = IREP(JTIME,IMESS)
30 DITEM(I) = TMIN
C GET LIST OF MESSAGES ALREADY ON COMNET & ITS LOCATION
31 MSLIST = IVAL(JMESS,ICOMNT)
32 IMS = IREP(JMESS,ICOMNT)
C STACK THE NEW MESSAGE ONTO THIS LIST
33 ITEM(IMS) = ISTACK(IMESS,MSLIST)
C UNLESS ADDRESSEE WAS UNSPECIFIED, ARE DONE.
34 IF (IADRES .NE. NULL) GO TO 50
C TRY NEXT NET
35 30 ICOMNT = NEXT(ICMNTS,KC)
36 GO TO 10
37 50 RETURN
38 END

FSN MODEL - COMOBS SUBROUTINE

15N

1

SUBROUTINE COMOBS

C ... THE PURPOSES OF THIS ROUTINE ARE AS FOLLOWS:
C 1. CREATE OBSERVATIONS FOR EACH PLAYER BASED ON ENTITIES HE
C OBSERVES DURING A CYCLE (LIST PASSED IN IOBSV)
C 2. CREATE "SELF-OBSERVATIONS" BASED ON CHANGES THE PLAYER
C UNDERWENT DURING THE PREVIOUS CYCLE
C 3. SUBJECT THESE OBSERVATIONS TO HUMAN FACTORS PROCESSING
C 4. MERGE THESE OBSERVATIONS WITH "ASSESSMENTS" THAT OCCURED IN THE
C ACTIVITY MODULE.
C 5. UPDATE A PLAYER'S PERCEPTIONS BASED ON HIS OBSERVATIONS AND
C ASSESSMENTS
C 6. CONVERT OBSERVATIONS TO MESSAGES AND ADD THESE MESSAGES TO THE
C APPROPRIATE COMMUNICATION NETS.
C 7. NULL OUT THE OBSERVATION AND ASSESSMENT LISTS AFTER PROCESSING

2

COMMON /STATEV/

3

DIMENSION IITEM(41900),DITEM(41900)

4

EQUIVALENCE (DTMIN,IITEM(1)),(DITEM(1))

5

COMMON /PARS/

6

EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)

7

REAL*8 DTPNAH,FLDNAM,FORMOT

8

COMMON /PARS1/

9

COMMON /PARS2/

10

COMMON /PARS3/

11

COMMON /DATAV/

12

DIMENSION ACTTIM(25)

13

EQUIVALENCE (ACTTIM(1),ACTRAT(1))

14

COMMON /RECREF/

15

DIMENSION RECREF(140)

16

EQUIVALENCE (RECREF(1),IGOALS)

17

COMMON // LMAX,PMIN,IMSG(50),PMSG(50),NMSG,POBSV,LOWMSG

+ IPLARF,IOBSRF,JOBSRF,IDUMRF,IASMNT(19),IPCPRF

18

DIMENSION INSTAT(2,5)

19

EQUIVALENCE (INSTAT(1,1),IOBSV(41))

20

KNSDR = 4

21

NNSDR = 5

FSN MODEL - COMOBS SUBROUTINE

ISN

C

C ... START PLAYER PROCESSING LOOP

C

22 ISIDE=KSFF

23 CALL CHGVAR(4,ISIDE)

C

C ... SET UP REPORT LISTING HEADING

C

24 1601 FORMAT('REPORT AT TIME: ',F6.2,/,/)

25 TTTTT = TMIN + DTMIN

26 WRITE(7,1601) TTTTT

27 LISPLA=IVAL1(NHWREF(LSITE,1,JGAUDS))

28 10 IPLARE=IFIRST(LISPLA,LPTPLA)

29 20 IF(IPLARE.EQ.NULL) GO TO 200

30 CALL PARSRF(IPLARE,IRT,IPLAYR,IFN)

31 CALL CHGVAR(3,IPLAYR)

C

C ... REPORT PLAYER STATUS

C

32 CALL REPCRT(IPLARE,7)

C

C ... IF PLAYER CANNOT OBSERVE, IGNORE

C

33 IF(ICANOB(IPLARE).EQ.NULL) GO TO 150

C

C ... BLOCK I-2 CREATE OBSERVATIONS.....

C

34 PMEN=1000.0

35 LDMMSG=0

36 NMSG=0

37 LISPCP=IVAL1(JPRCPS,IPLARE)

38 TPCPFF=NULL

39 LOCAL(?) = NULL

C

C ... GET LIMIT ON ITEMS OF INFORMATION PLAYER CAN RETAIN AND PROCESS

FSN MODEL - COMOBS SUBROUTINE

ISN
40 CALL DPCOM1(IPLARE,LMAX)
C
C ... START OBSERVATION PROCESSING LOOP*****
C
41 LTSOBS=IOBSV(IPLAYR)
42 TOBSRF=IFIRST(LTSOBS,LPTOBS)
43 30 IF(TOBSRF.EQ.NULL) GO TO 40
44 IF(TOBSRF.EQ.IPLARE) GO TO 35
C
C ... DETERMINE IF PLAYER HAS PERCEPTION OF OBJECT BEING OBSERVED
C IF HE DOES SET IPCPRF AND JOBSRF
45 LISPCP=IVAL(JPRCPS,IPLARE)
46 IPCPRF=LSEARCH(LISPCP,NULL,NULL,JID,TOBSRF)
47 31 IF(IPCPRF.EQ.NULL) GO TO 32
48 JOBSRF=IVAL(JVIEW,IPCPRF)
C IPCPRF ... POINTS TO PERCEPTION RECORD OR IS NULL
C JOBSRF ... POINTS TO PLAYER'S VIEW OF OBJECT
C TOBSRF ... POINTS TO OBJECT DESCRIPTION
C
C ... GET DESCRIPTION OF OBJECT WITH ONLY THOSE ATTRIBUTES
C PLAYER CAN OBSERVE(IDUMRF)
C
49 32 CALL DR0BS1
50 LOCAL(3)=IDUMRF
51 IF(IDUMRF.EQ.NULL) GO TO 35
C
C ... GET OBSERVATION PRIORITY FOR OBJECT
C
52 CALL DR0BS3
C
C ... GENERATE MESSAGES IF OBSERVATION <> PERCEPTION
C ADD TO TMSG IF QUALIFIED
C
53 CALL GENOBS
C
C ... IF PLAYER HAS PERCEPTION SET TIME UPDATED

FSN MODEL - COMOBS SUBROUTINE

TSN

```
C
54      IF(IPCPRF.NE.NULL) CALL CHGFLD(ITMF,IPCPRF,TMIN,1,IDUM)
C
C ... END OBSERVATION PROCESSING LOOP*****  

C
55 35    IOBSRF=NEXT(LISOBS,LPTOBS)
56    GO TO 30
C
C ... IF PLAYER IS SENSOR MONITOR PROCESS OBSERVATIONS
C     FROM SENSORS
C
57 40    IF(IVAL(JTYPE,IPLARE).NE.KSNMON) GO TO 50
58    IPTR = IREF(JSITN,IPLARE)
C
C ... START SENSOR MONITOR PROCESS LOOP
C
59    DO 45 I=1,NSNRTP
60      LISOBS=IMNOBS(I)
61      IMSGRF=FIRST(LISOBS,LPTOBS)
62 41    IF(IMSGRF.EQ.NULL) GO TO 45
63      IOBSRF=IVAL(JSUBJ,IMSGRF)
64      IF(IOBSRF.EQ.IPLARE) GO TO 43
C
C ... IF IOBSRF IS PERSON OF OPPOSITE ALLEGIANCE
C     UPDATE IPLARE'S SITUATION
65      IDUM = MOD(IOBSRF,10000000)
66      ITP = IDUM/10000
67      IF(ITP.NE.LPERSN) GO TO 45
68      IDUM = IVAL(JALLEG,IOBSRF)
69      IF(IDUM.NE.1) ITEM(IPTR) = KSITN4
C
C ... DETERMINE IF PLAYER HAS PERCEPTION
C
70 45    LISPCP=IVAL(JPPCPS,IPLARE)
71    IPCPREF=LSEARCH(LISPCP,NILL,NILL,ITD,IOBSRF)
72    IF(IPCPREF.EQ.NULL) GO TO 42
```

FSN MODEL - COMOBS SUBROUTINE

TSN
73 JOBSRF=IVAL(JVIEW,IPCPRF)
74 42 CALL DRCBS2()
75 LOCAL(3)=IDUMRF
76 IF(IDUMRF.EQ.NULL) GO TO 43
77 CALL DRCBS3
78 CALL GENOBS
79 IF(IPCPRF.NE.NULL) CALL CHGFLD(JTIME,IPCPRF,TMIN,1,JDJ)
80 43 TMSGRF=NEXT(LTSORS,LPTOBS)
81 GO TO 41
C
C ... END SENSOR MONITOR PROCESS LOOP
C
82 45 CONTINUE
C
C ... BLOCK I-3 CREATE OBSERVATIONS BASED ON OWN ACTIVITY.....
C
83 50 IOBSRF=IPLARF
84 IPCPRF=LSEARCH(LISPCP,NULL,NULL,JID,IOBSRF)
85 JOBSRF=NULL
86 IF(IPCPRF.NE.NULL) JOBSRF=IVAL(JVIEW,IPCPRF)
87 CALL DRCBS1
88 LOCAL(3)=IDUMRF
89 CALL DRCBS3
90 CALL GENOBS
C
C ... BLOCK I-4 HUMAN FACTORS PROCESSING
C
91 55 IF(NMSG.LE.0) GO TO 60
92 CONTINUE
C
C
C ... BLOCK I-5 MERGE OBSERVATIONS AND ASSESSMENTS
C
93 60 NASMNT=0
94 IF(NASMNT.EQ.NULL) GO TO 70
95 61 IPRFM=NULL

ISN

FSN - FDEL - COMBRS SUBROUTINE

```
C
C ... START LOOP THROUGH ASSESSMENTS
C
95      IMSGRF=IFIRST{IASMNT, ICUR)
97      IF(IMSGRF.EQ.NULL) GO TO 70
98      IF((VAL{JSOURC, IMSGRF).NE.IPLARF) GO TO 68
C
C ...CHECK TO SEE IF DATA ALREADY ON IMSG
C     BASED ON OBSERVATION
C
99      IF(NMSG.LE.0) GO TO 64
100     MSUR=VAL{JSUBJ, IMSGRF)
101     MATT=VAL{JATTR, IMSGRF)
102     DO E3 T=1,NMSG
103       MSUPA=VAL{JSUBJ, IMSG{1))
104       MATT=VAL{JATTR, IMSG{1))
105       IF(MSUR.NE.MSUPA .OR. MATT.NE.MATT) GO TO 63
C
C     ALREADY IN IMSG, REPLACE WITH ASSESSMENT
C
106     IMSG(T)=IMSGRF
107     GO TO E5
108   E3    CONTINUE
C
C ... PROCESS ASSESSMENT PREVIOUSLY MADE BY IPLARF
C     CALCULATE PRIORITY FOR SUBJECT OF ASSESSMENT
C
109   E4    IDBSRF=VAL{JSURJ, IMSGRF)
110     CALL DR0RS3
C
C ... CALCULATE PRIORITY FOR ATTRIBUTE ASSESSED
C
111     CALL PARSRF(IDBSRF, IDTYP, I1, I2)
112     IFEAT=IDTYP-LVTHIC+1
113     INDEX=0
114     GO TO {643,642,641}, IFEAT
```

154 INDEX=INDEX+NFLDS(LPERSON)
 155 541 INDEX=INDEX+NFLDS(VEHIC)
 156 542 INDEX=INDEX+IVAL(JATTR,IMSGRF)
 157 543 P=OBJSV*PORTRAY(INDEX)
 158
 C *** MERGE ASSESSMENT INTO IMSG
 C
 159 C CALL ADDVAL(IMSGRF,P,IMSG,PMSC,NMSG,LOWMSG,PMIN,LMAX)
 C *** SAVE ASSESSMENT FOR PERCEPTION UPDATE
 C
 160 55 NASMNT=NASMNT+1
 161 55 TASMNT(NASMNT)=IMSGRF
 C *** PRINT FROM LIST OF ASSESSMENTS
 C
 162 67 IF(IPREV,FO,NULL) GO TO 67
 C *** NOT FIRST ITEM ON LIST
 C
 163 LISTNEXT,IPREV)=LISTNEXT(ICUR)
 164 ICUR=IPREV
 165 GO TO 68
 C *** FIRST ITEM ON LIST
 C
 166 67 L5MNT=NULL
 167 CALL CHGVAR(11,NULL)
 168 ICUR=FO NULL GO TO 70
 169 IF(LISTNEXT(ICUR),FO,NULL) GO TO 70
 170 L5MNT=NEWDEFULLIST(LISTNEXT(ICUR),0)
 171 CALL CHGVAR(11,L5MNT)
 172 GO TO 61
 C *** END LOOP THROUGH ASSESSMENTS
 C

FSN MODEL - COMOBS SUBROUTINE

15N
133 68 IPREV=ICUR
134 INSGRF=NEXT(LASMNT,ICUR)
135 GO TO 62
C
C ... ADD OBSERVATIONS TO PERCEPTIONS
C
136 70 IF(NMSG.LE.0) GO TO 74
137 DO 72 I=1,NMSG
138 CALL NEWCP(INSGR(I),IPLARE)
139 72 CONTINUE
C
C ... ADD ASSESSMENTS TO PERCEPTIONS
C
140 74 IF(NASMT.EQ.0) GO TO 90
141 DO 86 I=1,NASMT
142 CALL NEWCP(INASMT(I),IPLARF)
143 86 CONTINUE
C
C ... BLOCK 1-6 ADD OBSERVATIONS TO NET
C
C ... START LOOP THROUGH PLAYER'S NETS
C
144 90 IF(NMSG.LE.0) GO TO 150
145 NETLIS=IVAL(JCMNTS,IPLARE)
146 NETREF=IFIRST(NETLIS,LPTNET)
147 100 IF(NETREF.EQ.NULL) GO TO 150
148 LLRF=IVAL(JMESS,NETREF)
149 DO 110 I=1,NMSG
150 LLRF=ISTACK(INSC(I)),LLRF)
151 CALL CHGFLD(JMESS,NETREF,LLRF,1,1DUM)
152 110 CONTINUE
C
C ... END NET LOOP
C
153 NETREF=NEXT(NETLIS,LPTNET)
154 GO TO 100

FSN MODEL - COMBOS SUBROUTINE

151 ISN
C
C ... NULL OUT OBSERVATIONS
C
155 150 ICBSV(IPLAYR)=NULL
156 CALL CHGVR1(9,NULL,IPLAYR)
C
C ... END PLAYER PROCESSING LOOP
C
157 190 IPLAYP=NEXT(LISPLA,LPTPLA)
158 GO TO 20
159 200 IF (ISIDE.EQ.KAFE) GO TO 210
160 ISIDE=KAFE
161 LISPLA=IVAL1(NEWREF(LSITE,1,JADVRS))
162 GO TO 10
C
C ... PROCESSING COMPLETE
C
163 210 LASMNT=NULL
164 LOCAL(2) = NULL
165 LOCAL(3) = NULL
166 CALL CHGVAR(11,NULL)
C
C ... CHECK FOR ANY INSIDERS DISCOVERED THIS CYCLE
C
167 DO 220 I=1,NINSDR
168 IF (INSTAT(2,I) .LT. 0) CALL SURFAC(INSTAT(1,I))
169 INSTAT(2,I) = 0
170 220 CONTINUE
C
C ... NULL OUT IMNCBS
171 DO 250 I=1,4
172 IMNCBS(I) = NULL
173 CALL CHGVR1(10,FULL,I)
174 250 CONTINUE
175 RETURN
176 END

FSN MODEL - COMPER SUBROUTINE

ISN

1

SUBROUTINE COMPER

C ... THE PURPOSE OF THIS ROUTINE IS TO UPDATE PLAYER
C PERCEPTIONS BASED ON INFORMATION RECEIVED OVER
C COMMUNICATION NETS. UPDATING OF PERCEPTIONS
C BASED ON OWN OBSERVATION HAS OCCURRED PREVIOUSLY
C IN COMBOS.
C ... PLAYER'S PLANS ARE UPDATED BASED ON ORDERS
C RECEIVED FROM THEIR LEADER
C ... INTERCEPTED COMMUNICATIONS ARE SAVED FOR
C RETRANSMISSION THE FOLLOWING CYCLE.
C ... AFTER ALL PLAYERS HAVE BEEN PROCESSED THE
C STALE MESSAGES ARE REMOVED FROM THE NET QUEUES.

2

COMMON /STATEV/

3

DIMENSION ITEM(41900),DITEM(41900)

4

EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

5

COMMON /PARS/

6

EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)

7

REAL*8 DTPNAM,FDNAM,FORMDT

8

COMMON /PARS1/

9

COMMON /PARS2/

10

COMMON /PARS3/

11

COMMON /DATAV/

12

DIMENSION ACTTIM(25)

13

EQUIVALENCE (ACTTIM(1),ACTRAT(1))

14

COMMON /RECRFF/

15

DIMENSION RECRFO(140)

16

EQUIVALENCE (RECRFO(1),IGOALS)

17

COMMON IMSG(50),IMSN(50),IINCP(50),PFMSG(50),PRMS(50),PRNCP(50)

18

LOGICAL INTCPT,PENDAL

C

C ... START LOOP THROUGH PLAYERS

C

19

ISIDE=K\$EE

20

CALL CHGVAR(4,ISIDE)

21

NNET=LSTREC(LCMNT)

FSN MODEL - COMPER SUBROUTINE

ISN

22 LISPLA=IVAL(JGARES,ISITE)

23 10 IPLARF=IFTEST(LISPLA,LPTPLA)

24 20 IF(IPLARF.EQ.NULL) GO TO 200

25 CALL PARSRF(IPLARF,IDL,IPLAYR,IDUM)

26 CALL CHGVAR(3,IPLAYR)

 C

 C INSURE PLAYER IS ACTIVE

 C

27 C

28 IF(ICANOR(IPLARF).EQ.NULL) GO TO 190

 C

 C GET LIMITATIONS ON MESSAGES THAT PLAYER

 C CAN PROCESS

 C LMSN IS LIMIT ON MISSION MESSAGES

 C LMSG IS LIMIT ON OTHER MESSAGES

 C LINCP IS LIMIT ON MESSAGES INTERCEPTED

 C LORDER IS LIMIT ON ORDERS RECEIVED

 C

28 CALL DRCON2(IPLARF,LMSN,LMSG,LINCP,LORDER)

29 PMSN=1000.0

30 PINCP=1000.0

31 PMSC=1000.0

32 MMSN=0

33 MINCP=0

34 MMSG=0

35 LISPCP=IVAL(JRCPS,IPLARF)

 C

 C START LOOP THROUGH COMNETS

 C

36 NNET=LSTREC(LCOMNT)

37 DO 100 INET=1,NNET

38 NETREF=NEWREF(LCOMNT,INET,0)

 C SEE IF PLAYER ON NET

 C

39 ION=LSERCH(IVAL(JRCVRS,NETREF),IPLARF,NULL,NULL,NULL)

40 IF(ION.EQ.NULL) GO TO 100

 C

FSN MODEL - COMPER SUBROUTINE

ISN C ... GET DELAY TIME FOR NET, SEE IF IPLAYR
C IS INTERCEPTING NET COMMUNICATIONS
C
41 IT = IVAL(JTYPE,NETREF)
42 DELAY=COMDLA(IT)
43 CALL DRCOM3(IPLARE,NETREF,INTCPT)
C
C START MSG PROCESS LOOP
C
44 LMSGEG=IVAL(JMESS,NETREF)
45 IMSGFF=IFIRST(LMSGEG,MPTR)
46 20 IF(IMSGRF.EQ.NULL) GO TO 100
C IF IPLARE IS SENDER, IGNORE
C
47 IF(IVALE(JSOURC,IMSGRF).EQ. IPLARE) GO TO 90
C
C ... IF DELAY TIME NOT ELAPSED, IGNORE MESSAGE
C
48 IF(VAL(JTIME,IMSGRF) + DELAY .GE. TMIN) GO TO 90
C
C PROCESS MESSAGE
C
49 IATTR=IVAL(JATTR,IMSGRF)
50 ICBSRF=IVAL(JSURJ,IMSGRF)
51 CALL PARSRF(IDBSRF, IDTYP,IRECNO,IFLD)
C
C CHECK FOR MISSION MESSAGE
C
52 IF(IDTYP .NE. LPERSN) GO TO 40
53 IF(IATTR.NE.JPLANS .AND. IATTR.NE.JSOPST) GO TO 40
C
C MISSION MESSAGE PROCESSING
C
C IF CONTENTS IS FUNCTION RECORD, CONVERT BY CALL TO IFNCVL
54 MCNT=IVAL(JCONT,IMSGFF)
55 CALL PARSRF(MCNT, ID, IN, IF)

FSN MODEL - COMPER SUBROUTINE

1SN
56 IF(LID .NE. LFUNC) GO TO 34
57 MCNT=IFNCVL(MCNT)
58 IF(MCNT .EQ. 0) GO TO 90
59 IF(IORSRF .NE. IPLARF) GO TO 90
60 34 IT = NFLDS(LVEHIC) + IATTR
61 P=PCBATT(IT)
62 IF(IOBSRF .NE. IPLARF) GO TO 35
C
C MISSION IS FOR PLAYER, CHECK SENDER
C
C
63 ISEADR=IVAL(JSOURCE,IMSGRF)
64 IF(ISENDP.NE.IVAL(JLEADR,IPLARF)) GO TO 35
C
C *** DECREMENT ORDER COUNTER
C
65 LORDER=LORDER-1
66 IF(LORDER.LT.0) GO TO 90
C
C UPDATE PLAYER'S PLANS/SOPS
C
67 CALL CHGFLD(IATTR,IPLARF,MCNT,1,NULL)
68 P=2.0*p
69 GO TO 90
C
C ADD INFO TO MISSION LIST
C
70 35 CALL ADDVAL(IMSGRF,P,IMSN,PRMSN,MMSN,LOWMSN,PMSN,LMSN)
71 GO TO 80
C
C *** PROCESS NON-MISSION MESSAGES
C IF PLAYER IS SUBJECT OF MESSAGE, IGNORE
C
72 -0 IF(IVAL(JSURJ,IMSGRF).EQ.IPLARF) GO TO 90
C
C IF PLAYER HAS NO PERCEPTION GO TO 50

FSN MODEL - COMPER SUBROUTINE

15N
73 IPCPRF=LSEARCH(LTSPCP,NULL,NULL,JID,IORSRF)
74 IF(IPCPRF.EQ.NULL) GO TO 50
C
C ... IF MSG TIME >= PCP TIME, SKIP
C
75 IF(VAL(JTIME,IMSGRF).LE. VAL(JTIME,IPCPRF)) GO TO 90
C
C ... IF PERCEPTION EQUAL MSG, SKIP
C
76 IF(IEQUAL(IPCPRF,IMSGRF)) GO TO 90
C
C ... COMPUTE PROCESSING PRIORITY
C
77 P0 CALL DRCBS4(P,IPLARE,IMSGRF)
C
C ... ADD TO LIST OF MESSAGES
C
78 CALL ADDVAL(IMSGRF,P,IMSG,PRMSG,MMSG,L0MSG,PMSG,LMSG)
C
C ... IF INTERCEPT, ADD TO IINCP
79 P0 IFI.NOT.INTCPTI GO TO 90
80 CALL ADDVAL(IMSGRF,P,IINCP,PRINCP,MINCP,L0INCP,PINCP,LINCP)
81 S0 IMSGRF=NEXT(LISMSG,MPTR)
82 GO TO 30
83 100 CONTINUE
C
C UPDATE PERCEPTIONS
C
84 110 IF(MMSG.LE.0) GO TO 130
C
C ... ADD MESSAGES TO PERCEPTIONS
C
85 DO 120 I=1,MMSG
86 CALL NEWPCP(IMSG(I),IPLARE)
87 120 CONTINUE
88 130 IF(MMSG.LE.0) GO TO 150

ISN

FSN MODEL - COMPER SUBROUTINE

```
C
C ... ADD MISSIONS TO PERCEPTIONS
C
89      DO 140 T=1,MMSN
90          CALL NEWPCP(IMSNT,IPARF)
91 140      CONTINUE
C
C ... SAVE INTERCEPTED MESSAGES
C
92 150      IF(MINCP.LE.0) GO TO 190
93          LISREF=NULL
94          DO 160 I=1,MINCP
95              LISREF=ISTACK(I,INCPI,LISREF)
96 160      CONTINUE
C
C ... PUT LISREF ON LIST
C
97          LINCP(IPARF)=LISREF
C
C ... END PLAYEP LOOP
C
98 190      IPARF=NEXT(LISPLA,LPTPLA)
99      GO TO 20
C
C .. END SIDE LOOP
C
100 200     IF(ISIDE.EQ.KAFE) GO TO 210
101     ISIDE=KAFE
102     LTSPLA=IVAL(JADVES,ISITE)
103     GO TO 10
C
C ... DELETE EXPIRED MESSAGES
C
104 210     DO 250 INFT=1,NNET
105         NETREF=NEWREF(LCOMMNT,INFT,0)
106         LTSMSG=JVAL(JMESS,NETREF)
```

FSN MODEL - COMPER SUBROUTINE

TSN

107 IT = IVAL(JTYPE,NETREF)

108 DELAY=CO TT)

109 215 IPREV=NULL

110 IMSGRF=IFIRST(LISMSG,ICUR)

111 220 IF(IMSGRF.EQ.NULL) GO TO 250

C

C ... START MESSAGE LOOP, IF MESSAGE

C NOT AVAILABLE THIS CYCLE, SKIP IT

112 IF(IVAL(JTIME,IMSGRF)+DELAY .GE. TMIN) GO TO 240

C

C ... REMOVE MESSAGE FROM LIST

C

113 IF(IPREV.EQ.NULL) GO TO 230

C ... NOT FIRST MESSAGE

114 LISTND(JNXT,IPREV)=LISTND(JNXT,ICUR)

115 ICUR=IPREV

116 GO TO 240

C ... FIRST ITEM ON LIST, NEW HEAD

117 230 IF(LISTND(JNXT,ICUR) .EQ. NULL) GO TO 235

118 LISMGS=NEWREFILLIST,LISTND(JNXT,ICUR),0)

119 CALL CHCFLD(JMESS,NETREF,LISMGS,1)

120 GO TO 215

C

C ... ONLY ITEM ON LIST

C

121 235 CALL CHCFLD(JMESS,NETREF,NULL,1,TDUM)

122 GO TO 250

C

C ... END MESSAGE LOOP

C

123 240 IPREV=ICUR

124 IMSGRF=NEXT(LISMSG,ICUR)

125 GO TO 220

126 250 CONTINUE

127 RETURN

128 END

FSN MODEL - CONDAC SUBROUTINE

ISN

1

SUBROUTINE CONDAC(IPLARF)

C
C ROUTINE INTERPRETS CONDITIONAL ACTIONS ASSOCIATED WITH PERSON
C IPLARF. A CONDITIONAL ACTION HAS THE FORM: WHEN{BOOLEAN EXPR,ACT}.
C BOTH THE WHEN(,) AND THE BOOLEAN EXPRESSION ARE REPRESENTED BY
C RELATION RECORDS. THE "ACT" PART IS REPRESENTED BY EITHER AN
C ACTION OR A FUNCTION RECORD.
C
C THIS ROUTINE SEQUENTIALLY TESTS ALL CONDITIONAL ACTIONS
C ASSOCIATED WITH THE DESIGNATED PERSON. IF IT ENCOUNTERS A CONDITION
C WHICH IS SATISFIED, IT ACTIVATES THE ASSOCIATED ACT. "ACTIVATE"
C MEANS EITHER (1) STACKING THE ACT ON THE PLAYER'S ACTIVITY LIST, IF
C THE ACT IS AN ACTION RECORD, OR (2) EVALUATING THE ACT, IF IT IS
C A FUNCTION RECORD. EVALUATION OF THE FUNCTION CAN RESULT IN AN
C ACTION RECORD, IN WHICH CASE ALTERNATIVE (1) IS FOLLOWED FOR THE
C RESULT.
C
C THIS ROUTINE RETURNS WHEN ALL OF THE PLAYER'S CONDITIONAL ACTIONS
C HAVE BEEN PROCESSED. THERE MAY ALSO BE CONDITIONAL ACTIONS PERTINENT
C TO A PLAYER STORED WITH THE RECORD DESCRIBING THE FORCE HE BELONGS
C TO.
C
C INPUT PARAMETERS:
C IPLARF RECORD REFERENCE TO THE PERSON BEING PROCESSED.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900), DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
7 REAL*8 DTPNAM, FLDNAM, FORMOT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 COMMON /NEW/
C
11 IFLAG = 0
12 IACTIV = IREF(JACTIV,IPLARF)

FSN MODEL - CONDAC SUBROUTINE

ISN

13 ICONDS = IVAL(JSOPS,IPLARP)

14 8 ICOND = IFIRST(ICONDS,KC)

15 10 IF (ICOND .EQ. NULL) GO TO 50

 C TEST WHETHER CONDITIONAL ACTION IS SATISPIED.

16 IACT = NRTEST(ICOND)

 C IF NOT, GO TRY NEXT CONDITIONAL ACTION

17 15 IF (IACT .EQ. 0) GO TO 40

 C IF SO, IS RESULT AN ACTION RECORD ?

18 CALL PARSRF(IACT, IDTYP,IRECNO, IDUM)

19 IF (IDTYP .NE. LACTN) GO TO 20

20 ITEM(IACTIV) = ISTACK(IACT, ITEM(IACTIV))

21 GO TO 40

 C IF RESULT IS A RELATION RECORD, TEST ITS TRUTH

22 20 IF (IDTYP .NE. LRELN) GO TO 30

23 IACT = NRTEST(IACT)

24 GO TO 15

 C IF RESULT IS A FUNCTION RECORD, EVALUATE IT.

25 30 IF (IDTYP .NE. LFUNC) CALL ERR(39,35, IDTYP, 0, 0)

26 IACT = IFNCVL(IACT)

27 GO TO 15

 C GET NEXT CONDITIONAL ACTION

28 40 ICOND = NEXT(ICONDS,KC)

29 GO TO 10

 C

 C HAVE ANY CONDITIONAL ACTIONS ASSOCIATED WITH THE FORCE THAT THE

 C BELONGS TO BEEN PROCESSED ?

30 50 IF (IFLAG .NE. 0) RETURN

31 IFLAG = 1

32 IFORCE = IVAL(JFORCE,IPLARP)

33 IF (IFORCE .EQ. NULL) RETURN

34 CALL PARSRF(IFORCE, IDTYP, IDUM, IDUM)

35 IF (IDTYP .NE. LFORCE) RETURN

36 ICONDS = IVAL(JFSOPS,IFORCE)

37 GO TO 8

38 END

FSN MODEL - CONTNT SUBROUTINE

ISN

1

SUBROUTINE CONTNT(IPLARF,IPRF)

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:
C 1. UPDATE THE CONTENTS OF ALL REGIONS AFTER
C A MOVEMENT HAS OCCURED

C

C -- INPUT PARAMETERS

C

C IPLARF -- PLAYER REFERENCE
C IPRF -- ORIGINAL PLACE REFER NCE BEFORE MOVE

C

C

C -- OUTPUT

C

C THIS SUBROUTINE ADJUSTS THE VALUE OF THE
C CONTENTS FIELD WITH RESPECT TO MOVEMENT

C

C

C -- PROCEDURE

C

C 1. DETERMINE IF PLAYER HAS CHANGED REGIONS
C 2. IF SO, REMOVE FROM OLD REGION CONTENTS AND
C ADD TO NEW REGION CONTENTS

C

C

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FORMOT
5 COMMON /PARS1/

C

C -- TEST IF NEW PLACE DIFFERENT FROM OLD PLACE

C

6 ILOC=IVAL(JLOCN,IPLARF)
7 IP=IPLACE(ILOC)
8 IF(IP .EQ. IPRF) GO TO 50

FSN MODEL - CONTNT SUBROUTINE

ISN

```
C
C -- NEW REGION, ADJUST CONTENTS OF OLD REGION
C
9      ICOLT=IVAL(JCONTS,IPRF)
10     ICORF=IFIRST(ICOLT,KA)
11     5   IF(ICORF .EQ. NULL) GO TO 20
C
12    IF(ICORF .NE. IPLARF) GO TO 10
C
C -- REMOVE FROM CONTENTS
C
13    CALL DELIST(KA,JCONTS,IPRF)
14    GO TO 20
C
C -- NEXT CONTENT
C
15    10 ICORF=NEXT(ICOLT,KA)
16    GO TO 5
C
C -- ADD IPLARF TO NEW REGION CONTENTS
C
17    20 ILT=IVAL(JCONTS,IP)
18    ILT=ISTACK(IPLARF,ILT)
19    CALL CHGFLD(JCONTS,IP,ILT,1,1DUM)
C
20    50 CONTINUE
21    RETURN
22    END
```

FSN MODEL - COVR FUNCTION

ISN
1 FUNCTION COVR(IPLARF,DIRC)
C
C -- THE PURPOSE OF THIS FUNCTION IS TO FIND
C THE AVAILABLE COVER FOR A PLAYER
C
C -- INPUT PARAMETERS
C
C IPLARF -- PLAYER REFERENCE
C
C DIRC -- DIRECTION DESIRED
C
C -- OUTPUT
C
C THE COVER AVAILABLE FOR PLAYER IPLARF
C
C
2 COMMON /PARS/
3 EQUIVALENCE (PNULL,NULL), (IPFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FORMOT
5 COMMON /PARS1/
6 COMMON /PARS2/
7 COMMON /PARS3/
C
8 DATA PI/ 3.14159/
C
C -- DETERMINE IF PLAYER IS AT A NODE
C
9 ILOC=IVAL(JLOCN,IPLARF)
10 IF (VAL(JFRAC,ILOC) .GT. 0.0) GO TO 10
C
C -- NODE MUST BE A YARD NODE FOR QUADRANT COVER
C
11 INODE=IVAL(JSOURC,ILOC)

FSN MODEL - COVR FUNCTION

```
ISN
12      CALL PARSIF(IEODE, IDYP, IRNO, IDUM)
13      IF (IDYP .NE. LYARD) GO TO 10
C
C      -- DETERMINE IF DIRECTION GIVEN
C
14      IF (DIRC .NE. FNULL) GO TO 8
C
C      -- PLAYER AT YARD NODE
C      DETERMINE ACTIVITY AND DIRECTION
C
15      IACTLT=IVAL(JACTIV, IPLARF)
16      IACTRF=IFIRST(IACTLT, KA)
17      IF (IACTRF .EQ. NULL) GO TO 10
18      IF (IVAL(JTYPE, IACTRF) .EQ. KMOVNG) GO TO 10
19      IF (IVAL(JTYPE, IACTRF) .EQ. KCAPNG) GO TO 10
20      IF (IVAL(JTYPE, IACTRF) .EQ. KFIRNG) GO TO 5
C
C      -- OBSERVING ACTIVITY
C
21      DIRC=VAL(JPAR1, IACTRF)
22      IF (DIRC .EQ. FNULL) GO TO 10
23      GO TO 8
C
C
C      -- FIRING ACTIVITY
C
24      5 ITRG=IVAL(JPAR1, IACTRF)
25      CALL DIREC(IPLARF, ITRG, DIRC, PHE)
C
C      -- FIND QUADRANT
C
26      8 DIRC=AMOD(DIRC, 2*PI)
27      IQUAD=JCOV1
28      IF (DIRC .GT. 0.5*PI) IQUAD=JCOV2
29      IF (DIRC .GT. PI) IQUAD=JCOV3
30      IF (DIRC .GT. (1.5*PI)) IQUAD=JCOV4
```

FSN MODEL - COVR FUNCTION

ISN C
31 COVR=VAL(IQUAD,INODE)
32 GO TO 20
C
C
C -- REGION COVER
C PLAYER MUST BE IN EITHER YARD, ROOM, OR HALL
C
33 10 IPL=IVAL(JPLACE,ILOC)
34 CALL PARSRF(IPL, IDY, IRNO, IDUM)
35 IF (IDY .EQ. LYARD) GO TO 15
36 IF (IDY .EQ. LHALL) GO TO 15
37 IF (IDY .EQ. LROOM) GO TO 15
C
38 COVR=0.0
39 GO TO 20
C
C
40 15 COVR=VAL(JCOV, IPL)
C
41 20 CONTINUE
C
42 RETURN
43 END

FSN MODEL - CREAT SUBROUTINE

ISN

```
1      SUBROUTINE CREAT(I TYPE, I PAR1, I PAR2, I PAR3, I MODS, I DO, I NEWRF)  
C  
C -- THE PURPOSE OF THIS SUBROUTINE IS TO:  
C     1. CREATE AN ACTION RECORD WITH THE APPROPRIATE  
C        FIELD VALUES  
C     2. AND, ADD IT TO PLAYER REFERENCE I DO'S ACTIVITY  
C        LIST  
C  
C  
C -- INPUT PARAMETERS  
C  
C     I TYPE - TYPE OF ACTION RECORD TO CREATE  
C     I PAR1 - PARAMETER 1  
C     I PAR2 - PARAMETER 2  
C     I PAR3 - PARAMETER 3  
C     I MODS -- MODIFIER, TERMINATION CONDITION  
C     I DO - PLAYER REFERENCE THIS ACTIVITY IS ASSIGNED  
C  
C  
C -- OUTPUT PARAMETERS  
C  
C  
C     I NEWRF - ACTIVITY RECORD REFERENCE THAT WAS  
C                CREATED  
C  
C  
C -- ASSUMPTIONS  
C  
C     IT IS ASSUMED THAT ALL PARAMETERS OF THE ACTION  
C     RECORDS CAN BE REPRESENTED AS INTEGERS OR  
C     RECORD REFERENCES.  
C  
C  
2      COMMON /PARS/  
3      EQUIVALENCE (FNULL, NULL), (IFATL, FAIL)
```

FSN MODEL - CREAT SUBROUTINE

1 ISN
2
3
4 REAL*8 DTPNAM,FLDNAM,FORMOT
5 COMMON /PARS1/
6 COMMON /PARS3/
7
8 C
9 C -- FIND AVAILABLE RECORD
10 C
11 C
12 C
13 C
14 C
15 C
16 C
17 C

```
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS3/
7
8      IACTNO=NEWREC(LACTN)
9      IF(IACTNO .EQ.NULL) GO TO 10
10     C
11     C -- GET ACTION RECORD & FILL WITH VALUES
12     C
13     C
14     C -- FIND ACTIVITY LIST FOR PLAYER IDO
15     C
16     C
17     C
```

10 IACTRF=IVAL(JACPIV,IDO)
11 C
12 C -- ADD NEW RECORD TO FRONT OF LIST
13 C
14 C
15 C
16 C
17 C

```
10      IACTRF=ISTACK(INEWRF,IACTRF)
11      C
12      C -- FIND OFFSET IN ITEM TO SAVE ACTIVITY REFERENCE
13      C
14      C
15      C
16      C
17      C
```

10 IACTRF=ISTACK(INEWRF,IACTRF)
11 C
12 C -- FIND OFFSET IN ITEM TO SAVE ACTIVITY REFERENCE
13 C
14 C
15 C
16 C
17 C

```
10      CALL CHGFLD(JACTIV,IDO,IACTRF,1,1DUM)
11      GO TO 20
12      C
13      C
14      C
15      C
16      C
17      C
```

10 CALL CHGFLD(JACTIV,IDO,IACTRF,1,1DUM)
11 GO TO 20
12 C
13 C
14 C
15 C
16 C
17 C

```
10      ERROR -----
11      C
12      C
13      C
14      C
15      C
16      C
17      C
```

10 ACTION RECORD NOT AVAILABLE
11 C
12 C
13 C
14 C
15 C
16 C
17 C

```
10      10 CONTINUE
11      C
12      C
13      C
14      C
15      C
16      C
17      C
```

10 10 CONTINUE
11 C
12 C
13 C
14 C
15 C
16 C
17 C

```
10      RETURN
11      C
12      C
13      C
14      C
15      C
16      C
17      C
```

PSN MODEL - CREAT SUBROUTINE

ISBN
18

END

FSN MODEL - CREREL SUBROUTINE

1 SUBROUTINE CREREL(ITYPE,IPAR1,IPAR2,IRF)

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. CREATE A RELATION RECORD WITH THE INPUT PARAMETERS

C

C -- INPUT PARAMETERS

C

C ITYPE - TYPE OF RELATION

C IPAR1 -- PARAMETER 1

C IPAR2 -- PARAMETER 2

C

C -- OUTPUT

C

C IRF -- REFERENCE TO RECORD

C

2 COMMON /PARS/

3 EQUIVALENCE (NULL,NULL), (FAIL,FAIL)

4 REAL*8 DTPNAM,FLDNAM,FORMOT

5 COMMON /PARS1/

6 COMMON /PARS3/

C

C

7 INO=NEWREC(LRELN)

8 IF (INO .EQ. NULL) GO TO 10

9 IRF=NEWREF(LRELN,INO,0)

C

C -- FIND OFFSET IN ITEM

C

10 CALL CHGFLD(JTYPE,IRF,ITYPE,1,1DUM)

11 CALL CHGFLD(JPAR1,IRF,IPAR1,1,1DUM)

12 CALL CHGFLD(JPAR2,IRF,IPAR2,1,1DUM)

13 GO TO 20

C

14 10 IRF=NULL

FSN MODEL - CREREL SUBROUTINE

15 C
15 20 CONTINUE
16 RETURN
17 END

FSN MODEL - DEAD SUBROUTINE

1SN

```
1      SUBROUTINE DEAD(IPLARE)
C
C   - THE PURPOSE OF THIS SUBROUTINE IS TO:
C     1. RELEASE ANY LISTS ASSOCIATED WITH PLAYER
C     2. IF PLAYER IS A LEADER, FIND NEW LEADER
C
C   -- INPUT PARAMETERS
C
C     IPLARE -- PLAYER REFERENCE
C
C
2      COMMON /STATEM/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (CTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FDNAM,FDRMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /NEW/
C
C   -- DELETE PERCEPTIONS
C
12     IPTR=IREFI(IPLARE)
13     ITEM(IPTR+IPFCPS)=NULL
C
C   CREATE MESSAGE OF PHYSICAL STATUS
C
14     ENSG=ENRGCE(EMESS,IPLARE,IPLARE,IPSTAT,TIN,KRAD)
C
C   -- REMOVE FROM COMMUNICATION NETS
```

FSN MODEL - DEAD SUBROUTINE

ISN

```
      C
15      ICONLT=IVAL(JCHNNTS,IPLARE)
16      ICON=IFIRST(ICONLT,KAI)
17      5      IF(ICON .EQ. NULL) GO TO 10
      C
      C ADD MESSAGE TO COMO NET
      C
18      ILT=IVAL(JMESS,ICON)
19      ILT=ISTACK(TMSG,ILT)
20      IPP=IREF1(ICON)
21      ITEM(IPP+JMESS)=ILT
      C
      C ... REMOVE IPLARE FROM LIST OF RECEIVERS
      C
22      ILLST=IVAL(JRCVRS,ICON)
23      IDUM=IFIRST(ILLST,KL)
24      6      IF(IDUM .EQ. NULL) GO TO 8
25      IF(IDUM .EQ. IPLARE) GO TO 7
26          IDUM=NEXT(ILLST,KL)
27          GO TO 6
28          7 CALL DELIST(KL,JRCVRS,ICON)
29          9      ICON=NEXT(ICONLT,KAI)
30          GO TO 5
      C
      C - IF PLAYER IS NOT A LEADER RETURN
      C
31      10 IF(IVAL(LEADER,IPLARE) .NE. NULL) GO TO 40
      C
      C -- PLAYER IS A LEADER FIND NEW ONE
      C
32      IFORCE=IVAL(JFORCE,IPLARE)
33      ILNEW=NEWLDF(IPLARE,IFORCE)
34      IF(ILNEW .EQ. NULL) GO TO 40
      C
      C -- UPDATE ALL SUBORDINATES ON FORCE TO NEW LEADER
```

FSN MODEL - DEAD SUBROUTINE

ISN
C ADJUST FORCE RECORD
C
25 IPTRF=IREF1(IFORCE)
36 ITEM(IPTRF+JFLDR)=ILNEW
C
C ... CREATE UPDATED LIST OF SUBORDINATES IN ISUBLT
C
37 ISUBLT = NULL
38 ILIS = IVAL(JSURS,IPLARF)
39 IPSN = IFIRST(ILIS,LPTR)
40 20 IF(IPSN .EQ. NULL) GO TO 30
C
C ... IGNORE ILNEW AND DEAD OR CAPTURED PLAYERS
C
41 IF(IPSN .EQ. ILNEW) GO TO 25
42 IPTR = IREF1(IPSN)
43 IF(ITEM(IPTR+JPSTAT) .LE. KCAPTR) GO TO 25
C
C ... GIVE IPSN NEW LEADER REF AND ADD TO ISUBLT
C
44 ITEM(IPTR+JLEADR) = ILNEW
45 ISUBLT = ISTACK(IPSN,ISUBLT)
46 25 IPSN = NEXT(ILIS,LPTR)
47 GO TO 20
C
48 30 IPTR=IREF1(ILNEW)
49 ITEM(IPTR+JLEADR)=NULL
50 ITEM(IPTR+JSURS)=ISUBLT
51 IPTR=IREF1(IPLARF)
52 ITEM(IPTR+JSURS)=NULL
C
53 40 CONTINUE
54 RETURN
55 END

PSN MODEL - DECIDE SUBROUTINE

ISN

1

SUBROUTINE DECIDE

C

C

THE PURPOSES OF THIS SUBROUTINE ARE TO:

1. RESOLVE ANY CAPTURE SITUATIONS WHICH HAVE RESULTED FROM ACTIVITY UPDATE OR SURRENDER DECISION.
2. SET SUPPRESSION LEVELS FOR DECISION MAKING .
3. FOLLOW PLANS GIVEN FROM LEADERS.
4. DECIDE TO MOVE , FIRE, OBSERVE, OR SURRENDER BASED ON DESIRABILITY OF PERFORMING THESE ACTIVITES
5. AND, ADD TO A PLAYERS ACTIVITY LIST THE ACTIVITY SELECTED TO BE PERFORMED.

C

C

ASSUMPTIONS

C

C

IT IS ASSUMED THAT ACTIVITY LISTS FOR ALL PLAYERS ARE NULL EXCEPT THOSE PLAYERS INVOLVED IN CAPTURE ACTIVITES. IF AN ACTIVITY IS DESIRED TO BE PERFORMED IN SUBSEQUENT TIME INTERVALS, THIS ACTION RECORD MUST BE THE FIRST ITEM ON THE PLANS LIST.

C

C

SUBROUTINE CALLED

C

C

NEWRC. -- USED TO CREATE RECORDS
DETFIR - USED TO DETECT IF A PLAYER PERCEIVES HIMSELF TO BE UNDER FIRE
TRGLST -- USED TO CREATE A TARGET LIST
IPATH -- USED FOR PATH GENERATION
MDESIR - DESIRABILITY FUNCTION FOR MOVING
FDESIR - DESIRABILITY FUNCTION FOR FIRING
ODESIR - DESIRABILITY FUNCTION FOR OBSERVING

FSN MODEL - DECIDE SUBROUTINE

ISN

```
C SDESIR - DESIRABILITY FUNCTION FOR SURRENDERING
C LOS - LINE OF SIGHT FUNCTION
C NRTEST - USED TO EVALUATE IF RELATION IS SATISIFIED
C NACTSU - DETERMINES THE SUB-ACTIVITY FOR PLAYER
C NWPNS -- WEAPON SELECTION FOR FIRING
C UPACTS -- UPDATE ACTIVITY LSITS OF PLAYERS
C
C
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
11 COMMON /DATAV/
12 DIMENSION ACTTIM(25)
13 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14 COMMON /RECREF/
15 DIMENSION RECRFQ(140)
16 EQUIVALENCE (RECRFQ(1),IGOALS)
17 COMMON /PARS4/
18 DIMENSION INSTAT(2,5)
19 EQUIVALENCE (IOBSV(41),INSTAT(1,1))

C
20 NINSDR=5
21 KINSDR=4

C
C -- UPDATE ACTIVITY LISTS OF PLAYERS
C
22 CALL UPACTS
C
```

PSN MODEL - DECIDE SUBROUTINE

1SN

```
C
C   -- ISITE IS A GLOBAL PARAMETER WHICH IS A POINTER TO THE
C       SITE RECORD , USED TO DEFINE THE LIST OF GUARDS
C       AND ADVERSITIES.
C
C   -- LOCAL VARIABLE JL IS USED TO SEARCH GUARD AND
C       ADVERSARIES LIST.
C
C   -- PROCESS GUARDS FIRST THEN ADVERSARIES.
C
23      JL=JGARDS
24      ISIDE=1
C
C   -- FIND REFERENCE TO LIST OF GUARDS, OR ADVERSARIES.
C
25      5 IPLRST=IVAL(JL,ISITE)
26      CALL CHGVAR(4,ISIDE)
C
C   -- FIND PLAYER REFERENCE TO FIRST PLAYER ON LIST
C   ** IPLARF=PLAYER REFERENCE
C
27      IPLARP=IFIRST(IPLRST,KA)
28      8 IF(IPLARP .EQ. NULL) GO TO 510
C
C   -- TEST PHYSICAL STATUS OF PLAYER
C       IF DEAD OR CAPTURED, REMOVE FROM LIST
C
29      IF(IVAL(JPSTAT,IPLARF) .EQ. KDEAD) GO TO 35
30      IF(IVAL(JPSTAT,IPLARF) .EQ. KCAPTR) GO TO 35
C
C   -- PROCESS ANY CONDITIONAL ACTIONS ASSOCIATED WITH PLAYER
C
31      CALL CONDAC(IPLARF)
C
C   ** IPLANO=PLAYER NUMBER
C   ** IPTRPL=OFFSET IN ARRAY ITEM,DITEM TO CHANGE VALUES OF
```

FSN MODEL - DECIDE SUBROUTINE

ISN

```

C          ANY FIELDS FOR PLAYER IPLARF
C
32         CALL PARSRF(IPLARF, IDY, IPLANO, IDU)
33         IPTRPL=IREF1(IPLARF)
34         IPLAYR=IPLANO
35         CALL CHGVAR(3, IPLAYR)

C          -- TEST IF INSIDER
C
36         IF (IVAL(JTYPE, IPLARF) .EQ. KINSDR) GO TO 60
C          -- TEST IF PLAYER SUPPRESSED
C
37         IF (VAL(JSUPRN, IPLARF) .LE. 0.0) GO TO 11
C          -- PLAYER IS SUPPRESSED ; IF PERCEIVED TO BE UNDER FIRE
C              REMOVE SUPPRESSION; IF NOT, DECREASE SUPPRESSION
C              BY SUPDEC(ISIDE)
C
38         CALL DETFIR(IPLARF, ITARRF)
39         IF (ITARRF .EQ. NULL) GO TO 13
C          -- PLAYER PERCEIVED TO BE UNDER FIRE
C              REMOVE SUPPRESSION
C
40         11 DITEM(IPTRPL+JSUPRN)=0.0
41         GO TO 15
C          -- PLAYER NOT PERCEIVED TO BE UNDER FIRE,
C              DECREASE SUPPRESSION BY SUPDEC(ISIDE)
C
42         13 DITEM(IPTRPL+JSUPRN)=DITEM(IPTRPL+JSUPRN)
*                  -SUPDEC(ISIDE)
C          -- SEARCH ACTIVITY LIST , FIND IF PLAYER IS INVOLVED IN
C              CAPTURE.

```

FSN MODEL - DECIDE SUBROUTINE

15N

```
C
C ** IACTLT= LIST REFERENCE FOR ACTIVITES OF PLAYER
C
43   15 IACTLT= IVAL(JACTIV,IPLARF)
C
C -- FIND LIST ACTION REFERENCE ON LIST
C ** IACTRF= CURRENT ACTION REFERENCE OF PLAYER
C
44   IACTRF=IFIRST(IACTLT,KB)
45   18 IF(IACTRF .EQ. NULL) GO TO 40
C
C -- TEST FOR ACTION OF CAPTURING
C
46   IF(IVAL(JTYPE,IACTRF) .EQ. KCAPNG) GO TO 20
C
C -- FIND NEXT ACTIVITY ON LIST
C
47   IACTRF=NEXT(IACTLT,KB)
48   GO TO 18
C
C -- PLAYER INVOLVED IN CAPTURING ACTIVITY
C     DETERMINE IF ACCETPTING CAPTURE
C
49   20 IF(IVAL(JPAR1,IACTRF) .EQ. IPLARF) GO TO 30
C
C -- PLAYER IS ACCEPTING CAPTURE, DETERMINE IF TIME HAS
C     PASSED TO MAKE CAPTURE EFFECTIVE.
C ** TMIN=GLOBAL VARIABLE FOR GAME TIME
C
50   IRF=IVAL(JMODS,IACTRF)
51   PAR1=VAL(JPAR2,IRF)
52   IF(PAR1 .GT. TMIN) GO TO 25
C
C -- TIME HAS PASSED FOR CAPTURE TO BECOME EFFECTIVE
C     REMOVE CAPTURE ACTIVITY FROM ACTIVITY LIST
C
```

FSN MODEL - DECIDE SUBROUTINE

ISN
53 CALL DELIST(KB,JACTIV,IPLARF)
54 GO TO 40
C
C -- PLAYER STILL IN CAPTURE ACTIVITY
C RESTRICTED IN MOVEMENT
C
55 25 VTHRES=1.1
56 GO TO 45
C
C -- PLAYER IS SURRENDERING, TEST IF CAPTURING PLAYER
C IS DEAD
C
57 30 IDORF=IVAL(JDOER,IACTR)
58 IF(IVAL(JPSTAT, IDORF) .NE. Kdead) GO TO 32
C
C -- CAPTURE UNSUCCESSFUL, REMOVE ACTIVITY FROM LIST
C
59 CALL DELIST(KB,JACTIV,IPLARF)
60 GO TO 40
C
C -- TEST IF TIME HAS PASSED TO COMPLETE CAPTURE,
C IF NOT THE PLAYER SUPPRESSED IN OBSERVATION
C
61 32 IRF=IVAL(JMODS,IACTR)
62 PAR1=VAL(JPAR2,IRF)
63 IF(PAR1 .GT. TMIN) GO TO 38
C
C -- TIME HAS PASSED
C CHANGE PLAYER STATUS TO CAPTURED
C
64 ICOND=IVAL(JPSTAT,IPLARF)
65 CALL CHGFLD(JPSTAT,IPLARF,KCAPTR,1, IDUM)
C
C DECREASE NUMBER OF PLAYERS ON FORCES
C
66 FORCES(ICOND,ISIDE)=FORCES(ICOND,ISIDE)-1.0

FSN MODEL - DECIDE SUBROUTINE

ISN

67 CALL CHGVR2(5, FORCES(ICOND, ISIDE), ICOND, ISIDE)
68 ICOND=KCAPTR
69 FORCES(ICOND, ISIDE)=FORCES(ICOND, ISIDE)+1.0
70 CALL CHGVR2(5, FORCES(ICOND, ISIDE), ICOND, ISIDE)
71 CALL DEAD(IPLARF)

C

C -- DELETE PLAYER FROM LIST OF PLAYERS AND
C FIND NEXT PLAYER ON LIST
C

72 35 CALL DELIST(KA, JL, ISITE)
73 GO TO 500

C

C -- MARK PLAYER SUPPRESSED IN OBSERVATION
C

74 38 DITEM(IPTRPL+JSUPRN)=SUPOBS(ISIDE)
75 GO TO 500

C

C -- SET THRESHOLDS ACCORDING TO SUPPRESSION
C ** VTHRES=TEMPORARY MOVEMENT THRESHOLD
C ** FTHRES=TEMPORARY FIRING THRESHOLD
C ** STHRES=TEMPORARY THRESHOLD FOR SURRENDERING
C

C -- SET MOVEMENT THRESHOLD
C

76 40 IF(VAL(JSUPRN, IPLARF) .GT. (1.-RESMOV(IPLANO))) GO TO 43
77 VTHRES=RESMOV(IPLANO)
78 IF(IVAL(JPSTAT, IPLARF) .EQ. KWOUND)
* VTHRES=VTHRES+RESWND(IPLANO)
79 GO TO 45
80 43 VTHRES=1.1

C

C -- SET FIRE THRESHOLD
C

81 45 IF(VAL(JSUPRN, IPLARF) .GT. (1.-RESFIR(IPLANO))) GO TO 48
82 FTHRES=RESFIR(IPLANO)
83 IF(IVAL(JPSTAT, IPLARF) .EQ. KWOUND)

FSN MODEL - DECIDE SUBROUTINE

ISN

```
* FTHRES=FTHRES+RESWND(IPLANO)
84      GO TO 50
85      48 FTHRES=1.1
C
C      -- SET SURRENDER THRESHOLD
C
86      50 STHRES=RESSUR(IPLANO)
87      GO TO 100
C
C      -- INSIDER TEST FOR PLANS
C
88      60 IF (IVAL(JPLANS,IPLARF) .EQ. NULL) GO TO 70
C
C      -- SET THRESHOLD FOR INSIDER
C
89      VTHRES=-999.0
90      FTHRES=-999.0
91      STHRES=999.0
92      GO TO 100
C
C      -- NO PLANS THEN INSIDER SURFACES
C
93      70 CALL SURFAC(IPLARF)
94      GO TO 300
C
C      -- BEGIN DECISION MAKING PROCESS
C
C      -- TEST FOR PLANS,
C          IF NO PLANS GO TO GENERAL DECISION MAKING
C
C
95      100 IPLANS=IVAL(JPLANS,IPLARF)
96      MVPLAN=0
97      MFPLAN=0
C
```

FSN MODEL - DECIDE SUBROUTINE

ISN

C -- SET PLAYER TIME FOR ACTIVITY

C

98 DITEM(IPTRPL+JTREQ)=DTSEC

C

C ** IACTRF=POINTER TO PLAYER ACTIVITY LIST

C

99 IACTRF=IVAL(JACTIV,IPLARF)

C

C -- FIND FIRST PLAN ACTIVITY ON LIST

C

100 IPLAN=IFIRST(IPLANS,KD)

101 IF(IPLAN .EQ. NULL) GO TO 300

C

C -- TEST IF PLAN IS FOR MOVING MISSION

C

102 IF(IVAL(JTYPE,IPLAN) .NE. KMOVNG) GO TO 150

C

C -- MOVEMENT PLAN, DETERMINE IF PLAYER IS SUPPRESSED

C IN MOVEMENT; IF SO, GO TO GENERAL RULES.

C

103 IF(VTHRES .GE. 1.0) GO TO 300

C

C ** IDESID= DESTINATION NODE ID, STORED AS PARAMETER 1

C OF MOVEMENT ACTION RECORD.

C

104 IDESID=IVAL(JPAR1,IPLAN)

105 IF(IDESID .EQ. NULL) GO TO 550

C

C -- MISSION TO MOVE

C ACTION RECORD PARAMETERS DEFINITION

C 1 -- DESTION LOCATION REFERENCE

C 2 -- PATH LIST REFERENCE , LIST ELEMENTS ARE

C LOACTION REFERENCES

C 3 -- NULL

C

C -- LOWER THRESHOLD FOR MOVEMENT

FSN MODEL - DECIDE SUBROUTINE

ISN C
106 VTHRES=VTHRES+RESPLN(IPLAN)
107 MVPLAN=1
C
C -- TEST IF PLAN IS COMPATABLE WITH DESTINATION
C
108 IPATLT=IVAL(JPAR2,IPLAN)
109 INODID=IFIRST(IPATLT,KE)
110 110 IF(INODID.EQ.NULL) GO TO 120
111 IF(INODID.EQ.IDESID) GO TO 125
112 INODID=NEXT(IPATLT,KE)
113 GO TO 110
C
C -- PATH IS NOT COMPATABLE WITH DESTINATION
C GENERATE NEW PATH
C
114 120 ILOC=IVAL(JLOCN,IPLARF)
115 ISOU=IVAL(JSINK,ILOC)
116 IF(ISOU.EQ.NULL) ISOU=IVAL(JSOURC,ILOC)
117 LISTRF=IPATH(IPLARF,ISOU,IDESID,TEMP)
118 GO TO 127
C
C -- PATH IS COMPATIBLE WITH DESTINATION
C
119 125 LISTRF=IPATLT
C
C -- EVALUATE PATH TO DESIRABILITY OF MOVEMENT
C
120 127 CALL MDESIR(IPLARF,LISTRF,TEMP)
121 IF(TEMP.LT.VTHRES) GO TO 300
C
C -- MOVE PLAN IS ACCEPTABLE, GENERATE NEW ACTION FOR
C PLAYER FROM PLANS
C
122 IF(LISTRF.EQ.IPATLT) GO TO 130
C

FSN MODEL - DECIDE SUBROUTINE

ISN

```
C   -- NEW PATH GENERATED, ADJUST PATH REFERENCE IN ACTION
C
123      IPTR=IREF1(IPLAN)
124      ITEM{IPTR+JPAR2}=LISTRF
C
C   -- ADD ACTIVITY TO PLAYER ACTIVITY LIST
C
125      130 IACTRF=ISTACK(IPLAN,IACTRF)
126      CALL CHGFLD(JACTIV,IPLARF,IACTRF,1,1DUM)
127      ISUCM=IFIRST(LISTRF,KKZ)
C
C
C   -- NO RELATIONS ADDED TO DETERMINE TERMINATION
C       CONDITION FOR MOVING ACTIVITY
C   -- SET PLAYER POSTURE FOR MOVING
C       FIND SUBACTIVITY FOR ACTIVITY MOVING
C       POSTURE IS INDEPENDENT OF COVER WHEN MOVING
C
128      140 ISUBAC=NACTSU(IPLARF,NULL)
129      DITEM(IPTRPL+JPOSTR)=POSTUR(ISUBAC)
130      GO TO 335
C
C   -- CHECK FOR FIRING PLAN
C
131      150 IF(IVAL(JTYPE,IPLAN) .NE. KFIRNG) GO TO 200
C
C   -- MISSION TO FIRE, TEST SUPPRESSION
C
132      IF(FTHRES .GE. 1.0) GO TO 300
C
C   -- LOWER THRESHOLD TO FOLLOW PLANS
C
133      FTHRES=FTHRES+RESPLN(IPLANO)
134      MFPLAN=1
C
C   -- NULL TARGET LIST
```

FSN MODEL - DECIDE SUBROUTINE

ISN

```
C
135      ITRGLT=NULL
C
C      -- DETERMINE IF PLAYER REFERENCE GIVEN AS TARGET
C
C          ACTION RECORD DEFINITION OF PARAMETERS
C              1. TARGET    PLAYER REFERENCE
C              2. LOCATION REFERENCE
C              3. WEAPON SELECTED
C              MODS. TERMINATION CONDITIONS
C
136      IF (IVAL (JPAR1,IPLAN) .EQ.NULL) GO TO 170
C
C      -- PLAYER REFERENCE GIVEN
C          TEST FOR DESIRABILITY TO FIRE
C
137      ITRGLT=IVAL (JPAR1,IPLAN)
C
C      -- CHECK FOR LINE OF SIGHT
C
138      ICC=LOS (IPLARF,ITRGLT)
139      IF (ICC .EQ. 0) GO TO 170
140      CALL FDESIR(IPLARF,ITRGLT,ICODE,TEMP,FTHRES)
141      IF (TEMP .GE. FTHRES) GO TO 195
C
C      -- GENERATE TARGET LIST    BASED ON REGION
C          GIVEN IN PLANS
C
142      170 ILOCID=IVAL (JPAR2,IPLAN)
143      CALL TRGLST(IPLARF,ILOCID,ITRGLT)
C
C      -- TEST IF TARGET LIST IS NULL
C
144      190 IF (ITRGLT .EQ. NULL) GO TO 300
C
C      -- TEST DESIRABILITY OF TARGETS ON TARGET LIST
```

448
ISN

FSN MODEL - DECIDE SUBROUTINE

C
145 CALL FDESIR(IPLARF,ITRGLT,ICODE,TEMP,FTHRES)
146 IF(TEMP .LT. FTHRES) GO TO 300
C
C -- TARGET FOUND, REFERENCE IS ICODE.
C SELECT WEAPON, IF NO WEAPON AVAILABLE GO TO GENERAL RULE
C
147 195 CALL NWPNNS(IPLARF,ICODE,IWPNRF,IWPN)
148 IF(IWPN .EQ. 0) GO TO 300
C
149 IPTR=IREF1(IPLAN)
150 ITEM(IPTR+JPAR1)=ICODE
151 ITEM(IPTR+JPAR2)=IVAL(JLOCN,ICODE)
152 ITEM(IPTR+JPAR3)=IWPNRF
C
C -- ADD ACTION TO PLAYER ACTIVITY LIST
C
153 IACTRF=ISTACK(IPLAN,IACTRF)
154 CALL CHGFLD(JACTIV,IPLARF,IACTRF,1,1DUM)
C
C -- SET TERMINATION CONDITIONS FOR FIRING
C
155 198 CALL PARSRF(ICODE,1DY,IRR,1DUN)
156 IPAR1=NEWREF(1DY,IRR,JPSTAT)
157 IPAR2=KDEAD
158 IRF = NEWRC3(LRELN,KEQ,IPAR1,IPAR2)
C
C -- ADD RELATION TO ACTIVITY
C
159 IPTR=IREF1(IACTRF)
160 ITEM(JMODS+IPTR)=IRF
C
C -- DETERMINE SUB-ACTIVITY FOR PLAYER
C
161 ISUBA=NACTSU(IPLARF,NULL)
162 FTEM=FNULL

FSN MODEL - DECIDE SUBROUTINE

ISM
163 DITEM(IPTRPL+JPOSTR)=POSTUR(ISUBA)+COVR(IPLARF,FTEM)
164 GO TO 500
C
C -- CHECK FOR OBSERVATION PLAN
C
165 200 IF(IVAL(JTYPE,IPLAN) .NE. KDETNG) GO TO 560
C
C -- OBSERVATION MISSION, TEST IF PLAYER
C PERCEIVED TO BE UNDER FIRE
C
166 CALL DETFIR(IPLARF,ITARRF)
167 IF(ITARRF .NE. NULL) GO TO 300
C
C -- NOT PERCEIVED TO BE UNDER FIRE,
C IMPLEMENT OBSERVE ACTIVITY
C
C -- ACTION RECORD PARAMETERS DEFINITION
C
C 1. DIRECTION REAL VALUE IN RADIANS
C 2. REGION LOCATION REFERENCE
C MODS. TERMINATION CONDITIONS
C
168 IPTR=IREF1(IPLAN)
169 IF(VAL(JPAR1,IPLAN) .NE. FNULL) GO TO 230
170 INODID=IVAL(JPAR2,IPLAN)
171 IF(INODID .EQ. NULL) GO TO 220
172 CALL DIREC(IPLARF,INODID,THETA,PHI)
173 GO TO 225
174 220 THETA=FNULL
175 225 DITEM(IPTR+JPAR1)=THETA
C
176 230 IACTRF=ISTACK(IPLAN,IACTRF)
177 CALL CHGFLD(JACTIV,IPLARF,IACTRF,1,1DUM)
C
C -- NO TERMINATION CONDITIONS FOR OBSERVING
C ASSUMED ACTIVITY OCCURS FOR 1 TIME CYCLE

FSN MODEL - DECIDE SUBROUTINE

ISN

```
C -- DETERMINE SUBACTIVITY FOR OBSERVING
C
178   240 ISUBA=NACTSU(IPLARF,NULL)
179       DITEM(IPTRPL+JPOSTR)=POSTUR(ISUBA)+COVR(IPLARF,THETA)
180       GO TO 500
C
C -- GENERAL DECISION RULE
C -- TEST FOR LOCAL MOVE, IF ON LINK TEST END NODES
C     IF AT NODE TEST NEIGHBORS
C
181   300 IF(VTHRES .GT. 1.0) GO TO 350
182       ILOCRF=IVAL(JLOCN,IPLARF)
183           IF(VAL(JFRAC,ILOCRF) .GT. 0.0) GO TO 310
C
C -- PLAYER IS AT NODE, TEST IF THIS AN ADVERSARY
C
184       IF(IVAL(JALLEG,IPLARF) .EQ. KSFE) GO TO 305
C
C THIS IS AFE PLAYER, TEST IF AT GOAL NODE
C     SEARCH LIST OF ACTIVE GOALS
C
185       IPLACE=IVAL(JPLACE,ILOCRF)
C
C ** IGOALS - POINTER TO A LIST OF ACTIVE GOALS
C
186       IGL=IFIRST(IGOALS,KP)
187       303     IF(IGL .EQ. NULL) GO TO 305
188           IF(IVAL(JID,IGL) .EQ. IPLACE) GO TO 350
C
189       IGL=NEXT(IGOALS,KP)
190       GO TO 303
C
C -- PLAYER AT NODE BUT NOT GOAL NODE
C     LIST ALL NEIGHBORS OF NODE
C
191       305 INODRF=IVAL(JPLACE,ILOCRF)
```

FSN MODEL - DECIDE SUBROUTINE

ISN
192 LISTRF=IVAL(JNBRS,INODRF)
C
C -- ADD LOCATION NODE
C
193 INODE=IVAL(JSOURC,ILOCRF)
194 LISTRF=ISTACK(INODE,LISTRF)
195 GO TO 320
C
C -- PLAYER NOT AT NODE, CREATE LIST OF NODES
C AS END NODES OF LINK
C
196
197 310 INOD1=IVAL(JSOURC,ILOCRF)
198 INOD2=IVAL(JSINK,ILOCRF)
199 LISTRF=LIST(INOD1,INOD2,NULL,NULL,NULL,NULL)
C
C -- EVALUATE EACH NODE ON LISTRF
C FOR DESIRABILITY
C
200 320 TTHRES=-1.0
201 ISUCM=NULL
202 INODE=IFIRST(LISTRF,KG)
203 321 IF(INODE .EQ. NULL) GO TO 330
C
C -- EVALUATE DESIRABILITY
C
204 CALL MDESIR(IPLARF,INODE,TEMP)
205 IF(TEMP .LT. VTHRES) GO TO 325
206 IF(TEMP .LE. TTHRES) GO TO 325
207 TTHRES=TEMP
208 ISUCM=INODE
C
C -- NEXT NODE ON LIST
C
209 325 INODE=NEXT(LISTRF,KG)
210 GO TO 321

FSN MODEL - DECIDE SUBROUTINE

ISN

C

C -- TEST IF ANY MOVE WAS DESIRABLE

C

211 330 IF (ISUCM .EQ. NULL) GO TO 340

212 IF (ISUCM .EQ. IVAL(JSOURC,ILOCRF)) GO TO 350

C

C -- MOVE SUCCUSSFUL,

C RULE IF MOVE FOUND DESIRABLE THEN MOVE

C -- CREATE MOVE ACTION

C

213 CALL CREATC(KMOVNG,ISUCM,ISUCM,NULL,NULL,IPLARF,INEWRF)

C

C SET PLAYER POSTURE,

C

214 IACTS=NACTSU(IPLARF,NULL)

215 DITEM(IPTR+JPOSTR)=PCSTUR(IACTS)

C

C -- DO NOT ADD ACTION REFERENCE TO PLANS

C -- TEST IF DESTINATION IS SOURCE NODE

C

216 335 ILOCRF=IVAL(JLOCN,IPLARF)

217 IF (IVAL(JSOURC,ILOCRF) .NE. ISUCM) GO TO 500

C

C -- REVERSE DIRECTION ALONG LINK

C

218 IPTR=IREF1(ILOCRF)

219 ITEM(IPTR+JSOURC)=IVAL(JSINK,ILOCRF)

220 ITEM(IPTR+JSINK)=ISUCM

221 DITEM(IPTR+JFRAC)=1.0-DITEM(IPTR+JFRAC)

222 GO TO 500

C

C -- NOT DESIRABLER TO MOVE

C CHECK IF GIVEN PLAN TO MOVE;

C IF SO , THEN MARK PLAYER SUPPRESSED IN

C MOVEMENT

C

FSN MODEL - DECIDE SUBROUTINE

ISN

```
223      340 IF (MVPLAN .EQ. 0) GO TO 350
C
C      --PLAYER GIVEN PLAN TO MOVE BUT DID NOT
C          DECIDE TO MOVE -- SUPPRESSED IN MOVEMENT
C
224      DITEM (IPTRPL+JSUPRN)=DITEM (IPTRPL+JSUPRN) +
*                      SUPMOV (ISIDE)
225      TEM=VAL (JSUPRN, IPLARF)
226      CALL CHGFLD (JSUPRN, IPLARF, TEM, 1, IDUM)
C
C-- GENERAL DECISION
C      FIRE AT TARGET
C      -- GENERATE TARGET LIST
C
227      350 ITRGLT=NULL
228      ILOCN=NULL
229      CALL TRGLST (IPLARF, ILOCN, ITRGLT)
230      IF (ITRGLT .EQ. NULL) GO TO 400
C
C      -- TEST DESIRABILITY OF TARGETS ON LIST
C
231      CALL FDESIR (IPLARF, ITRGLT, ICODE, TEMP, FTHRES)
C
232      IF (TEMP .LT. FTHRES) GO TO 390
C
C      -- TARGET FOUND, REFERENCE IS ICODE
C          CREATE ACTION RECORD
C      -- SELECT WEAPON
C
233      CALL NWPN (IPLARF, ICODE, IWPNRF, IWPN)
234      IF (IWPN .EQ. 0) GO TO 390
235      CALL CREAT (KFIORG, ICODE, NULL, IWPNRF, NULL, IPLARF, INEWRF)
C
C      -- ADD ACTIVITY TO PLANS, THIS WILL ALLOW
C          ACTION TO CARRY OVER SEVERAL TIME
C          INTERVALS
```

FSN MODEL - DECIDE SUBROUTINE

FSN MODEL - DECIDE SUBROUTINE

ISN

C -- CALCULATE DIRECTION OF OBSERVATION

C

250 CALL DIREC(IPLARF,ICODE,THETA,PHI)

C

C -- ADD ACTION TO PLAYER ACTIVITY LIST

C

251 425 CALL CREAT(KDETNG,THETA,NULL,NULL,NULL,IPLARF,INEWRF)

C

252 GO TO 240

C

C -- GENERATE DEFAULT ANGLE

C

253 430 THETA=FNNULL

254 GO TO 425

C

C -- SUPPRESSED IN OBSERVATION,

C CHECK FOR SURRENDER

C

255 450 CALL SDESIR(IPLARF,TEMP)

256 IF(TEMP .LT. STHRES) GO TO 490

C

C -- PLAYER DECIDES TO SURRENDER,

C SURRENDER TO PLAYER PERCEIVED TO

C BE UNDER FIRE FROM ITARRF

C -- REMOVE ALL ACTIVITES EXCEPT CAPTURE FROM ACTIVITY

C LIST FOR ITARRF

C

257 INEWRF=NULL

258 IACTLT=IVAL(JACTIV,ITARRF)

259 IACTRF=IFIRST(IACTLT,FR)

260 455 IF(IACTRF .EQ. NULL) GO TO 470

C

C -- TEST IF ACTIVITY IS CAPTURE

C

261 IF(IVAL(JTYPE,IACTRF) .NE. KCAPNG) GO TO 460

C

FSN MODEL - DECIDE SUBROUTINE

1SN
C -- ADD ACTIVITY TO NEW LIST
C
262 INEWRF=IQUEUE(IACTR,INEWRF)
C
C -- NEXT ACTIVITY
C
263 460 IACTR=NEXT(IACTLT,KR)
264 GO TO 455
C
C -- CHANGE ACTION REFERENCE TO NEW LIST
C
265 470 IPTR=IREF1(ITARRF)
266 ITEM(IPTR+JACTIV)=INEWRF
C
C -- CREATE ACTION CAPTURE ADD TO
C ITARRF ACTIVITY LIST
C -- TERMINATION CONDITIONS FOR CAPTURING
C
267 ITYPE=KGEF
268 IPAR1=NEWREP(1,1,JTMIN)
269 IPAR2=TMIN+ACTTIM(KCPTRG)/60.
270 IRF = NEWRC3(LRELN,ITYPE,IPAR1,IPAR2)
271 CALL CREACT(KCAPNG,IPLARF,NULL,NUL,IIRF,ITARRF,INEWRF)
C
C -- TEST IF IPLARF ACCEPTING CAPTURE
C
272 IACTLT=IVAL(JACTIV,IPLARF)
273 IACTRF=IFIRST(IACTLT,KS)
274 473 IF(IACTR.EQ.NULL) GO TO 480
C
C -- TEST IF ACTIVITY IS CAPTURING
C
275 IF(IVAL(JTYPE,IACTR).NE.KCAPNG) GO TO 475
C
C -- PLAYER INVOLVED IN CAPTURE, RELEASE
C CAPTURE ACTIVITY

FSN MODEL - DECIDE SUBROUTINE

ISN

```
C
276      ICAPP=IVAL(JPAR1,IACTRF)
277      IPTR1=IREF1(ICAPP)
278      ITEM(IPTR1,JACTIV)=NULL
C
C      -- NEXT ACTIVITY
C
279      475 IACTRF=NEXT(IACTLT,KS)
280      GO TO 473
C
C      -- ADD CAPTURE EVENT TO IPLARF ACTIVITY LIST
C
281      480 IACTRF=IVAL(JACTIV,IPLARF)
282      IACTRF=ISTACK(INEWRP,IACTRF)
283      ITEM(IPTRPL+JACTIV)=IACTRF
C
C      -- MARK PLAYER SUPPRESSED IN OBSERVATION
C
284      490 TEM=SUPOBS(ISIDE)
285      CALL CHGFLD(JSUPRN,IPLARF,TEM,1,1DUM)
C
C      -- NEXT PLAYER
C
286      500 MSITN=IVAL(JSITN,IPLARF)
287      MSITN=MINO(MSITN,KSITN)
288      IPTR=IREF1(IPLARF)
289      ITEM(IPTR+JSITN)=MSITN
290      IPLARF=NEXT(IPLA$T,KA)
291      GO TO 0
C
C      -- GUARD OR ADVERSARY LIST COMPLETE
C
292      510 IF(JL .EQ. JADVRS) GO TO 500
293      JL=JADVRS
294      ISIDE=2
295      GO TO 5
```

FSN MODEL - DECIDE SUBROUTINE

ISM

```
C
C
C      **** ERROR-
C          MOVE MISSION DOES NOT CONTIAN DESTINATION
C
296      550 I1=IVAL(JPAR1,IPLAN)
297          I2=IVAL(JPAR2,IPLAN)
298          I3=IVAL(JPAR3,IPLAN)
299          CALL ERR(24,26,I1,I2,I3)
300          GO TO 600
C
C      ***** ERROR-
C          PLAN GIVEN BUT ILLGAL ACTIVITY TYPE
C
301      560 I1=IVAL(JTYPE,IPLAN)
302          CALL ERR(25,26,I1,I2,I3)
C
303      600 CONTINUE
C
304      RETURN
305      END
```

FSN MODEL - DETFIR SUBROUTINE

1 SUBROUTINE DETFIR(IPLARF,ITARRF)

T THE PURPOSE OF THIS SUBROUTINE IS TO:

1. DETERMINE IF PLAYER REFERENCED HAS A
PERCEPTION OF BEING FIRED UPON.

-- INPUT PARAMETERS

IPLARF - PLAYER REFERENCE

-- OUTPUT VARAIBLES

ITARRF - PLAYER REFERENCE DESIGNATING THE PERSON
PERCEIVED BY IPLARF TO BE FIRING AT HIM.
VALUE IS NULL WHEN NOT PERCEIVED TO
BE UNDER FIRE

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4 REAL#8 DTPNAM,FLDNAM, FORMOT
5 COMMON /PARS1/
5 COMMON /PARS2/
7 COMMON /PARS3/

-- FIND POINTER TO LIST OF PERCEPTIONS

8 IPERLT=IVAL(JPPCFS,IPLARF)

-- SEARCH LIST OF PERCEPTIONS, IPERRF IS PERCEPTION RECORD
REFERENCE

FSN MODEL - DETFIR SUBROUTINE

1 ISN
2
3 9 IPERRF=IFIRST(IPERLT,KW)
4 10 IF(IPERRF .EQ. NULL) GO TO 50
5 C
6 C -- FIND REFERENCE RECORD THAT DESCRIBES PERCEPTION
7 C
8 11 ICURRF=IVAL(JVIEW,IPERRF)
9 C
10 12 CALL PARSPE(ICURRF, IDYY,IRECN, IDUM)
11 IF(IDYY .NE. EPERSN) GO TO 40
12 C
13 C -- TEST PERSONS ACTUAL PHYSICAL STATUS
14 C
15 14 ITRUE=IVAL(IID,IPERRF)
16 IF(IVAL(IPSTAT,ITRUE) .LE. KCAPTR) GO TO 40
17 C
18 C -- FIND ACTIVITY LIST FOR PERCEPTION
19 C
20 16 IACTLT=IVAL(JACTIV,ICURRF)
21 C
22 C -- SEARCH LIST OF PERCEIVED ACTIVITIES TO FIND FIRING
23 C
24 17 IACTRF=FIRST(IACTLT,KX)
25 20 IF(IACTRF .EQ. NULL) GO TO 40
26 IF(IVAL(JTYPE,IACTRF) .NE. KFIRNG) GO TO 30
27 C
28 C -- FINDING ACTIVITY FOUND, DETERMINE IF IPLARE IS TARGET
29 C
30 20 IF(IVAL(JPAP1,IACTRF) .EQ. IPLARE) GO TO 60
31 C
32 C -- NEXT ACTIVITY PERCEIVED
33 C
34 21 IACTRF=NEXT(IACTLT,KX)
35 22 GO TO 20
36 C
37 C -- NEXT PERCEPTION
38 C

FSN MODEL - DETFIR SUBROUTINE

1SN
23 40 IPERRF=NEXT(IPERLT,KW)
24 GO TO 10
C
C -- LIST OF PERCEPTIONS SEARCHED
C NOT PERCEIVED TO BE UNDER FIRE
C
25 50 ITARFF=NULL
26 GO TO 70
C
C -- PERCEIVED TO BE UNDER FIRE
C
27 60 TTARFF=TMAL(JID,IPERRF)
C
28 70 CONTINUE
29 RETURN
30 END

FSN MODEL - DRCOM1 SUBROUTINE

ISM

```
1      SUBROUTINE DRCOM1(IPLARF,LMAX)
C ... THIS SUBROUTINE IS CALLED TO DETERMINE HOW MANY ITEMS
C OF INFORMATION A PLAYER CAN ASSIMILATE AND COMMUNICATE
C DURING A SINGLE GAME CYCLE.
C
C ... INPUT PARAMETERS:
C           IPLARF ... POINTER TO PLAYER DESCRIPTION RECORD
C
C ... OUTPUT PARAMETERS:
C           LMAX   ... MAX NUMBER OF ITEMS
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /DATAV/
8      DIMENSION ACTTIM(25)
9      EQUIVALENCE (ACTTIM(1),ACTRAT(1))
C
C ... CURRENTLY, DATA TAKEN DIRECTLY FROM HPLIMS
C
10     LMAX=HFLIMS(KLIM1,ISIDE)
11     IF (IVAL(JTYPE,IPLARF).EQ.KSNMON) LMAX=HFLIMS(KLIM2,ISIDE)
12     RETURN
13     END
```

ISN

FSN MODEL - DRCOM2 SUBROUTINE

```
1      SUBROUTINE DRCOM2(IPLARF,LMSN,LMSG,LINCP,LORDER)
C ... CALLED BY COMPER TO ESTABLISH LIMITS FOR MESSAGE
C PROCESSING BY A PLAYER.
C ... INPUT PARAMETERS:
C           IPLARF ... PLAYER RECORD REFERENCE
C
C ... OUTPUT PARAMETERS
C           LMSN ... # OF MISSION MESSAGES
C           LMSG ... # OF NON-MISSION MESSAGES
C           LINCP ... # OF INTERCEPTED MESSAGES THAT
C                         MAY BE SAVED AND RETRANSMITTED
C           LORDER... # OF ORDERS FROM LEADER
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS2/
6      COMMON /DATAV/
7      DIMENSION ACTTIM(25)
8      EQUIVALENCE (ACTTIM(1),ACTRAT(1))
C
9      LMSN=HFLIMS(KLIM3,ISIDE)
10     LMSG=HFLIMS(KLIM5,ISIDE)
11     LORDER=HFLIMS(KLIM4,ISIDE)
C
C ... LIMIT ON INTERCEPTED MESSAGES NOT PRESENTLY
C ENTERED EXPLICITLY.
C
12    LINCP=HFLIMS(KLIM1,ISIDE)
13    RETURN
14    END
```

FSN MODEL - DRCOM3 SUBROUTINE

ISN

```
1      SUBROUTINE DRCOM3(IPLARF,NETREF,INTCPT)
C ... SUBROUTINE TO DETERMINE IF A PLAYER WHO CAN
C     RECEIVE ON A NET IS RECEIVING MESSAGES DIRECTED
C     TO HIM, OR IS INTERCEPTING THE OTHER SIDE'S
C     TRANSMISSION.
C
C ... INPUT PARAMETERS:
C     IPLARF ... PLAYER RECORD REFERENCE
C     NETREF ... COMNET RECORD REFERENCE
C
C ... OUTPUT PARAMETERS:
C     INTCPT ... .TRUE. PLAYER INTERCEPTING
C                 .FALSE. PLAYER VALID NET MEMBER
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     LOGICAL INTCPT
11     DIMENSION KOUNT(2)
C
C ... SEE IF PLAYER CAN XMIT ON THIS NET
C
12     INTCPT=.FALSE.
13     NETLIS=IVAL(JCMNTS,IPLARF)
14     IXMIT=LSEARCH(NETLIS,NETREF,NULL,NULL,NULL)
15     IF(IXMIT.NE.NULL) RETURN
C
C ... PLAYER CAN NOT TRANSMIT ON THIS NET
C     DETERMINE ALLEGIANCE OF PLAYER'S THAT CAN
C
```

PSN MODEL - DRCOM3 SUBROUTINE

ISN

```
16      IRCLIS=IVAL(JRCVRS,NETREF)
17      IRCVRF=IFIRST(IRCLIS,LPTR)
18      KOUNT(KAFE)=0
19      KOUNT(KSFE)=0
20      10     IF(IRCVRF.EQ.NULL) GO TO 20
21      INDEX=IVAL(JALLEG,IRCVRP)
22      KOUNT(INDEX)=KOUNT(INDEX)+1
23      IRCVRF=NEXT(IRCLIS,LPTR)
24      GO TO 10
C
C ... IF KOUNT(ISIDE) IS MAX, NO INTERCEPT
C
25      20     IF(KOUNT(ISIDE).GE.KOUNT(3-ISIDE)) RETURN
C
C ... PLAYER IS INTERCEPTING
C
26      INTCPT=.TRUE.
27      RETURN
28      END
```

FSN MODEL - DROBS1 SUBROUTINE

ISN

```
1      SUBROUTINE DROBS1
C ... DROBS1 IS CALLED TO FORM A DUMMY OBJECT DESCRIPTION
C RECORD CONTAINING ONLY THOSE ATTRIBUTES THAT A PLAYER
C MAY OBSERVE ABOUT AN OBJECT. IN THE CASE A PLAYER IS
C "OBSERVING" HIMSELF A COPY OF HIS DESCRIPTION IS
C RETURNED.
C
C INPUT PARAMETERS: (IN BLANK COMMON)
C     IPLARF ... POINTS TO DESCRIPTION OF OBSERVER
C     IOBSRF ... POINTS TO RECORD OF OBJECT SEEN
C
C OUTPUT PARAMETERS: (IN BLANK COMMON)
C     IDUMRF ... POINTS TO PSEUDO RECORD CREATED
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NULL),(FAIL,PAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMAT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON // LMAX,PMIN,IMSG(50),PMSG(50),NMSG,POBSV,LOWMSG
+       IPLARF,IOBSRF,JOBSRF,IDUMRF,IASMNT(19),IPCPRF
C
C ... FOR SELF OBSERVATION CREATE COPY OF PLAYER DESCRIPTION
C
12    IF (IPLARF.NE.IOBSRF) GO TO 10
13    IDUMRF=ICOPY(IPLARF)
14    RETURN
C
C ... SET UP FOR PROCESSING OF NORMAL CALL
C
15    10    CALL PARSRF(IOBSRF,IDLTYPE,IRECNO,IFLD)
16    IF (IDLTYPE.EQ.LVEHIC) GO TO 100
```

FSN MODEL - DROBS1 SUBROUTINE

ISN

17 IF (IDTYP.EQ.LPERSN) GO TO 200
18 IF (IDTYP.EQ.LSENS) GO TO 300
19 IDUMRF=NULL
20 RETURN

C

C ... PROCESS OBSERVED VEHICLE

C

21 100 NFLD=NPLDS(LVEHIC)
22 IDUMRF=NEWREF(LVEHIC, NEWREC(LVEHIC), 0)
23 IPTR = IREF1(IDUMRF)
24 DO 150 I=1,NPLD
25 IF (I.EQ.JCONTS .OR. I.EQ.JVEL) GO TO 150
26 ITEM(IPTR+I) = IVAL(I, IOBSRF)

27 150 CONTINUE
28 RETURN

C

C ... PROCESS OBSERVED PERSON

C

29 200 IDUMRF=NEWREF(LPERSN, NEWREC(LPERSN), 0)
30 IPTR = IREF1(IDUMRF)
31 DO 250 I=1,JPOSTR
32 IF (I .EQ. JSUPRN) GO TO 250
33 ITEM(IPTR+I) = IVAL(I, IOBSRF)

34 250 CONTINUE

C

C ... PROCESS ACTIVITY

C

35 LISACT=IVAL(JACTIV, IOBSRF)
36 IACTRF=IFIRST(LISACT, LPTACT)
37 260 IF (IACTRF.EQ.NULL) GO TO 280
38 IACTYP=IVAL(JTYPE, IACTRF)
39 IF (IACTYP.LT.KDETNG) GO TO 265

C

C ... SUPPRESSION STATUS MAY BE OBSERVED

C

40 ITEM(IPTR+JSUPRN) = IVAL(JSUPRN, IOBSRF)

FSN MODEL - DROBS1 SUBROUTINE

ISN
41 GO TO 280
C
C ... ACTIVITY AND EQUIPMENT MAY BE OBSERVED
C
42 265 ITEM(IPTR+JACTIV) = IACTRF
43 ITEM(IPTR+JWEAPS) = IVAL(JWEAPS,IOBSRF)
44 GO TO 280
45 270 IACTRF=NEXT(LISACT,LPTACT)
46 GO TO 260
47 280 RETURN
C
C ... PROCESS OBSERVED SENSOR
C
48 300 IDUMRF=NEWREF(LSENS,NEWREC(LSENS),0)
49 IPTR = IREY1(IDUMRF)
50 ITEM(IPTR+JTYPE) = IVAL(JTYPE,IOBSRF)
51 ITEM(IPTR+JPSTAT) = IVAL(JPSTAT,IOBSRF)
52 ITEM(IPTR+JLOCN) = IVAL(JLOCN,IOBSRF)
53 RETURN
54 END

FSN MODEL - DROBS2 SUBROUTINE

1SN

```
1      SUBROUTINE DROBS2(ISENSR)
C ... DECISION RULE CALLED BY COMOBS TO FORM A DUMMY
C     OBJECT DESCRIPTION CONTAINING ONLY THOSE FEATURES OF
C     AN OBJECT THAT COULD BE DISCERNED BASED ON THE TYPE
C     OF SENSOR REPORT.
C
C ... INPUT PARAMETERS: (OR IN BLANK COMMON)
C     ISENSR ... SENSOR REPORT TYPE
C     IPLARF ... POINTS TO RECORD OF OBSERVER
C     IOBSRF ... POINTS TO RECORD OF OBJECT TRIPPING SENSOR
C ... OUTPUT PARAMETERS: (IN BLANK COMMON)
C     IDUMRF ... POINTS TO PSEUDO RECORD CREATED
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900), DITEM(41900)
4      EQUIVALENCE (DTMIN, ITEM(1), DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL, NULL), (FAIL, FAIL)
7      REAL*8 DTPNAM, FLDNAM, FORMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON // LMAX, PMIN, IMSG(50), PMSG(50), NMSG, POBSV, LOWMSG
+        IPLARF, IOBSRF, JOBSRF, IDUMRF, IASMNT(19), IPCPRF
C
C ... BRANCH ON SENSOR REPORT TYPE.  IF INVALID
C     RETURN NULL VALUE IN IDUMRF
C
11     IDUMRF=NULL
12     CALL PARSRF(IOBSRF, IDTYP,IRECNO,IFLD)
13     IF(ISENSR.EQ.KBINPL .OR. ISENSR.EQ.KBINRC) GO TO 100
14     IF(ISENSR.EQ.KPICFL .OR. ISENSR.EQ.KPICRC) GO TO 200
15     RETURN
C
C ... CURRENTLY BINARY SENSOR CAN OBSERVE LOCATION ONLY
C
16     100    IDUMRF=NEWREF(IDTYP, NEWREC(IDTYP), 0)
```

FSN MODEL - DROBS2 SUBROUTINE

ISN
17 I = IREF(JLOCN, IDUMRF)
18 ITEM(I) = IVAL(JLOCN, IOBSRF)
19 RETURN
C
C ... CURRENTLY PICTURE SENSOR TREATED LIKE NORMAL OBSERVATION
C
20 200 CALL DROBS1
21 300 RETURN
22 END

ISN

FSN MODEL - DROBS3 SUBROUTINE

1 SUBROUTINE DROBS3
C ... DECISION RULE CALLED BY COMOBS TO DETERMINE OBSERVATION
C PRIORITY BASED ON THE OBJECT BEING OBSERVED.
C ... INPUT PARAMETERS: (IN BLANK COMMON)
C IPLARF ... RECORD REFERENCE FOR OBSERVER DESCRIPTION
C IOBSRF ... RECORD REFERENCE FOR OBJECT TO BE OBSERVED
C ... OUTPUT PARAMETERS: (IN BLANK COMMON)
C POBSV ... OBSERVATION PRIORITY
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900), DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL), (FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 COMMON /DATAV/
11 DIMENSION ACTTIM(25)
12 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
13 COMMON // LMAX,PMIN,IMSG(50),PMSG(50),NMSG,POBSV,LOWMSG
+ IPLARF,IOBSRF,JOBSRF,IDUMRF,IASMNT(19),IPCPRF
C
C ... UNDER CURRENT RULE ONLY VEHICLES, PERSONS AND
C SENSORS MAY BE OBSERVED.
C
14 POBSV=0.0
15 CALL PARSRF(IOBSRF, IDTYP, IRECNO, IDUM)
16 IF (IDTYP.EQ.LVEHIC) GO TO 100
17 IF (IDTYP.EQ.LPERSH) GO TO 200
18 IF (IDTYP.EQ.LSENS) GO TO 300
19 RETURN
C
C ... OBJECT IS VEHICLE
C
20 100 POBSV=POBVEH(1SIDE,IVAL(JOWNER,IOBSRF))

FSN MODEL - DROBS3 SUBROUTINE

150 ISM

21 RETURN

22 C

23 C ... OBJECT IS PERSON

24 C

25 200 IACTRF=IFIRST(IVAL(JACTIV,IOBSRF),LPTR)

26 IF(IACTRF.EQ.NULL) RETURN

27 IACT=IVAL(JTYPE,IACTRF)

28 IALG=IVAL(JALLEG,IOBSRF)

29 POBSV=POBPSN(ISIDE,IALG,IACT)

30 RETURN

31 C

32 C ... OBJECT IS SENSOR

33 C

34 300 POBSV=POBSEN(ISIDE)

35 RETURN

36 END

FSN MODEL - DROBS4 SUBROUTINE

ISN

```
1      SUBROUTINE DROBS4(P,IPLARF,IMSGRF)
C      THIS SUBROUTINE IS CALLED BY COMPER TO DETERMINE
C      PRIORITY ASSOCIATED WITH A PLAYER PROCESSING A
C      PARTICULAR MESSAGE HE HAS RECEIVED.
C ... INPUT PARAMETERS:
C         IPLARF ... REF TO PLAYER PROCESSING MESSAGE
C         IMSGRF ... REF TO MESSAGE
C ... OUTPUT PARAMETERS
C         P      ... ASSOCIATED PROCESSING PRIORITY
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMAT
8      COMMON /PARS1/
9      COMMON /PARS3/
10     COMMON /DATAV/
11     DIMENSION ACTTIM(25)
12     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
C
C ... GET ATTRIBUTE PRIORITY
C
13     P=0.0
14     IOBSRF=IVAL(JSUBJ,IMSGRF)
15     CALL PARSRF(IOBSRF,IDLTYPE,IRECNO,IFLD)
16     IATTR=IVAL(JATTR,IMSGRF)
17     INDEX=0
C
C ... COMPUTE INDEX=FEATURES OF OBJECTS
C
18     IFEA=IDLTYPE-LVEHIC+1
19     GO TO (30,20,10),IFEAS
20     10    INDEX=INDEX+NFLDS(LPERSN)
```

FSN MODEL - DROBS4 SUBROUTINE

ISN

21 20 INDEX=INDEX+NFLDS(LVEHIC)
22 30 INDEX=INDEX+IATTR
23 PATTR=POBATT(INDEX)
24 IF(PATTR.LE.0.0) RETURN
C
C ... GET PRIORITY ASSOCIATED WITH OBJECT
C
25 GO TO(40,50,60),IFEA
C
C ... OBJECT IS VEHICLE
C
26 40 P=PATTR*PCBVEM(ISIDE,IVAL(JOWNER,IOBSRF))
27 RETURN
C
C ... OBJECT IS PERSON
C
28 50 IACTRF=IFIRST(IVAL(JACTIV,IOBSRF),LPTR)
29 IF(IACTRF.EQ.NULL) RETURN
30 IACT=IVAL(JTYPE,IACTRF)
31 IALG=IVAL(JALLEG,IOBSRF)
32 P=PATTR*POBPSN(ISIDE,IACT,IALG)
33 RETURN
C
C ... OBJECT IS SENSOR
C
34 60 P=PATTR*POBSEN(ISIDE)
35 RETURN
36 END

FSN MODEL - EFFTIM FUNCTION

ISN

1

FUNCTION EFFTIM(IPLARF,ISOU,ISIN,IACSUB)

C

C

C -- THE PURPOSE OF THIS FUNCTION IS TO
C 1. EVALUATE A LINK ACCORDING TO THE PERCIEVED
C DANGER AND ADJUST THE TRAVEL TIME.

C

C -- INPUT PARAMTERS

C

C IPLARF -- PLAYER REFERENCE
C ISOU - SOURCE NODE OF LINK
C ISIN -- SINK NODE OF LINK
C IACTSUB -- ACTIVITY SUB-TYPE

C

C -- OUTPUT

C

C THIS FUNCTION RETURNS THE EFFECTIVE TIME NEEDED
C TO TRAVERSE A GIVEN LINK

C

C -- PROCEDURE

C

C 1. DETERMINE THE TRAVEL TIME FROM ISOU TO ISIN

C

C 2. CALCULATE THE PENALTY

C PENALTY --- THE SUM OF PERCIEVED DANGERS

C

C 3. EFFTIM= TRAVEL TIME* (1+PENALTY)

C

2

COMMON /STATEV/

3

DIMENSION ITEM(41900),DITEM(41900)

4

EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

5

COMMON /PARS/

6

EQUIVALENCE (NULL,NULL), (IFAIL,FAIL)

FSN MODEL - EFPTIM FUNCTION

```
1SN
 7      REAL*8 DTPNAM,FLDNAM,FORMOT
 8      COMMON /PARS1/
 9      COMMON /PARS2/
10      COMMON /PARS3/
11      COMMON /DATAV/
12      DIMENSION ACTTIM(25)
13      EQUIVALENCE (ACTTIM(1),ACTRAT(1))

C
C -- FIND TRAVEL TIME
C
14      TIM=TRAVEL(IPLARF,ISOU,ISIN,IACSUB,ISUM)
C
C -- CALCULATE PENALTY
C
C       IF PERCIEVED THREATS LOCATION HAS LINE OF
C           SIGHT WITH ISIN THEN ADD DANGER FACTOR
C           ADJUST DANGER FACTOR FOR EACH THREAT BY DANGER
C           OF WEAPON PERCIEVED TO BE CARRYING
C
15      IALL=IVAL(JALLEG,IPLARF)
16      IPERLT=IVAL(JPRCPS,IPLARF)
17      DNGER=0.0
18      IPERRF=IFIRST(IPERLT,KA)
19      IF(IPERRF .EQ. NULL) GO TO 50

C
C -- FIND REFERENCE TO PERCEPTION INFORMATION
C
20      IVRF=IVAL(JVIEW,IPERRF)
21      CALL PARSRF(IVRF,IDX,IDUM, IDUM)
22      IF(IDY .EQ. LFORCE) GO TO 30

C
C -- SET SECRECY PLAN
C
23      ISEC=ISECPL(IALL)
C
C -- DETERMINE DETECTOR TYPE
```

FSN MODEL - EFPTIM FUNCTION

1SN
C
24 IF (IDY .EQ. LSENS) GO TO 20
25 IF (IDY .EQ. LPERSN) GO TO 7
C
C -- FIND ALLEGIANCE OF PERCEPTION
C
26 IF (IVAL (JOWNER, IVRF) .EQ. NULL) GO TO 30
27 IF (IVAL (JOWNER, IVRF) .EQ. IALL) GO TO 30
28 GO TO 6
29 7 IF (IVAL (JALLEG, IVRF) .EQ. NULL) GO TO 30
30 IF (IVAL (JALLEG, IVRF) .EQ. IALL) GO TO 30
C
C -- DETERMINE LOCATION
C
31 6 IF (IVAL (JLOCN, IVRF) .EQ. NULL) GO TO 30
32 IVLOC=IVAL (JLOCN, IVRF)
C
C -- CHECK FOR LINE OF SIGHT
C
33 ICOD=LOS (ISIN, IVLOC)
34 IF (ICOD .EQ. 0) GO TO 30
35 IF (IDY .EQ. LVEHIC) GO TO 15
C
C -- FIND MOST DANGEROUS WEAPON FOR PERSON
C
36 IDETT=KDETPN
37 IVET=NVEHTP+NPSNTP
38 IFELD=NWEPTP+1
39 DNG=DNGWEP (IVET, IFELD)
40 DT=DIST (ISIN, IVRF)
41 IWPLT=IVAL (JWEAPS, IVRF)
42 IWPN=IFIRST (IWPLT, KC)
43 8 IF (IWPN .EQ. NULL) GO TO 12
C
C -- IGNORE EQUIPMENT
C

FSN MODEL - EFPTIM FUNCTION

158
44 CALL PARSRF(IWPN, IDY, IDUM, IDUM)
45 IF (IDY .NE. LWEAP) GO TO 10
46 IWPNTY=IVAL(JTYPE, IWPN)
47 IF (DT .GT. RNGMAX(IWPNTY)) GO TO 10
48 IF (DNGWEP(IVET, IWPNTY) .LE. DNG) GO TO 10
49 DNG=DNGWEP(IVET, IWPNTY)
50 10 IWPN=NEXT(IWPLT, KC)
51 GO TO 8
C
C -- ADD TO DANGER
C
52 12 DNGER=DNGER+DNGDET(IDETT, ISEC)*DNG
53 GO TO 30
C
C -- FIND MOST DANGEROUS WEAPON ON VEHICLE
C
54 15 IDETT=KDETVE
55 IVET=IVAL(JTYPE, IVRF)
56 IFELD=KNVGAS+1
57 DNG=DNGWEP(IVET, IFELD)
58 IWLT=IVAL(JCONTS, KIVRF)
59 IWPN=IFIRST(IWLT, KD)
60 16 IF (IWPN .EQ. NULL) GO TO 12
C
C -- IGNORE PERSON AND EQUIPMENT
C
61 CALL PARSRF(IWPN, IDY, IDUM, IDUM)
62 IF (IDY .NE. LWEAP) GO TO 18
63 IWPNTY=IVAL(JTYPE, IWPN)
64 IF (DNGWEP(IVET, IWPNTY) .LE. DNG) GO TO 18
65 DNG=DNGWEP(IVET, IWPNTY)
66 18 IWPN=NEXT(IWLT, KD)
67 GO TO 16
C
C -- SENSOR DETECTION
C

FSN MODEL - EFFTIM FUNCTION

```
ISN          C ... SENSORS SHOULD PROBABLY BE IGNORED FOR SFE MEMBERS...
68          20 IF(IALL .EQ. KSFE) GO TO 30
69          ITEMP=IVAL(JTYPE,IVRF)
70          IF(ITEMP .EQ. NULL) GO TO 30
71          IDLTT=ITEMP+KDETPN
C
C ... SEE IF IN COVERAGE FIELD
C
72          ISNREF = IVAL(JID,IPERRF)
73          ICOVL  = IVAL(JCOVRC,ISNREF)
74          ION = LSERCH(ICOVL,ISIN,NULL,NULL,NULL)
75          IF(ION .EQ. NULL) GO TO 30
76          DNGER=DNGER+DNGDET(IDETT,ISEC)
C
77          30 IPERRF=NEXT(IPERLT,KA)
78          GO TO 5
C
C
C
79          50 EFFTIM=TIM*(1.0+DNGER)
C
80          RETURN
81          END
```

FSN MODEL - ESCAPE SUBROUTINE

15N
1 SUBROUTINE ESCAPE(IEXITS,LEADER,IFORCE)
C
C ROUTINE FOR THE LEADER TO SEND INSTRUCTIONS TO MEMBERS OF HIS
C FORCE TO ESCAPE FROM THE SITE.
C
C INPUT PARAMETERS:
C IEXITS A NODE OR LIST OF POSSIBLE NODES AT WHICH TO EXIT FROM
C THE SITE. IF A LIST IS PASSED, THE PLAYER IS TO CHOOSE
C HIS OWN EXIT POINT FROM THE ALTERNATIVES SPECIFIED.
C LEADER A REFERENCE TO THE LEADER GIVING ORDERS
C IFORCE A LIST OF PERSONS WHOM THE LEADER WISHES TO INSTRUCT
C TO EXIT THE SITE.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,PLDNAM,FORMAT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
C
C ITERATE OVER PLAYERS OF INTEREST
11 IPERSN = IFIRST(IFORCE,KP)
12 10 IF (IPERSN .EQ. NULL) RETURN
C GET FIRST POSSIBLE EXIT LOCATION
13 IEXIT = IFIRST(IEXITS,KE)
14 IF (IEXIT .EQ. NULL) RETURN
C BRANCH IF THERE IS JUST ONE EXIT SPECIFIED
15 IF (LISTND(JNXT,KE) .LE. 0) GO TO 20
C OTHERWISE SELECT THE NEAREST EXIT TO THE PLAYER
16 DISTMN = 99999.
17 15 D = DIST(IPERSN,IEXIT)
18 IF (D .GT. DISTMN) GO TO 18

FSN MODEL - ESCAPE SUBROUTINE

```
15N  
19          DISTMN = D  
20          IX = IEXIT  
21 18      C TRY NEXT EXIT  
21 18      IEXIT = NEXT(IEXITS,KE)  
22          IF (IEXIT .NE. NULL) GO TO 15  
23          IEXIT = IX  
24 20      C INSTRUCT PERSON TO MOVE TO EXIT  
24 20      IACT = NEWRC6(LACTN,KMOVNG,IEXIT,NULL,NULL,NULL,IPERSN)  
25          IMSG = NEWRC5(LMESS,LEADER,IPERSN,JPLANS,TMIN,IACT)  
26          CALL COMMO(LEADER,IPERSN,IMSG)  
27 20      C GET NEXT PLAYER  
27 20      IPERSN = NEXT(IFORCE,KP)  
28          GO TO 10  
29          END
```

FSN MODEL - FASMNT SUBROUTINE

ISN

```
1      SUBROUTINE FASMNT(IPLARF,ITRG,IPHS,IRF)
C
C   -- THE PURPOSE OF THE SUBROUTINE IS TO :
C       1. CREATE AN ASSESSMENT OF THE TARGETS
C          PHYSICAL STATUS
C
C   -- INPUT PARAMETERS
C
C
C       IPLARF -- PLAYER REFERENCE
C       ITRG --- TARGET REFERENCE
C       IPHS -- RESULTS OF FIRING
C
C
C   -- OUTPUT VARIABLES
C
C       IRF - REFERENCE TO MESSAGE RECORD CREATED
C
C   -- PROCEDURE
C
C       THE RESULTS OF FIRING ARE STORED IN
C       MESSAGE FORMAT AND ARE ADDED
C       TO THE LIST LASMNT
C
C
2     COMMON /STATEV/
3     DIMENSION ITEM(41900),DITEM(41900)
4     EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5     COMMON /PARS/
6     EQUIVALENCE (PNULL,NULL),(IFAIL,FAIL)
7     REAL*8 DTPNAM,FLDNAM,FORMAT
8     COMMON /PARS1/
9     COMMON /PARS3/
10    COMMON /DATAV/
11    DIMENSION ACTTIM(25)
```

PSN MODEL - FASMNT SUBROUTINE

```
1SN  
12      EQUIVALENCE (ACTTIM(1),ACTRAT(1))  
13      COMMON /RECREF/  
14      DIMENSION RECRPQ(140)  
15      EQUIVALENCE (RECRPQ(1),IGOALS)  
C  
C      CALL HUMAN FACTORS  
C      CALL HFAC3  
C  
C      -- TEST IF RESULTS OF FIRE WAS A MISS  
C  
16      IF (IPHS .EQ. 0) IPHS=IVAL(JPSTAT,ITRG)  
C  
C      -- DETERMINE IF IPLARF HAS MADE AN ASSESSMENT OF ITRG  
C  
17      IRF=IFIRST(LASMNT,KA)  
18      5      IF(IRF .EQ. NULL) GO TO 30  
C  
19      IF (IVAL(JSOURC,IRF) .NE. IPLARF) GO TO 20  
20      IF (IVAL(JSUBJ,IRF) .NE. ITRG) GO TO 20  
C  
C      --MESSAGE EXISTS  
C      TEST IF CONTENTS DIFFERENT  
C  
21      IF (IVAL(JCONT,IRF) .EQ. IPHS) GO TO 50  
C  
C      -- ADJUST CONTENTS  
C  
22      IPTR=IREF1(IRF)  
23      ITEM(JCONT+IPTR)=IPHS  
24      GO TO 50  
C  
C      -- NEXT ASSESSMENT  
C  
25      20 IRF=NEXT(LASMNT,KA)  
26      GO TO 5  
C
```

FSN MODEL - FASMNT SUBROUTINE

ISN

```
C-- NEW ASSESSMENT
C -- GENERATE NEW MESSAGE
C
27      30 TT1M = TMIN + (DTSEC - VAL(JTREQ,IPLARF)) / 60.
28          IRF = NEWRC5(LMESS,IPLARF,ITRG,JPSTAT,TTIM,IPHS)
C
C -- ADD RECORD TO LASMNT
C
29      CALL CHGLST(IRF,1,7,0)
30      LASMNT=ISTACK(IRF,LASMNT)
C
31      50 CONTINUE
32          RETURN
33      END
```

FSN MODEL - FDESIR SUBROUTINE

ISN

1

SUBROUTINE FDESIR(IPLARF,ITRG,ICODE,TMAX,FTHRES)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

- C 1. EVALUATE THE DESIRABILITY FOR A PLAYER TO
C FIRE AT A TARGET.
C 2. IF MORE THAN ONE POTENTIAL TARGET, EVALUATE
C ALL MEMBERS OF THE LIST AND RETURN THE TARGET
C WITH THE LARGEST DESIRABILITY.

C

C

C -- INPUT PARAMETERS

C

C IPLARF -- PLAYER REFERENCE FOR PLAYER TO FIRE

C

C

C ITRG -- PLAYER REFERENCE OR LIST REFERENCE
C OF POTENTIAL TARGETS

C

C

C -- OUTPUT VARIABLES

C

C

C ICODE -- PLAYER REFERENCE, TARGET SELECTED AS MOST
C DESIRABLE.

C

C

C TMAX -- VALUE OF DESIRABILITY TO BE COMPARED TO FIRE
C THRESHOLD.

C

C

C -- PROCEDURE

C

C

C 1. SEARCH TARGET LIST

C 2. IDENTIFY EACH CASE SUBSET THAT IS APPROPRIATE

C 3. AND, ADD THE VALUE TO THE DESIRABILITY

FSN MODEL - FDESIR SUBROUTINE

ISN

C

```
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /DATAV/
12     DIMENSION ACTTIM(25)
13     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14     COMMON /RECREF/
15     DIMENSION RECRFQ(140)
16     EQUIVALENCE (RECRFQ(1),IGOALS)
17     COMMON /NEW/
```

C

C

```
18    TDMAX=0.
19    IDCOD=NULL
20    ICODE=NULL
21    TMAX=0.
22    IF (EVALN(30) .LE. 0) EVALN(30)=1.0
```

C

```
C -- TEST FOR RESTRICTED FIRING
```

C

```
23    IF (IFCOND(ISIDE) .GE. 3) GO TO 7
```

C

```
C -- TEST FOR DO NOT FIRE
```

C

```
24    IF (IFCOND(ISIDE) .LT. 2) GO TO 120
```

C

```
C -- TEST FOR FIRE ONLY WHEN FIRED UPON
```

C

```
25    CALL DETFIR(IPLARF,ITTR)
```

FSN MODEL - FDESIR SUBROUTINE

1SN

26 IF(ITTR .EQ. NULL) GO TO 120

C

C -- SEARCH TARGET LIST

C

27 ITAR=IFIRST(ITRG,KA)

28 IF(ITAR .EQ. NULL) GO TO 120

C

C -- TEST PHYSICAL STATUS OF TARGET

C

29 IF(IVAL(JPSTAT,ITAR) .LE. KCAPTR) GO TO 110

C

C -- DETERMINE IF TARGET WITHIN RANGE OF ANY WEAPON

C

30 CALL NWPNNS(IPLARF,ITAR,IWPNRF,IWPN)

31 IF(IWPN .EQ. 0) GO TO 110

C

C -- SEARCH PERCEPTIONS TO FIND REFERENCE TO TARGET

C

32 IPERLT=IVAL(JPRCPS,IPLARF)

33 IPER=IFIRST(IPERLT,KB)

34 IF(IPER .EQ. NULL) GO TO 110

35 IF(IVAL(JID,IPER) .EQ. ITAB) GO TO 50

36 IPERH=NEXT(IPERLT,KB)

37 GO TO 30

C

38 IPERRF=IVAL(JVIEW,IPER)

39 CALL PARSRF(IPERRF, IDYY, IRRN, IDDUM)

40 IF(IDYY .EQ. LFORCE) GO TO 35

41 TEMP=0.0

C

C -- CASE SUBSET 1

C TARGET FIRING DIRECTION

C

C FIND PERCIEVED ACTIVITY OF TARGET

C

42 IACRF=IVAL(JACTIV,IPERRF)

FSN MODEL - FDESIR SUBROUTINE

1 SN
43 IACT=IFIRST(IACRF,KC)
44 IF(IACT .EQ. NULL) GO TO 57
45 IF(IVAL(JTYPE,IACT) .NE. KFIRNG) GO TO 55
46 IF(IVAL(JPAK1,IACT) .NE. IPLARF) GO TO 53
C
C PERCIEVED TO BE TARGET
C
47 TEMP=TEMP+EVALN(21)
48 GO TO 60
C
C FIRING BUT NOT TARGET
C
49 53 TEMP=TEMP+EVALN(22)
50 GO TO 65
C
C NOT FIRING
C
51 55 TEMP=TEMP+EVALN(23)
52 GO TO 60
C
C NO ACTIVITY PERCIEVED
C
53 57 TEMP=TEMP+EVALN(23)
54 GO TO 65
C
C -- CASE SUBSET 2
C TARGET PERCIEVED TO BE MOVING
C
C TEST ACTIVITY FOR MOVING
C
55 60 IF(IVAL(JTYPE,IACT) .NE. KMVNG) GO TO 65
56 TEMP=TEMP+EVALN(25)
57 GO TO 70
C
C NOT MOVING
C

PSN MODEL - FDESIR SUBROUTINE

1SN
58 65 TEMP=TEMP+EVALN(24)
C
C -- CASE SUBSET 3
C COVER OF PLAYER
C
59 70 CALL DIREC{ITAK,IPLARF,THE1,PHE1)
60 IF(COVR(IPLARF,THE1) .LE. 0.1) GO TO 75
61 TEMP=TEMP+EVALN(26)
62 GO TO 80
C
C NO COVER AVAILABLE
C
63 75 TEMP=TEMP+EVALN(27)
C
C -- CASE SUBSET 4
C COVER OF TARGET
C
64 80 CALL DIREC{IPLARF,ITAR,THE2,PHE2)
65 IF(COVR(ITAR,THE2) .LE. 0.1) GO TO 85
C
C COVER AVAILABLE
C
66 TEMP=TEMP+EVALN(31)
67 GO TO 90
C
C NO COVER
C
68 85 TEMP=TEMP+EVALN(32)
C
C -- CASE SUBSET 5
C SENSOR TARGET
C
69 90 CALL PARSRF(ITAR, IDY, IRR, IDU)
70 IF(IDY .EQ. LSENS) GO TO 95
C
C NOT A SENSOR

FSN MODEL - FDESIR SUBROUTINE

ISN
C
71 TEMP=TEMP+EVALN(29)
72 GO TO 100
C
C SENSOR
C
73 95 TEMP=TEMP+EVALN(28)
C
C
74 100 IF(TEMP/EVALN(30) .LE. TMAX) GO TO 110
C
C TARGET DESIRABLE
C TEST IF ANYONE ELSE FIRING AT THE TARGET
C
75 JL=JGARDS
76 IF(ISIDE .EQ. KAFE) JL=JADVRS
77 IPLT=IVAL(JL,ISITE)
C
78 ICOLE=IFIRST(IPLT,KB)
79 101 IF(ICOLE .EQ. IPLARF) GO TO 105
C
C TEST PLANS OF PLAYER
C
80 IPLNS=IVAL(JPLANS,ICOLE)
81 IPLN=IFIRST(IPLNS,KZ)
82 IF(IPLN .EQ. NULL) GO TO 102
83 IF(IVAL(JTYPE,IPLN) .NE. KFIRNG) GO TO 102
C
C TEST TARGET OF FIRING PLAN
C
84 IF(IVAL(JPAR1,IPLN) .NE. ITAR) GO TO 102
C
C FRIENDLY HAS DECIDED TO FIRE AT TARGET
C SAVE FOR LATTER PROCESSING
C
85 IF(TEMP/EVALN(30) .LE. TDMAX) GO TO 102

FSN MODEL - FDESIR SUBROUTINE

1SN
86 TDMAX=TEMP/EVALN(30)
87 IDCOD=ITAR
88 GO TO 110
 C
89 102 ICOLE=NEXT(IPLT,KB)
90 GO TO 101
 C
91 105 THAX=TEMP/EVALN(30)
92 ICODE=ITAR
 C
93 110 ITAR=NEXT(ITRG,KA)
94 GO TO 10
 C
 C TEST IF TARGET FOUND
 C
95 120 IF(ICODE .EQ. NULL) GO TO 121
 C
 C TEST IF TARGET DESIRABLE
 C
96 IF(TMAX .GE. FTHRES) GO TO 125
 C
 C TARGET NOT FOUND
 C
97 121 IF(IDCOD .EQ. NULL) GO TO 125
98 THAX=TDMAX
99 ICODE=IDCOD
 C
 C TARGET FOUND
 C
100 125 IF(TMAX .GT. 1.0) TMAX=1.0
101 RETURN
102 END

FSN MODEL - GENOBS SUBROUTINE

15N

1

SUBROUTINE GENOBS

C ... THIS SUBROUTINE IS CALLED TO COMPARE A PERSON'S PERCEPTION
C OF AN OBJECT WITH A DESCRIPTION OF AN OBJECT CONTAINING ONLY
C THOSE ATTRIBUTES HE COULD OBSERVE. IF A DIFFERENCE EXISTS, OR
C THE PERSON HAS NO PERCEPTION OF THE OBJECT, A MESSAGE IS CREATED
C FOR EACH ATTRIBUTE THAT IS DIFFERENT. THE MESSAGES AND THEIR
C ASSOCIATED OBSERVATION PRIORITIES ARE ADDED TO THE ARRAYS
C IMMSG AND PMMSG. LMAX DETERMINES HOW MANY MESSAGES CAN BE
C GENERATED BY A PARTICULAR PLAYER

C

C ... INPUT PARAMETERS: (IN BLANK COMMON)

C IPCPRF ... REFERENCE TO PERCEPTION RECORD, OR NULL
C JOBSRF ... REFERENCE TO PERSON'S VIEW OF OBJECT
C IDUMRF ... REFERENCE TO PSEUDO OBJECT DESCRIPTION
C IPLARF ... REFERENCE TO OBSERVERS DESCRIPTION RECORD
C IOBSRF ... REFERENCE TO TRUE OBJECT DESCRIPTION

C

C ... OUTPUT PARAMETERS: (IN BLANK COMMON)

C IMMSG ... ARRAY OF MESSAGE POINTERS
C PMMSG ... ARRAY OF ASSOCIATED PRIORITY VALUES
C NMSG ... NUMBER OF ENTRIES IN IMMSG
C LOWMSG .. INDEX OF LOWEST PRIORITY MESSAGE IN IMMSG
C PMIN ... PRIORITY OF LOWEST PRIORITY MSG IN IMMSG

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PAHS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 COMMON /DATAV/
11 DIMENSION ACTTIM(25)
12 EQUIVALENCE (ACTTIM(1),ACTRAT(1))

FSN MODEL - GENOBS SUBROUTINE

```
1SN  
13      COMMON /RECREP/  
14      DIMENSION RECREP(140)  
15      EQUIVALENCE (RECREP(1),IGOALS)  
16      COMMON // LMAX,PMIN,IMSG(50),NMSG,POBSV,LOWMSG  
+      IPLARF,IOBSRF,JOBSRF,IDUMRF,IASMNT(19),IPCPRF  
17      DIMENSION INSTAT(2,5)  
18      EQUIVALENCE (INSTAT(1,1),IOBSV(41))  
19      KNSDR = 4  
20      NINSDR = 5  
C  
C ... UNLESS OBJECT IS VEHICLE, PERSON OR SENSOR CREATE  
C     NO OBSERVATIONS  
C  
21      CALL PARSRF(IDUMRF, IDTYP,IRECNO,IFLD)  
22      IF(IDTYP.EQ.LVEHIC) GO TO 100  
23      IF(IDTYP.EQ.LPERSON) GO TO 200  
24      IF(IDTYP.EQ.LSENS) GO TO 300  
25      RETURN  
C  
C ... SET UP TO PROCESS VEHICLE  
C  
26 100  NFEPTR=0  
27  NSUBR=1  
28  GO TO 400  
C  
C ... SET UP TO PROCESS PERSON  
C  
29 200  NFEPTR=NFLDS(LVEHIC)  
30  NSUBR=2  
C  
C ... MAKE CHECK FOR SFE OBSERVING INSIDER  
C  
31  IF(IVAL(JALLEG,IPLARF).NE. KSFE) GO TO 400  
32  IF(IVAL(JTYPE,IOBSRF).NE. KNSDR) GO TO 400  
C  
C ... SFE OBSERVING INSIDER, FIND INSIDER NUMBER  
C
```

FSN MODEL - GENOBS SUBROUTINE

ISN
33 DO 210 I=1,NINSDR
34 IF (INSTAT(1,I) .NE. IOBSRF) GO TO 210
35 J=I
36 GO TO 220
37 210 CONTINUE
38 GO TO 400
C
C ... IF INSIDER HAS NOT PERFORMED TRIGGER ACTION THIS
C CYCLE HE IS OBSERVED AS SFE
C NEGATIVE TRIGGER MEANS ALREADY OBSERVED
C
39 220 IF (INSTAT(2,J)) 400,230,240
C
C ... NO TRIGGER ACTION
C
40 230 IPTR = IREF1(IDUMRF)
41 ITEM(IPTR + JALLEG) = KSFE
42 GO TO 400
C
C ... NOT PREVIOUSLY SEEN, MARK AS SEEN
C
43 240 INSTAT(2,J) = -INSTAT(2,J)
44 GO TO 400
C
C ... SET UP TO PROCESS SENSOR
C
45 300 NFEPTR=NFLDS(LVEHIC) + NFLDS(LPERSN)
46 NSUBR=3
C
C ... SET UP POINTERS
C
47 400 NF=NFLDS(IDTYP)
48 NBASE=IREF1(NEWREF(IDTYP,NRECS(IDTYP),0))
49 IPTR=IREF1(IDUMRF)
50 IF (IPCPRF.NE.NULL) JPTR=IREF1(JOBSRF)
C

FSN MODEL - GENOBS SUBROUTINE

1SN
C ... START FIELD COMPARISON LOOP
C
51 DO 500 IFLD=1,NF
C
C ... SKIP FIELD IF NO INFO IN IDUMRF
C
52 IF(ITEM(IPTR+IFLD).EQ.NULL) GO TO 500
C
C ... SKIP FIELD IF PRIORITY IS 0.0
C
53 P=POBATT(NFEPTR+IFLD)*POBSV
54 IF(P.LE.0.0) GO TO 500
C
C ... IF LIST FULL AND P TOO LOW, IGNORE FIELD
C
55 IF(NMSG.EQ.LMAX. AND. P.LE.PMIN) GO TO 500
C
C ... IF PLAYER HAS NO PERCEPTION, GO TO CREATE OBS
C
56 ITYP=ITEM(NBASE+IFLD)
57 IF(IPCPRF.EQ.NULL) GO TO 450
58 IF(ITYP.GT.2) GO TO 420
C
C ... ATTRIBUTE IS SCALAR, SKIP FIELD IF EQUAL
C
59 IF(ITEM(IPTR+IFLD).EQ.ITEM(JPTR+IFLD)) GO TO 500
60 GO TO 450
C
C ... ATTRIBUTE IS REFERENCE, CALL APPROPRIATE
FUNCTION AND SKIP FIELD IF EQUAL
C
61 420 GO TO (421,422,423),NSUBR
C
C ... OBJECT IS VEHICLE
C
62 421 IF(ICMPVE(IFLD,JPTR,IPTR) .EQ. IFAIL) GO TO 450

FSN MODEL - GENOBS SUBROUTINE

150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175

ISN
63 GO TO 500
C
C ... OBJECT IS PERSON
C
64 422 IF (ICMPPN(IFLD,JPTR,IPTR).EQ.IFAIL) GO TO 450
65 GO TO 500
C
C ... OBJECT IS SENSOR
C
66 423 IF (ICMPSN(IFLD,JPTR,IPTR).EQ.IFAIL) GO TO 450
67 GO TO 500
C
C ... NEW INFORMATION, CREATE A MESSAGE
C
68 450 IF (ITYP.LE. 2) ICONT = IVAL(IFLD, IDUMRF)
69 IF (ITYP.GT. 2) ICONT = ICOPY(ITEM(IPTR+IFLD))
70 IMGRF = NEWRC5(LMESS, IPLARF, IOBSRF, IFLD, TMIN, ICONT)
C
C ... MERGE IMGRF INTO IMSG
C
71 470 CALL ADDVAL(IMGRF,P,IMSG,PMSG,NMSG,LOWMSG,PMIN,LMAX)
C
C ... STACK IMGRF ON LOCAL(2) TO PROTECT FROM GARBAGE COL FCFOR
C
72 LOCAL(2) = ISTACK(IMGRF,LOCAL(2))
C
C ... END COMPARISON LOOP
C
73 500 CONTINUE
74 RETURN
75 END

FSN MODEL - ICANOB FUNCTION

1SN

```
1      FUNCTION ICANOB(IPLARF)
C ... FUNCTION CALLED BY COMOBS TO DETERMINE IF PLAYER
C IS CAPABLE OF MAKING AND REPORTING OBSERVATIONS.  VALUE
C OF NULL IS RETURNED IF PLAYER CAN NOT OBSERVE, IOK
C OTHERWISE
C INPUT PARAMETER:
C           IPLARF ... RECORD REFERENCE FOR PLAYER DESCRIPTION
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /DATAV/
11     DIMENSION ACTTIM(25)
12     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
13     ICANOB=NULL
14     IF(VAL(JPSTAT,IPLARF).LE.KCAPTR) RETURN
15     IF(VAL(JSUPRN,IPLARF).GE.RESOBS(IPLAYR)) RETURN
16     ICANOB=IOK
17     RETURN
18     END
```

FSN MODEL - ICHOOS FUNCTION

```
1      FUNCTION ICHOOS(NOPT, PROBS, ISFED)
C
C      ROUTINE TO MAKE A RANDOM CHOICE AMONGST N ALTERNATIVES BASED ON
C      A DISTRIBUTION OF PROBABILITIES FOR CHOOSING EACH OF THE OPTIONS.
C      FUNCTION RETURNS AN INDEX OF THE CHOICE SELECTED.
C
C      INPUT PARAMETERS:
C          NOPT      NUMBER OF OPTIONS
C          PROBS     A VECTOR OF NOPT PROBABILITIES WHICH TOGETHER SUM TO 1.0.
C          ISEED     RANDOM NUMBER SEED
C
2      DIMENSION PROBS(NOPT)
C
C      GET A RANDOM NUMBER BETWEEN 0.0 AND 1.0.
3      RN = URAND(ISEED)
4      P = 0.
C      TEST WHAT INTERVAL THE RANDOM NUMBER IS IN
5      DO 10 ICHOOS=1,NOPT
6          P = P + PROBS(ICHOOS)
7          IF (RN .LE. P) RETURN
8      10    CONTINUE
9      ICHOOS = NOPT
10     RETURN
11     END
```

FSN MODEL - ICMPPN FUNCTION

ISM

```
1      FUNCTION ICMPPN(IFLD,IPTR,JPTR)
C ... FUNCTION CALLED BY GENOBS TO COMPARE NON-SCALAR FIELDS
C     IN PERSON DESCRIPTIONS.  THE VALUE RETURNED IS NULL IF AN
C     INVALID FIELD IS REFERENCED, IOK IF THE PERCEIVED FIELD
C     VALUE IS EQUIVALENT TO THE ACTUAL VALUE, AND IFAIL IF
C     THE FIELDS ARE NOT EQUIVALENT
C
C ... INPUT PARAMETERS:
C           IFLD ... FIELD NUMBER TO BE COMPARED
C           IPTR ... ZERO OFFSET POINTER TO "VIEW" RECORD
C           JPTR ... ZERO OFFSET POINTER TO DESCRIPTION RECORD
C
2     COMMON /STATEV/
3     DIMENSION ITEM(41900),DITEM(41900)
4     EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5     COMMON /PARS/
6     EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7     REAL*3 DTPNAM,FLDNAM,FORMOT
8     COMMON /PARS1/
C
9     ICMPPN=NULL
10    IF(IFLD.EQ.JLOCN) GO TO 100
11    IF(IFLD.EQ.JWEAPS) GO TO 200
12    IF(IFLD.EQ.JACTIV) GO TO 300
13    RETURN
C
C ... PROCESS LOCATION FIELD
C
14   100  LOCR1=ITEM(IFLD+IPTR)
15   LOCR2=ITEM(IFLD+JPTR)
16   ICMPPN=IFAIL
17   IF(LOCR1.EQ.NULL) RETURN
C
C ... ONLY THE PLACE FIELD HAS TO AGREE
C
```

FSN MODEL - ICMPPN FUNCTION

```
18      IF (IVAL(JPLACE,LOCRF1).EQ.IVAL(JPLACE,LOCRF2)) ICMPPN=IOK
19      RETURN
20      200  ICMPPN=IFAIL
21      LWPRF2=ITEM(IFLD+JPTR)
22      JTYP=IVAL(JTYPE,LWPRF2)
23      CALL PARSRF(LWPRF2,JDTYP,IDUM1,IDUM2)
24      LWPLIS=ITEM(IFLD+IPTR)
25      LWPRF1=IFIRST(LWPLIS,LPTR)
26      210  IF (LWPRF1.EQ.NULL) RETURN
27      CALL PARSRF(LWPRF1,ITDYP, IDUM1, IDUM2)
28      ITYP=IVAL(JTYPE,LWPRF1)
29      IF (ITYP.NE.JTYP .OR. IDTYP.NE.JDTYP) GO TO 220
30      ICMPPN=IOK
31      RETURN
32      220  LWPRF1=NEXT(LWPLIS,LPTR)
33      GO TO 210
34      300  ICMPPN=IFAIL
35      JACTRF=ITEM(IFLD+JPTR)
36      JACTYP=IVAL(JTYPE,JACTRF)
37      LACLIS=ITEM(IFLD+IPTR)
38      IACTRF=IFIRST(LACLIS,LPTR)
39      .
40      310  ICMPPN=IFAIL
41      CALL PARSRF(LACTRF,ITDYP, IDUM1, IDUM2)
42      ITYP=IVAL(JTYPE,IACTRF)
43      IF (ITYP.NE.JTYP .OR. IDTYP.NE.JDTYP) GO TO 440
44      ICMPPN=IOK
45      RETURN
46      440  ICMPPN=IFAIL
47      END
```

FSN MODEL - ICMPFN FUNCTION

ISN
39 310 IF (IACTR.F.EQ.NULL) RETURN
40 IF (JACTYP.NE.IVAL(JTYPE,IACTR)) GO TO 320
41 ICMPFN=IOK
42 RETURN
43 320 IACTRF=NEXT(LACLIS,LPTR)
44 GO TO 310
45 END

FSN MODEL - ICMPSEN FUNCTION

ISN

```
1      FUNCTION ICMPSEN(IFLD,IPTR,JPTR)
C ... FUNCTION CALLED BY GENOBS TO COMPARE NON-SCALAR FIELDS
C IN SENSOR DESCRIPTIONS. THE VALUE RETURNED IS NULL IF AN
C INVALID FIELD IS REFERENCED, IOK IF THE PERCEIVED FIELD
C VALUE IS EQUIVALENT TO THE ACTUAL VALUE, AND IFAIL IF
C THE FIELDS ARE NOT EQUIVALENT
C
C ... INPUT PARAMETERS:
C         IFLD ... FIELD NUMBER TO BE COMPARED
C         IPTR ... ZERO OFFSET POINTER TO "VIEW" RECORD
C         JPTR ... ZERO OFFSET POINTER TO DESCRIPTION RECORD
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITLM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMAT
8      COMMON /PARS1/
C
9      ICMSN=NULL
10     IF(IFLD.EQ.JLOCN) GO TO 100
11     IF(IFLD.EQ.JCOVRG) GO TO 200
12     IF(IFLD.EQ.JINCOM) GO TO 200
13     RETURN
C
C ... PROCESS LOCATION FIELD
C
14    100   LOCRF1=ITEM(IFLD+IPTR)
15    LOCRF2=ITEM(IFLD+JPTR)
16    ICMSN=IFAIL
17    IF(LOCRF1.EQ.NULL) RETURN
C
C ... ONLY THE PLACE FIELD HAS TO AGREE
C
```

PSN MODEL - ICMPSON FUNCTION

```
154  
18      IF ({IVAL(JPLACE,LOCRF1) .EQ. IVAL(JPLACE,LOCRF2)}) ICMPSON=IOK  
19      RETURN  
C  
C ... PROCESS COVERAGE OR "IN COMM0 WITH" FIELD  
C  
20      200 ICMPSON=FAIL  
21      JCOVRF=ITEM(IFLD+JPTR)  
22      ICOLIS=ITEM(IFLD+IPTR)  
23      ICOVRF=IFIRST(ICOLIS,LPTR)  
24      210 IF (ICOVRF.EQ.NULL) RETURN  
25      IF (ICOVRF.NE.JCOVRF) GO TO 220  
26      ICMPSON=IOK  
27      RETURN  
28      220 ICOVRF=NEXT(ICOLIS,LPTR)  
29      GO TO 210  
30      END
```

FSN MODEL - ICMPVE FUNCTION

ISM

```
1      FUNCTION ICMPVE(IFLD,IPTR,JPTR)
C ... FUNCTION CALLED BY GENOBS TO COMPARE NON-SCALAR FIELDS
C     IN VEHICLE DESCRIPTIONS. THE VALUE RETURNED IS NULL IF AN
C     INVALID FIELD IS REFERENCED, IOK IF THE PERCEIVED FIELD
C     VALUE IS EQUIVALENT TO THE ACTUAL VALUE, AND IFAIL IF
C     THE FIELDS ARE NOT EQUIVALENT
C
C ... INPUT PARAMETERS:
C     IFLD ... FIELD NUMBER TO BE COMPARED
C     IPTR ... ZERO OFFSET POINTER TO "VIEW" RECORD
C     JPTR ... ZERO OFFSET POINTER TO DESCRIPTION RECORD
C
2     COMMON /STATEV/
3     DIMENSION ITEM(41900),DITEM(41900)
4     EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5     COMMON /PARS/
6     EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7     REAL*8 DTPNAM,FLDNAM,FORMOT
8     COMMON /PARS1/
9     ICMPVE=NULL
10    IF(IFLD.NE.JLOCN) RETURN
C
C ... ONLY LOCATION FIELD PROCESSED FOR VEHICLE
C
11    LOCRFL=ITEM(IFLD+IPTR)
12    LOCRFR=ITEM(IFLD+JPTR)
13    ICMPVE=IFAIL
14    IF(LOCRFL.EQ.NULL) RETURN
C
C ... ONLY THE PLACE FIELD HAS TO AGREE
C
15    IF(IVAL(JPLACE,LOCRFL).EQ.IVAL(JPLACE,LOCRFR)) ICMPVE=IOK
16    RETURN
17    END
```

FSN MODEL - INBLK SUBROUTINE

1 SUBROUTINE INBLK(STATV,STATV2,PARS,DATAV,NSTATV,NSTV2,NPARS,
+ NDATAV)
C
C ROUTINE TO READ LONG BINARY RECORDS PASSED FROM THE INPUT
C PREPROCESSORS.
C
C INPUT PARAMETERS:
C STATV THE FIRST PORTION OF COMMON /STATEV/
C STATV2 THE SECOND PORTION OF COMMON /STATEV/
C PARS A VECTOR EQUIVALENCED TO /PARS/
C DATAV A VECTOR EQUIVALENCED TO /DATAV/
C NSTATV THE SIZE OF STATV
C NSTV2 THE SIZE OF STATV2
C NPARS THE SIZE OF PARS
C NDATAV THE SIZE OF DATAV
C
2 DIMENSION STATV(NSTATV),STATV2(NSTV2),PARS(NPARS),DATAV(NDATAV)
C
3 READ (4) DATAV
4 READ (5) STATV
5 READ (5) STATV2
6 READ (5) PARS
7 RETURN
8 END

FSN MODEL - INIT SUBROUTINE

1 SUBROUTINE INIT
C
C ROUTINE TO INITIALIZE STATE VARIABLES AT THE BEGINNING OF THE
C SIMULATION.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLENAM,FORMAT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
11 COMMON /DATAV/
12 DIMENSION ACTTIM(25)
13 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14 COMMON /PREF/
15 DIMENSION RECREO(140)
16 EQUIVALENCE (RECREO(1),IGOALS)
17 COMMON /GARCOL/
18 LOGICAL*1 ACTIVE
19 COMMON /NEW/
20 DIMENSION MCALSL(10)
21 DIMENSION INSTAT(2,5)
22 EQUIVALENCE (INSTAT(1,1),IOBSV(1))
23 REAL*8 FNAME(6),SNAME(8)
24 DATA PNAME/'RESMOV','RESFIR','RESOBS','RESSUR','RESWND','RESPLN'/
25 DATA SNAME/'SKILL11','SKILL21','SKILL31','SKILL41',
+ 'SKILL12','SKILL22','SKILL32','SKILL42'/
C
C
C ... TEMPORARY CODE UNTIL PARS CORRECTED
26 KINSDR = 4
27 NINSDR = 5

FSN MODEL - INIT SUBROUTINE

ISN

```

28      ISITE = NEWREF(ILSITE,1,0)
29      IPSITE=NEWREF(ILSITE,2,0)
30      DTMIN = DTSEC / 60.
31      ISECPL(1) = 1
32      ISECPL(2) = 2
33      READ(9,101) (IFCOND(I),I=1,2), (IRN(I),I=1,6)
34      101 FORMAT(2I1,6I10)
35      TMPSRP = 1.E10
36      TMIN = 0.0
37      CALL CHGVAR(1,TMIN)
38      NMSTAT = KSAFF
39      CALL CHGVAR(2,NMSTAT)
40      CALL MULTIFY(PFCRF0(4),NRCRF-3)
41      IPHSOB(1) = 0
42      IPHSOB(2) = NWEPTP
43      IPHSOB(3) = IPHSCR(2) + NEOPTP
44      IPHSOB(4) = IPHSCR(3) + NVEHTP
45      IPHSOB(5) = IPHSCR(4) + NPSNTP
46      IPHSOB(6) = IPHSCR(5) + NSENTP

C
C      ESTABLISH INTERNAL BUFFER
47      CALL FTNCMD(*BUFFER 99 LENGTH=300*,20)
C
C      CALL FTNCMD(*SET MODECHECK=CFF*,17)
C
C      INITIALIZE NUMBER OF PLAYERS ON EACH SIDE IN EACH PHYSICAL COND.
C      AND THE INSIDER ARRAY
C
49      DO 5 I=1,NINSDR
50          INSTAT(1,I) = NULL
51          INSTAT(2,I) = 0
52      5 CONTINUE
53      CALL NZERF(FFRCES,8)
54      ISIDE = KSEFF
55      NN = 0

```

FSN MODEL - INIT SUBROUTINE

```
1SN
56      DD 20 ILIST=JGARDS,JADVPS
57      IPSNS = IVAL(ILIST,1SITE)
58      IPSN = IFIRST(IPNSN,KP)
59 10      IF (IPSN .EQ. NULL) GO TO 20
60      CALL PAPSRF(IPSN,IDX,IPLAYR,IPH)
C
C ... REVIEW FUNCTION
C
61      WRITE(7,1002)
62      WRITE(7,1002)
63      WRITE(7,1002)
64      WRITE(7,1002)
65 130  WRITE(7,1003) IPLAYR
66      LOCREF= IVAL(JLOCN,IPSN)
67      IF(LOCREF .EQ. NULL) GO TO 135
68      CALL PARSRF(LOCREF, ID,IREC, IDUM)
69      CALL OUTPLX(ID,IREC,IREC,7,0,0)
C
C ... PLANS
70 135  WRITE(7,1004)
71      IPLIS=IVAL(JPLANS,IPSN)
72      IF(IPLIS .EQ. NULL) GO TO 140
73      CALL OUTREC(IPLIS,7)
C
C ... EQUIPMENT
74 140  WRITE(7,1005)
75      IPLIS=IVAL(JWFAPS,IPSN)
76      IF(IPLIS .EQ. NULL) GO TO 150
77      CALL OUTREC(IPLIS,7)
C
C ... THRESHOLDS
78 150  WRITE(7,1006) -NAME
79      I = IPLAYR
80      WRITE(7,1007) PESMOV(I),PESFIR(I),PESOBS(I),RESSUR(I),
+      RESWND(I),PESPLN(I)
C
```

FSN MODEL - INIT SUBROUTINE

TSN

```

C ... SKILLS
81      153 WRITE(7,1006) SNAME
82          WRITE(7,1007) ((SKILL(J,K,I),J=1,4),K=1,2)
83          ICOND = IVAL(JPSTAT,IPSN)
84          FORCES(ICOND,ISIDE) = FORCES(ICOND,ISIDE) + 1.0
85          IF (IVAL(IITYPE,IPSN) .NE. KINSDR) GO TO 15
86          NN = NN+1
87          INSTAT(1,NN) = IPSN
88      15      IPSN = NEXT(IPSNS,KP)
89          GO TO 10
90      20      IF (ISIDE .NE. KAFF) ISIDE = KAFF
C
91          DO 22 IS=KSFE,KAFF
92          DO 22 IC=1,NPHCAT
93          CALL CHGVR2(5,FORCES(IC,IS),IC,IS)
94      22      CONTINUE
C
C ESTABLISH LIST OF GOALS
C
95          ICOALS = NULL
96          CALL CHGVAR(8,IGEALS)
97          NG = NRECS0(LGOAL)
98          IF (NG .LE. 0) GO TO 40
99          CALL NZERO(MGOALS,NG)
C MARK ALL GOALS WHICH ARE SUCCESSORS OF SOME OTHER GOAL
100         DO 30 I=1,NG
101             IGOLRF = NEWREF(LGOAL,I,0)
102             TSUCRS = IVAL(TSUCRS,IGOLRF)
103             IF (TSUCRS .EQ. NULL) GO TO 30
104             TSUCR = ITPST(TSUCRS,IGI)
105         25     IF (TSUCR .EQ. NULL) GO TO 30
106             CALL PARSRF(TSUCR,TDTYPE,IRECNO,TDUM)
107             MGOALS(IRECNO) = 1
108             TSUCR = NEXT(TSUCRS,IGI)
109             GO TO 25
110         20     CONTINUE

```

FSN MODEL - INIT SUBROUTINE

ISN C CHAIN TOGETHER ALL UNMARKED GOALS
111 DO 35 I=1,NG
112 IGOALRF = NEWREF(IGOAL,I,0)
113 IF (MGOALS(I)) .NE. 0 .OR. TVAL(JADVRO,IGOALRF) .EQ. NULL)
+ GO TO 35
114 CALL CHGLST(IGOAL,2,1,0)
115 IGOALS = IQUEUE(IGOALRF,IGOALS)
116 35 CONTINUE
117 40 CONTINUE
C
118 NEWGOL = NULL
119 LDCAFE = NULL
C
120 IAIMPT(1) = 1
121 IAIMPT(2) = 3
C
C INITIALIZE SENSOR MONITOR STATION. (FOR THIS VERSION, ASSUME JUST
C ONE STATION)
122 IMONRM = NULL
123 NS = NRECS0(LSENS)
124 DO 50 I=1,NS
125 ISENRF = NEWREF(LSENS,I,0)
126 IF(TVAL(JINCOM,ISENRF) .EQ. NULL) GO TO 50
127 IMONRM = IVAL(JINCM,ISENRF)
128 IMONRM = IFIRST(IMONRM,10)
129 GO TO 52
130 50 CONTINUE
131 F2 CONTINUE
C
C INITIALIZE NUCLEAR MATERIAL ROOM. (FOR THIS VERSION, ASSUME JUST
C ONE SUCH ROOM)
132 ISNNRM = NULL
133 NR = NRECS0(LROOM)
134 DO 55 I=1,NR
135 IREF = NEWREF(LROOM,I,0)
136 ICNT = IVAL(JCANTS,IREF)

FSN MODEL - INIT SUBROUTINE

TSN

137 IF (ICONT .EQ. NULL) GO TO 55
138 IFIND = LSERCH(ICONT,NULL,LEQUIP,JTYPE,KSNM)
139 IF (IFIND .NE. NULL) GO TO 60
140 55 CONTINUE
141 GO TO 65
142 60 ISNMRM = IREF
C
C INITIALIZE EXITS FROM SITE. FOR THE CURRENT VERSION ANY GOAL NODE
C WHICH IS NOT THE NUCLEAR MATERIAL ROOM OR SENSOR MONITOR
C STATION WILL BE ASSUMED TO BE AN EXIT.
143 65 NG = NRECS0(LGOAL)
144 IEXITS = NULL
145 DO 70 I=1,NG
146 IREF = NEWREF(LGOAL,I,0)
147 IF (IVAL(JTMRO,IREF) .EQ. NULL) GO TO 70
148 LOC = TMALL(IIC,IREF)
149 IF (LOC .EQ. ISNMRM .OR. LOC .EQ. IMONRM) GO TO 70
150 IEXITS = ISTACK(LOC,IEXITS)
151 70 CONTINUE
C
C INITIALIZE LEADERS' PERCEPTIONS OF OWN FORCES' SITUATIONS TO 1.
152 CALL INITVL(IFRCPC,NRECS0(LPERSN),1)
C
153 JFORCE = JACSEN
154 1002 FORMAT(1H0)
155 1003 FORMAT(' STATUS: PERSON',13,/,3X,'LOCATION')
156 1004 FORMAT('OPLANS')
157 1005 FORMAT('OEQUIPMENT')
158 1006 FORMAT('O',8A8)
159 1007 FORMAT(1X,RFP.2)
160 RETURN
161 END

FSM MODEL - INTLOS FUNCTION

1 FUNCTION INTLOS(I1, I2)
2 C FUNCTION TO CHECK WHETHER NODES I1, I2 HAVE INTERIOR LINE-OF-SIGHT.
3 COMMON /PARS/
4 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
5 REAL*8 DTPNAM,FLDNAM,FORMOT
6 COMMON /PARS1/
7 COMMON /PARS2/
8 COMMON /PARS3/
9 INTLOS = 0
10 IF (I1 .EQ. I2) GO TO 60
11 C FIND A NEIGHBORING REGION
12 NBRR = NBRREG(I1)
13 C HAVE AN EXTERIOR NEIGHBOR--DO NOT USE THIS ROUTINE
14 10 RETURN
15 C NO REGION NEIGHBORS
16 20 NBRR = I1
17 C ITERATE THROUGH NBRS OF THE REGION
18 30 NLIS = IVAL(JNBRS,NBRR)
19 NBR = IFIRST(NLIS,N1)
20 40 IF (NBR .EQ. NULL) RETURN
21 IF (NBR .EQ. I2) GO TO 60
22 C IF THE NBR IS A SEE-THRU PORTAL, CHECK OTHER SIDE
23 CALL PARSRF(NBR, ID, IDUM, IDUM)
24 IF (ID .NE. LDOOR .AND. ID .NE. LWIND) GO TO 50
25 IF (ISEEPT(NBR) .EQ. 0) GO TO 50
26 IF (IVAL(JPORT,NBR) .EQ. I2) GO TO 60
27 50 NBR = NEXT(NLIS,N1)
28 GO TO 40
29 60 INTLOS = 1
30 RETURN
31 END

FSN MODEL - INTSCT FUNCTION

ISN

```

1      FUNCTION INTSCT(A,B,X)
C      A FUNCTION TO CHECK WHETHER OR NOT TWO LINE SEGMENTS ON THE
C      XY PLANE DEFINED BY A;[ (A1,A2), (A3,A4) ] AND B;[ (B1,B2), (B3,B4) ]
C      INTERSECT. THE FUNCTION RETURNS 0 IF NO INTERSECTION, 1 IF
C      A UNIQUE INTERSECTION IN ARRAY X(1), X(2), AND 2 IF THE SEGMENTS
C      ARE COLLINEAR BUT DISJOINT.
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMAT
5      DIMENSION A(1),B(1),X(2)
6      NFLAG = 0
7      INTSCT = 0
8      DXA = A(1)-A(3)
9      DYB = A(2)-A(4)
10     DXB = B(1)-B(3)
11     DYB = B(2)-B(4)
C      IS THE FIRST LINE PARALLEL TO THE Y-AXIS?
12     AX = ABS(DXA)
13     IF (AX .GT. .1) GO TO 50
14     IF (ABS(DYA) .LT. .1) RETURN
15     NFLAG = 1
C      IS THE SECOND LINE PARALLEL TO THE Y-AXIS?
16     50 AX = ABS(DXB)
17     IF (AX .GT. .1) GO TO 80
18     IF (ABS(DYB) .LT. .1) RETURN
19     IF (NFLAG .EQ. 0) GO TO 70
20     AX = ABS(A(1)-B(1))
21     IF (AX .GT. .1) RETURN
C      THE 2 PARALLEL TO THE Y-AXIS LINES ARE APPROXIMATELY COLLINEAR. IS THERE ANY OVERLAP?
22     60 AH = AMAX1(A(2),A(4)) + .001
23     AL = AMIN1(A(2),A(4)) - .001
24     BH = AMAX1(B(2),B(4)) + .001
25     BL = AMIN1(B(2),B(4)) - .001
26     65 IF (BL.LT.AL .AND. BH.GT.AL) RETURN
27     IF (AL.LE.BL .AND. AH.GT.BL) RETURN

```

FSN MODEL - INTSCT FUNCTION

1SN
28 INTSCT = 2
29 RETURN
30 C SECOND LINE PARALLEL TO THE Y-AXIS, FIRST NOT
31 70 IY = INTV(B(1),A(1),X(2))
32 IF (IY .EQ. 0) RETURN
33 X(1) = B(1)
34 GO TO 130
35 80 IF (NFLAG .EQ. 0) GO TO 100
36 C FIRST LINE PARALLEL TO THE Y-AXIS, SECOND NOT
37 IY = INTV(A(1),B(1),X(2))
38 IF (IY .EQ. 0) RETURN
39 X(1) = A(1)
40 GO TO 130
41 C NEITHER LINE PARALLEL TO THE Y-AXIS -- FIND THE INTERSECTION
42 100 S1 = DYA/DXA
43 S2 = DYB/DXB
44 R1 = A(1)*S1 - A(2)
45 R2 = B(1)*S2 - B(2)
46 D = S2 - S1
47 IF (ABS(D) .GT. .1) GO TO 120
48 C PARALLEL LINES -- ARE THEY COLLINEAR?
49 D = ABS(B(2)-A(2)-S1*(B(1)-A(1)))
50 DEL = ABS(.05*(S1+1))
51 IF (D .GT. DEL) RETURN
52 C LINES ARE COLLINEAR -- CHECK FOR OVERLAP
53 AH = AMAX1(A(1),A(3)) + .001
54 AL = AMIN1(A(1),A(3))
55 BH = AMAX1(B(1),B(3)) + .001
56 BL = AMIN1(B(1),B(3))
57 GO TO 65
58 C UNIQUE INTERSECTION
59 120 X(1) = (R2-R1)/D
60 AI = AMAX1(A(1),A(3)) + .001
61 IF (X(1) .GT. AI) RETURN
62 AI = AMIN1(A(1),A(3)) - .001
63 IF (X(1) .LT. AI) RETURN

FSN MODEL - INTSCT FUNCTION

ISN
58 AI = AMAX1(B(1),B(3)) + .001
59 IF (X(1) .GT. AI) RETURN
60 AI = AMIN1(B(1),B(3)) - .001
61 IF (X(1) .LT. AI) RETURN
62 X(2) = (S1*R2 - S2*R1)/D
63 AI = AMAX1(A(2),A(4)) + .001
64 IF (X(2) .GT. AI) RETURN
65 AI = AMIN1(A(2),A(4)) - .001
66 IF (X(2) .LT. AI) RETURN
67 AI = AMAX1(B(2),B(4)) + .001
68 IF (X(2) .GT. AI) RETURN
69 AI = AMIN1(B(2),B(4)) - .001
70 IF (X(2) .LT. AI) RETURN
71 130 INTSCT = 1
72 RETURN
73 END

FSN MODEL - INTER FUNCTION

ISN

```
1      FUNCTION INTER(IPLACE)
C      FUNCTION TO DETERMINE THE 'INTERIOR STATUS' OF AN PLACE.
C      RETURNS ONE OF THESE VALUES
C          -1 = ERROR
C              0 = COMPLETELY OUTSIDE
C              1 = OUTSIDE, BUT AT AN OPEN PORTAL
C              2 = INSIDE, BUT AT AN OPEN EXTERIOR PORTAL
C              3 = INSIDE, BUT AT AN OPEN INTERIOR PORTAL
C              4 = INSIDE, NOT AT A PORTAL
C      HERE 'OPEN' MEANS OPEN, DESTROYED, OR TRANSPARENT.
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      CALL PARSRF(IPLACE, IDTYP, IDUM, IDUM)
9      IF (IDTYP .EQ. LYARD) GO TO 900
10     IF (IDTYP .EQ. LROOF) GO TO 900
11     IF (IDTYP .EQ. LROOM) GO TO 904
12     IF (IDTYP .EQ. LHALL) GO TO 904
13     N = NBRRE(IPLACE)
14     IF (N) 200,904,300
15     C      HAVE AN EXTERIOR PORTAL OR STAIR
16     200    IF (IDTYP .EQ. LSTAIR) GO TO 900
17     C      CAN SEE THROUGH PORTAL?
18     16      IF (ISEEPT(IPLACE) .EQ. 0) GO TO 900
19     17      GO TO 901
20     C      HAVE AN INTERIOR PORTAL OR STAIR
21     18      IF (IDTYP .EQ. LSTAIR) GO TO 904
22     19      IF (ISEEPT(IPLACE) .EQ. 0) GO TO 904
23     C      IS THE OTHER SIDE ALSO INTERIOR?
24     20      IPORT = IVAL(JPORT,IPLACE)
25     21      NOPP = NBRREG(IPORT)
```

FSN MODEL - INTER FUNCTION

ISN

22		IF (NOOP .LT. 0) GO TO 902
23		GO TO 903
24	900	INTER=0
25		RETURN
26	901	INTER=1
27		RETURN
28	902	INTER=2
29		RETURN
30	903	INTER=3
31		RETURN
32	904	INTER=4
33		RETURN
34		END

FSN MODEL - INTV FUNCTION

ISN

```
1      FUNCTION INTV(X,C,YVAL)
C      FUNCTION TO SEE IF THE LINE X:[(X1,X2),(X3,X4)], PARALLEL TO
C      THE Y-AXIS INTERSECTS THE
C      LINE DEFINED BY C:[(C1,C2),(C3,C4)]. RETURNS 1 IF
C      INTERSECTION, 0 IF NONE, AND THE Y COORDINATE IN YVAL.
2      DIMENSION C(4),X(4)
3      INTV = 0
4      XL = AMIN1(C(1),C(3))
5      IF (X(1) .LT. XL) RETURN
6      XR = AMAX1(C(1),C(3))
7      IF (X(1) .GT. XR) RETURN
8      YVAL = (X(1)-C(1))*(C(2)-C(4))/(C(1)-C(3)) + C(2)
9      YL = AMIN1(X(2),X(4))
10     IF (YVAL .LT. YL) RETURN
11     YH = AMAX1(X(2),X(4))
12     IF (YVAL .GT. YH) RETURN
13     INTV = 1
14     RETURN
15     END
```

FSN MODEL - IPATH FUNCTION

ISN

1

FUNCTION IPATH(IPLAYR,ISOURC,IGOAL,DMEASR)

C

C ROUTINE TO COMPUTE AN OPTIMAL PATH OF A PLAYER FROM AN INITIAL
C POSITION TO A DESTINATION SPECIFIED BY IGOAL. THE FUNCTION USES
C DIJKSTRA'S METHOD TO COMPUTE AN OPTIMAL PATH. IT CALLS ON AN
C EVALUATION FUNCTION TO COMPUTE A NON-NEGATIVE MEASURE OF
C DIFFICULTY OR TIME FOR EACH LINK ENCOUNTERED. THE ALGORITHM
C ATTEMPTS TO MINIMIZE THIS MEASURE OVER THE PATH. IF TWO PATHS
C HAVE THE SAME MEASURE THE ALGORITHM MAKES AN ARBITRARY SELECTION
C BETWEEN THEM. IF THERE DOES NOT EXIST ANY PATH FROM ISOURC TO
C IGOAL AN ERROR COMMENT IS PRINTED.

C THE ROUTINE RETURNS THE PATH COMPUTED AS ITS VALUE. IT ALSO
C RETURNS THE MEASURE COMPUTED BY THE EVALUATION FUNCTION SUMMED
C OVER THE LINKS OF THE PATH.

C

C INPUT PARAMETERS:

C IPLAYR A REFERENCE TO THE PERSON OR VEHICLE TO BE TRAVERSING
C THE PATH.

C ISOURC A REFERENCE TO A PERSON, VEHICLE, SENSOR, EFFECTOR
C (ACTIVATED DELAY) OR TO A SITE NODE MARKING THE
C BEGINNING OF THE PATH.

C IGOAL A REFERENCE TO A PERSON, VEHICLE, SENSOR, EFFECTOR
C (ACTIVATED DELAY) OR TO A SITE NODE MARKING THE
C END OF THE PATH.

C

C OUTPUT PARAMETERS:

C DMEASR THE SUM OF THE EVALUATION MEASURE OVER THE OPTIMAL
C PATH.

C

2 COMMON /PARS/

3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)

4 REAL*8 DTPNAM,FLDNAM,FORMOT

5 COMMON /PARS1/

6 COMMON /PARS2/

7 COMMON /PARS3/

FSN MODEL - IPATH FUNCTION

```
1$N
 8      COMMON // IPREV(362), IACTIV(100), D(362)
 9      DATA NACLIS/100/
C
C  INITIALIZATIONS
C
10     IO = IRCOFF(LYARD)
11     NNODES = IRCOFF(LWEAP) - IO
12     CALL NULIFY(IPREV,NNODES)
13     CALL NZERO(D,NNODES)
14     CALL NULIFY(IACTIV,NACLIS)
C
C  DETERMINE SOURCE AND GOAL NODES
C
15     NODSOU = ISOURC
16     CALL PARSRF(ISOURC, IDTYP, IDUM, IDUM)
17     IF (IDTPOK(IDTYP) .EQ. 2) NODSOU = IVAL(JPLACE, IVAL(JLOCN, ISOURC))
18     IF (IDTPOK(IDTYP) .LE. 0) CALL ERR(17, 14, ISOURC, IDTYP, 0)
C
19     NODGOL = IGOAL
20     CALL PARSRF(IGOAL, IDTYP, IDUM, IDUM)
21     IF (IDTPOK(IDTYP) .EQ. 2) NODGOL = IVAL(JPLACE, IVAL(JLOCN, IGOAL))
22     IF (IDTPOK(IDTYP) .LE. 0) CALL ERR(17, 14, IGOAL, IDTYP, 0)
C
23     NODCUR = NODSOU
24     DMCUR = 0.
25     CALL PARSRF(NODCUR, IDTYP1, IRECNO, IDUM)
26     IOFF = IRCOFF(IDTYP1) - IO + IRECNO
27     IPREV(IOFF) = -1
28     NA = 0
C
C  CHECK WHETHER HAVE REACHED THE GOAL NODE.
C
29 10     IF (NODCUR .EQ. NODGOL) GO TO 100
C
C  ITERATE THROUGH THE NEIGHBORS OF THE CURRENT NODE.
C
```

FSN MODEL - IPATH FUNCTION

```

1SN
30      NBRS = IVAL(JNBRS,NODCUR)
31      NBR = IFIRST(NBRS,KN)
32      NBR2 = NULL
33  15      IF (NBR .EQ. NULL) GO TO 32
C
C      CHECK WHETHER THIS NEIGHBOR HAS BEEN "USED" OR IS "ACTIVE".
C
34          CALL PARSRF(NBR, IDTYP,IRECNO, IDUM)
35          IOFF = IRCOFF(IDTYP) - IO + IRECNO
36          IF (IPREV(IOFF) .NE. NULL) GO TO 30
C      IF FOUND A NON-ACTIVE, NON-USED NODE, ADD IT TO THE ACTIVE LIST
37          DM = EFTIM(IPLAYR,NODCUR,NBR,KRUN) + DMCUR
38          NA = NA + 1
39          IF (NA .GT. NACLIS) CALL ERR(18,14,1,IPLAYR,NACLIS)
40          IACTIV(NA) = IOFF
41          D(IOFF) = DM
42          IPREV(IOFF) = NODCUR
C
C      TEST IF PORTAL NEIGHBOR CHECKED
C
43      30 IF (NBR2 .NE. NULL) GO TO 40
C      GET NEXT NEIGHBOR
44          NBR = NEXT(NBRS,KN)
45          GO TO 15
C      IF THIS IS A DOOR, WINDOW OR STAIRS NODE, GET SPECIAL NEIGHBOR IF
C      HAVEN'T YET
46  32      IF (IDTYP1 .LT. LSTAIR .OR. IDTYP1 .GT. LWIND) GO TO 40
47          IF (IDTYP1 .EQ. LSTAIR) NBR2 = IVAL(JSTAIR,NODCUR)
48          IF (IDTYP1 .NE. LSTAIR) NBR2 = IVAL(JPORT,NODCUR)
49          IF (NBR .EQ. NBR2) GO TO 40
50          NBR = NBR2
51          GO TO 15
C
C      GET NODE FROM ACTIVE LIST HAVING MINIMUM ASSOCIATED MEASURE
C
52  40      DMIN = 1.E8

```

FSN MODEL - IPATH FUNCTION

1SN
53 IF (NA .LE. 0) CALL ERR(19,14,NODCUR,ISOURC,IGOAL)
54 DO 50 IA=1,NA
55 IOFF = IACTIV(IA)
56 IF (D(IOFF) .GT. DMIN) GO TO 50
57 DMIN = D(IOFF)
58 ICUR = IA
59 50 CONTINUE
C
C CONVERT OFFSET OF MINIMUM NODE TO A NODE REFERENCE
C
60 IOFF = IACTIV(ICUR)
61 DO 55 IDTYP=LYARD,LVEHIC
62 IF (IOFF .GT. IRCOFF(IDTYP) - IO) GO TO 55
63 IDTYP1 = IDTYP - 1
64 NODCUR = 1000000000 + 10000*IDTYP1 + IOFF - IRCOFF(IDTYP1) + IO
65 GO TO 60
66 55 CONTINUE
C
C DELETE NODE FROM ACTIVE LIST
C
67 60 DMCUR = DMIN
68 IACTIV(ICUR) = IACTIV(NA)
69 NA = NA - 1
70 GO TO 10
C
C HAVE FOUND PATH. SAVE MEASURE. RETRACE STEPS AND GENERATE PATH
C IN FORWARD DIRECTION.
C
71 100 DMEASH = DMCUR
72 IPATH = NODCUR
73 110 CALL PAESRF(NODCUR, IDTYP,IRECNO, IDUM)
74 IOFF = IRCOFF(IDTYP) - IO + IRECNO
75 NODCUR = IPREV(IOFF)
76 IF (NODCUR .LT. 0) RETURN
77 IPATH = ISTACK(NODCUR,IPATH)
78 GO TO 110

FSN MODEL - IPATH FUNCTION

ISBN
79

END

FSN MODEL - ISEEPT FUNCTION

ISN

```
1      FUNCTION ISEEPT(IPORT)
C      FUNCTION TO CHECK WHETHER CAN SEE THROUGH A PORTAL.
C      RETURNS 0 IF NO, 1 IF YES
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      ISEEPT = 1
9      IF (IVAL(JSTAT,IPORT) .LE. KOPEN) RETURN
10     IF (IVAL(JVIS,IPORT) .EQ. KTRANP) RETURN
11     ISEEPT = 0
12     RETURN
13     END
```

FSN MODEL - TERM FUNCTION

ISM

```

1      FUNCTION ITEM(ICOND)
C ... THIS FUNCTION CHECKS FOR GAME TERMINATION CONDITIONS
C AND RETURNS APPROPRIATE VALUE:
C           0 ... CONTINUE GAME
C           KSFE ... TERMINATE GAME, SFE WINS
C           KAFE ... TERMINATE GAME, AFE WINS
C
C ... CONDITIONS FOR SFE WIN:
C   1. AFE DOES NOT HAVE ENOUGH PLAYERS LEFT
C   2. SIMULATED ARRIVAL OF RESPONSE FORCE BEFORE
C      AFE ACHIEVES A FINAL GOAL
C
C ... CONDITIONS FOR AFE WIN:
C   1. SFE DOES NOT HAVE ENOUGH PLAYERS LEFT
C   2. AFE ACHIEVES FINAL GOAL BEFORE ARRIVAL
C      OF RESPONSE FORCE.
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTAIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7      REAL*P DTPNAM,FLDNAM,FORMAT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /D&TAM/
12     DIMENSION ACTTIM(25)
13     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14     COMMON /RECREF/
15     DIMENSION RECREF(1/0)
16     EQUIVALENCE (RECREF(1),IGOALS)
17     COMMON /INW/
18     DIMENSION INSTATE(2,5)
19     EQUIVALENCE (INBSV(4),INSTAT(1,1))

```

FSN MODEL - TTERM FUNCTION

ISN
20 TSURF = TFIN
21 TNSIDL = NULL
22 KINSDR = 4
C
23 NEWGOL = NULL
24 IFLAG=NULL
25 IFLAG1 = NULL
C
C ... CHECK THAT AFE HAS ENOUGH PLAYERS LEFT
C
26 MFORCE=FORCES(KWCOND,KAFE)+FORCES(KWHOLE,KAFE)
27 IF(MFORCE.GE.MINPLR(KAFE)) GO TO 10
28 TTERM=KSFF
29 RETURN
C
C ... CHECK THAT SEE HAS ENOUGH PLAYERS LEFT
C
30 10 MFORCE=FORCES(KWCOND,KSFE)+FORCES(KWHOLE,KSFE)
31 IF(MFORCE.GE.MINPLR(KSFE)) GO TO 20
32 TTERM=KAFE
33 RETURN
C
C ... START LOOP THROUGH ACTIVE GOALS
C
34 20 IGOLRF=IFIRSTIGOALS,ICUR)
35 22 IF(IGOLRF.EQ.NULL) GO TO 50
C
C ... COUNT NUMBER OF EFFECTIVE AFE AT GOAL WITH
C PROPER EQUIPMENT
C
36 30 TDAFE=IVAL(JID,IGOLRF)
37 32 NFAFE=0
C
C ... MAKE COPY OF EQUIPMENT REQUIREMENTS
C
38 40 IEQLIS=LCOPY(IVAL(JEOPRO,IGOLRF))

FSN MODEL - ITEM FUNCTION

ISN

```

C
C ... START LOOP THROUGH CONTENTS OF GOAL
C
35      LISREF=IVAL(JCENTS, IDRF)
40      IPLARF=IFIRST(LISREF, LPTR)
41      IF(IPLARF.EQ.NULL) GO TO 40
C
C ... IF REQUIREMENTS NOT MET, GET NEXT CONTENT
C
42      CALL PARSRF(IPLARF, IDTYP,IRECNU,IFLD)
43      IF(IDTYP.NE.LPERSN) GO TO 35
44      IF(IVAL(JALLEG,IPLARF).NE.KAFE) GO TO 35
45      IF(IVAL(IPSTAT,IPLARF).LT.KWOUND) GO TO 35
46      IF (IVAL(JTYPE,IPLARF).EQ. KINSR) INSLD = ISTACK(IPLARF,INSLD)

C
C ... EFFECTIVE AFE AT GOAL, LOOP THROUGH
C EQUIPMENT REQUIREMENTS, CROSSING OFF ANY SATISFIED BY PERSON
C
47      IEORF1=IFIRST(IEOLIS,LPTE01)
48      IF(IEORF1.EQ.NULL) GO TO 33
49      CALL PARSRF(IEORF1, IDTYP,IRECNU,IFLD)
50      ITYP=IVAL(JTYPE,IEQRF1)
51      JECL'S=IVAL(JWEAPS,IPLARF)

C
C ... DOES PLAYER HAVE THIS EQUIPMENT?
C
52      IFIND=LSFRCH(IEOLIS,NULL, IDTYP,JTYPE,ITYP)
53      IF(IFIND.NE.NULL) CALL DELTST(LPTE01,0,IEOLIS)

C
C ... END EQUIPMENT REQUIRED LOOP
C
54      IEORF1=NEXT(IEOLIS,LPTE01)
55      GO TO 26
C
C ... EFFECTIVE AFE AT GOAL
C

```

FSN MODEL - ITERM FUNCTION

ISM
56 33 NEFAFE=NEFAFF+1
C
C ... END LOOP THRGUTH GOAL CONTENTS
C
57 35 IPLARF=NEXT(LISREF,LPTP)
58 GO TO 24
C
C ... IF AFF DOES NOT CONTROL GOAL, CHECK NEXT GOAL
C
59 40 IF(NEFAFE.LT.IVAL(JADVR0,IGOLRF)) GO TO 49
C
C ... CHECK WHETHER ALL EQUIPMENT REQUIREMENTS SATISFIED
C
60 IF (IEOLTS .NE. NULL) GO TO 49
C
C ... INCREMENT TIME CONTROLLED, CHECK FOR COMPLETION
C
61 INDEX=TRSF(JTIM,IGOLRF)
IF(ITEM(INDEX) .EQ. NULL) DITEM(INDEX)=0.0
63 DITEM(INDEX)=DITEM(INDEX)+DTMIN
IF(DITEM(INDEX).LT.VAL(JTIMR0,IGOLRF)) GO TO 49
C
C ... GOAL COMPLETED
C
C IF AN INSIDER GOAL, SET FLAG FOR COMPLETED ACTION
65 INSIDR = IFIRST(INSIDE,INS)
66 401 IF (INSIDR .EQ. NULL) GO TO 405
C FIND THE INSIDER'S INDEX IN THE INSIDER ARRAY
67 DO 402 K=1,5
68 IF (INSTAT(1,K) .EQ. INSIDR) GO TO 403
69 402 CONTINUE
70 CALL EPR(41,38,INSIDR,0,0)
71 GO TO 404
72 403 INSTAT(2,K) = 1
73 404 INSIDR = NEXT(INSIDE,INS)
74 405 INSIDE = NULL

FSN MODEL - ITEM FUNCTION

ISN

75 NEWGOAL = 1
 76 IFLAG=ISTACK(IGOLRF,IFLAG)
 77 NXTLIS=IVAL(JSUCRS,IGOLRF)
 78 IF(NXTLIS.EQ.NULL) GO TO 49

C
 C ... ACTIVATE SUCCESSOR GOALS
 C

79 41 NXTREF=IFIRST(NXTLIS,LPTR)
 80 42 IF(NXTREF.EQ.NULL) GO TO 49
 81 CALL CHGLST(NXTREF,2,1,0)
 82 IFLAG1 = IQUEUE(NXTREF,IFLAG1)
 83 NXTREF=NEXT(NXTLIS,LPTR)
 84 GO TO 42

C
 C ... END LOOP THROUGH GOALS
 C

85 49 IGOLRF=NEXT(I GOALS,ICUR)
 86 GO TO 22

C
 C DELETE COMPLETED GOALS FROM LIST
 C

87 EQ IF(IFLAG .EQ. NULL) GO TO 90
 88 ILST=IFIRST(IFLAG,KAI)
 89 55 IF(ILST .EQ. NULL) GO TO 80

C
 C FIND ILST ON GOALS LIST
 C

90 IGLL=IFIRST(I GOALS,ICUR)
 91 60 IF(IGLL .EQ. NULL) GO TO 70
 92 IF(IGLL .EQ. ILST) GO TO 65
 93 IGLL=NEXT(I GOALS,ICUR)
 94 GO TO 60

C
 95 65 IF (ICUR .NE. NULL) GO TO 57
 95 I GOALS =NULL
 97 GO TO 80

FSN MODEL - TTERM FUNCTION

15N
98 57 CALL DELIST(ICUR,0,IGOALS)
C
99 70 TEST=NEXT(IFLAG,KA)
100 GO TO 55
C
C TEST IF ANY GOALS LEFT
C
101 80 IGOALS = JCIN(IGOALS,IFLAG)
102 IF(IGOALS .NE. NULL) GO TO 90
103 ITTERM=KAFE
104 RETURN
C ... IF TIME EXPIRED, SEE WINS
C
105 90 ITTERM=0
C IF IT IS TIME TO SURFACE, INSIDERS BECOME COMBATANTS
106 DO 100 K=1,5
107 IPRF = INSTAT(1,K)
108 IF (IPRF .EQ. NULL) GO TO 200
109 100 CALL SURFAC(IPRF)
110 200 TSURF = TFIN + 1.
111 IF(TFIN.GT.TMIN) RETURN
112 ITTERM=KSFE
113 RETURN
114 END

FSN MODEL - KEYS SUBROUTINE

ISN

1 SUBROUTINE KEYS(IPLARF,IPORT,IEQRF)
C
C -- THE PURPOSE OF THIS SUBROUTINE IS TO:
C 1. DETERMINE IF A PLAYER HAS KEYS TO
C LOCK OR UNLOCK A PORTAL
C
C -- INPUT PARAMETERS
C
C IPLARF -- PLAYER REFERENCE
C IPORT -- PORTAL REFERENCE
C
C -- OUTPUT VARIABLES
C
C IEQRF -- EQUIPMENT REFERENCE
C
C -- PROCEDURE
C 1. DETERMINE THAT PORTAL IS LOCKED
C 2. FIND ACCESS TYPE (KEY PARAMETER)
C 3. SEARCH PLAYERS EQUIPMENT LIST TO
C FIND KEY WITH PARAMETER EQUAL TO
C ACCESS OF PORTAL
C
C
2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FORMOT
5 COMMON /PARS1/
6 COMMON /PARS2/
7 COMMON /PARS3/
C
C
8 IEQRF=NULL
C
C -- PORTAL STATUS
C

FSN MODEL - KEYS SUBROUTINE

ISN
9 IF(IVAL(JSTAT,IPORT) .LT. KLOCKD) IEQRF=0
10 IF(IVAL(JSTAT,IPORT) .NE. KLOCKD) GO TO 100
C
C -- ASSUMPTION
C IF LOACKABLE WITH KEY STATUS THEN ALSO
C UNLOKABLE WITH KEY
C IF LOCKABLE WITHOUT KEY THEN ALSO UNLOCKABLE
C WITHOUT KEY
C
C FIND LOCKABLITY OF PORTAL
C
11 ILOC=IVAL(JLOCK,IPORT)
12 IF(ILOCK .EQ. 1) GO TO 100
13 IF(ILOCK .EQ. 2) GO TO 5
C
C --LOCKABLE WITHOUT KEY
C
14 IEQRF=0
15 GO TO 100
C
C -- FIND ACCESS TYPE
C
16 5 IACS=IVAL(JDRACC,IPORT)
C
C -- FIND KEY TO MATCH
C
17 IEQLT=IVAL(JWEAPS,IPLARF)
18 IEQUIP=IFIRST(IEQLT,KA)
19 10 IF(IEQUIP .EQ. NULL) CO TO 100
C
C -- IGNORE WEAPONS
C
20 CALL PARSRF(IEQUIP, IDY, IRR, IDU)
21 IF(IDY .NE. LEQUIP) GO TO 20
C
C -- TEST FOR EQUIPMENT TYPE KEY

FSN MODEL - KEYS SUBROUTINE

ISN

C

22 IF (IVAL(JTYPE,IEQUIP) .NE. KKEY) GO TO 20
23 IF (IVAL(JPAR,IEQUIP) .EQ. 0) GO TO 40
24 IF (IVAL(JPAR,IEQUIP) .EQ. IACS) GO TO 40

C

25 20 IEQUIP=NEXT(IEQLT,KA)
26 GO TO 10

C

C -- KEY FOUND

C

27 40 IEQRF=IEQUIP

C

28 100 CONTINUE
29 RETURN
30 END

*

FSN MODEL - LDRPER SUBROUTINE

1 ISN

1 SUBROUTINE LDRPER(IPLARF,LDRFRC,MSITNF)

2 C ... THE PURPOSE OF THIS SUBROUTINE IS TO UPDATE LEADER FORCE

3 C PERCEPTIONS. A LEADER HAS A PERCEPTION OF EACH FORCE ON HIS

4 C SIDE, AND A SINGLE FORCE PERCEPTION INCLUDING ALL OF THE

5 C OPPONING PLAYERS OF WHICH HE IS AWARE.

6 C ... FOR A DESIGNATED LEADER, PROCESS IS AS FOLLOWS:

7 1. SEARCH PLAYERS LEADER PERCEIVES TO INSURE THAT HE

8 C HAS A FORCE PERCEPTION OF THE APPROPRIATE FORCE.

9 2. INSURE THAT PLAYER IS INCLUDED IN PERCEPTION OF

10 C APPROPRIATE FORCE CONTENTS.

11 3. PROCESS ONE FORCE PERCEPTION AT A TIME, UPDATING

12 C IT FROM THE LEADER'S PERCEPTIONS OF THE PLAYERS

13 C CONSTITUTING THE FORCE.

14 C ... FP=0 IS ALWAYS THE ENTRY IN THE JID FIELD FOR THE

15 C OPPONING FORCE PERCEPTION.

16 C

17 C INPUT PARAMETERS:

18 IPLARF REFERENCE TO LEADER

19 LDRFRC REFERENCE TO FORCE THAT LEADER COMMANDS

20 MSITNF LEADER'S PREVIOUS PERCEPTION OF HIS FORCE'S SITUATION

21 C

22 C OUTPUT PARAMETERS:

23 MSITNF LEADER'S CURRENT PERCEPTION OF HIS FORCE'S SITUATION

24 C

25 2 COMMON /STATEM/

26 3 DIMENSION ITEM(41900),DITEM(41900)

27 4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))

28 5 COMMON /PARS/

29 6 EQUIVALENCE (ENULL,NULL),ETFAIL,FAIL)

30 7 REAL*8 DTPNAM,FLENAM,FORMOT

31 8 COMMON /PARS1/

32 9 COMMON /PARS2/

33 10 COMMON /PARS3/

34 11 COMMON /RECREF/

35 12 DIMENSION RECREFO(140)

FSN MODEL - LDRPER SUBROUTINE

15N
13 EQUIVALENCE (RECREF(1),IGOALS)
14 COMMON /NEW/
C
C -- UPDATE LOCAFE BY PLACE OF LEADER
C
15 IF(IVAL(JALLEG,IPLARE) .EQ. KAFE)
* LOCAFE=PLACE(IVAL(JLOCN,IPLARE))
C
C RETURN IF NULL PERCEPTIONS
C
16 IF(IVAL(JPRCPS,IPLARE) .EQ. NULL) RETURN
C
C START LIST THROUGH LEADER'S PERCEPTIONS
C
17 IPCLIS=IVAL(JPRCPS,IPLARE)
18 IPCPREF=IFIRST(IPCLIS,N2)
19 15 IF(IPCPREF .EQ. NULL) GO TO 100
20 IRECRF=IVAL(JID,IPCPREF)
C
C IF PERCEPTION NOT OF A PERSON, IGNORE
C
21 IF(IRECRF .EQ. NULL) GO TO 90
22 CALL PARSRF(IRECRF,TDTYP,IRECNO,1DUM)
23 IF(TDTYP .NE. LPERSN) GO TO 90
C
C IF PERCEPTION OF OPPONENT REQUIRE FR=0
C
24 IF(IVAL(JALLEG,IRECRF) .EQ. ISIDE) GO TO 20
25 IFORCE=NEWREF(IFORCE+0,0)
26 GO TO 30
C
C IF FRIEND, NEED SPECIFIC FORCE
C
27 20 IFORCE=IVAL(JFORCE,IRECRF)
C
C IF LEADER HAS PERCEPTION OF IFORCE

FSN MODEL - LDRPER SUBROUTINE

1SN
C
29 30 ION=LSERCH(IPCLIS,NULL,NULL,SID,IFORCE)
29 IF(ION .NE. NULL) GO TO 40
C
C CREATE APPROPRIATE FORCE PERCEPTION
C
30 ION=NEWREF(LPERCP,NEWREC(LPERCP),0)
31 CALL CHGFLD(JIDIV,ION,IFORCE,1,NULL)
32 NEWFC=NEWREC(IFORCE,NEWPEC(LFORCE),0)
33 CALL CHGFLD(JVIEW,ION,NEWFC,1,NULL)
C
C ADD PERCEPTION TO IPCLIS
C
34 IF(N2.NE.NULL) GO TO 35
C
C ... CURRENTLY SINGLE REF, NOT LIST
C
35 IPCLIS = ISTACK(IPCLIS,ION)
36 IPTR = IREF(JPRCPS,IPLARF)
37 ITEM(IPTR) = IPCLIS
38 IPCPREF = IFTRST(IPCLIS,N2)
39 GO TO 40
40 25 N3=NEWREC(LLTST)
41 LISTND(JNXT,N3)=LISTND(JNXT,N2)
42 LISTND(JNXT,N2)=N3
43 LISTND(JVAL,N3)=ION
C
C ... INSURE PLAYER INCLUDED IN CONTENTS OF PERCEIVED FORCE
C
44 40 IFRCRF = IVAL(JVIEW,ION)
45 IDNLIS=IVAL(JCNTS,IFRCRF)
46 TON=LSERCH(TCNLIS,IPECRF,NULL,NULL,NULL)
47 IF(TON .NE. NULL) GO TO 40
C
C ADD PLAYER TO CONTENTS LIST OF FORCES PERCEIVED

FEN MODEL - LOWER SUBROUTINE

```

1SN
48      TONLIS=ISTACK(IRECFF,IONLIS)
49      CALL CHCFLD(JCONTS,IFRCFF,IONLIS,L,NULL)
C
C      END FIRST LOOP THROUGH PERCEPTIONS
C
50  00      IPCPREF=NEXT(IPCLIS,N2)
51      GO TO 15
C
C      ALL PLAYERS THAT LEADER PERCEIVES ARE INCLUDED AS CONTENTS
C      IN ONE OF HIS FORCE PERCEPTIONS. NOW PROCESS HIS FORCE
C      PERCEPTIONS, UPDATING THE INFORMATION BASED ON HIS PERCEPTIONS
C      OF PLAYERS. UPDATING PROCEDURES AS FOLLOWS:
C
C      FIELD          FRIEND          FOE
C      ITYPE          GET FROM ACTUAL FORCE  DEFAULT=FIRST ECHelon
C
C      JALLEG         FRIEND          FOE
C
C      JFDP           GET FROM ACTUAL FORCE  DEFAULT=NULL
C
C      JCONTS         AS NOTED        AS NOTED
C
C      JFSTAT         COMPUTED FROM PLAYER    COMPUTED FROM PLAYER
C                      PERCEPTIONS          PERCEPTIONS
C      JSITN          MAX OF PERSONAL    MAX OF PERSONAL
C                      JSITN OF CONTENTS  JSITN OF CONTENTS
C
C      JFPLNS         COPY FROM ACTUAL FORCE  NULL
C
C      JFSOPS         COPY FROM ACTUAL FORCE  NULL
C
C      ... START SECOND LOOP THROUGH PERCEPTIONS
C      LOOKING FOR FORCE PERCEPTIONS TO UPDATE
52  100     IPCLIS=IVAL(JPRCPF,IPLARE)
53     IPCPREF=IFIRST(IPCLIS,N2)
54  115     IF(IPCPREF .EQ. NULL) RETURN

```

FSN MODEL - LDRPER SUBROUTINE

```
1SN
55      IRECRF=IVAL(JVIEW,IPCPRF)
C
C ... IGNORE IF NOT A FORCE
C
56      IF(IRFCRF .EQ. NULL) GO TO 170
57      CALL PARSRF(IRECRF, IDTYP,IRECNO, IDUM)
58      IF(IDTYP .NE. LFORCE) GO TO 170
59      IPTR=IREF1(IRECRF)
C
C INITIALIZE STATUS AND SITUATION COUNTERS
C
60      CONTT=0.0
61      WHOLE=0.0
62      MSITN=1
C
C GET FIXED VALUES
C
63      IBASRF=IVAL(JID,IPCPRF)
64      IF(IBASRF .EQ. NULL) GO TO 170
65      CALL PARSRF(IBASRF, IDTYP,IRECNO, IDUM)
66      IF(IRECNO .EQ. 0) GO TO 120
C
C SETUP FOR FRIEND
C
67      ITEM(IPTR+JTYPE)=IVAL(JTYPE,IBASRF)
68      ITEM(IPTR+JALEG)=ISIDE
69      ITEM(IPTR+JFLDR)=IVAL(JFLDR,IBASRF)
70      ITEM(IPTR+JFPLNS)=IVAL(JFPLNS,IBASRF)
71      ITEM(IPTR+JFSOPS)=IVAL(JFSOPS,IBASRF)
72      GO TO 130
C
C SET UP FOR FOE
C
73      120     ITEM(IPTR+JTYPE)=KECH1
74      ITEM(IPTR+JALEG)=NALLEG+1-ISIDE
```

FSN MODEL - LDRPER SUBROUTINE

1SN C LOOP THROUGH PERCEIVED FORCE CONTENTS
 C USING PERCEPTIONS OF MEMBERS TO UPDATE STATUS ...
 75 130 LISCON=IVAL(JCONTNS,IRECREF)
 76 IPRF=IFIRST(LISCON,N3)
 77 135 IF(IPRF .EQ. NULL) GO TO 160
 78 CALL PARSREF(IPRF,ITTYP,TRECDN,IDUM)
 79 IF(ITYP .NE. EPERSON) GO TO 150
 C
 C GET VIEW OF PLAYER
 C
 80 TON=LSEARCH(IPCLIS,NULL,NULL,1ID,IPRF)
 81 IF(TON .EQ. NULL) GO TO 150
 82 CONTT=CONTT+1.0
 83 IVWREF=IVAL(JVIEW,TON)
 84 IF(IVAL(IPSTAT,IVWREF) .GE. KWOUND .OR.
 + IVAL(IPSTAT,IVWREF) .EQ. NULL)
 + WHOLE=WHOLE+1.0
 85 MSITN=MAX0(MSITN,IVAL(JSITN,IVWREF))
 C
 C END LOOP THROUGH CONTENTS
 C
 86 150 IPRF=NEXT(LISCON,N3)
 87 GO TO 135
 C
 C UPDATE FORCE PERCEPTION
 C
 88 150 STATUS=0.0
 89 MSIDE=IVAL(JFALEG,IRECREF)
 90 IF(MSIDE .EQ. KSFE .AND. MSITN .GE. KSITN3)
 + MSITN=MSITN+1
 91 IF(CONTT .LE. 0.01) GO TO 165
 92 STATUS=WHOLE/CONTT
 C
 C IF STATUS DROPS, LEADER PLANNING
 C
 93 IF(VAL(JFSTAT,IRECREF) .LE. STATUS) GO TO 165

FEN MODEL - LDRPER SUBROUTINE

15N
94 IF(ISIDE .EQ. KSEED) MSITN=KSITN5
95 IF(ISIDE .EQ. KAFFE) MSITN=KSITN4
96 165 CALL CHGFLD(JSTAT,IRECRF,STATUS,1,NULL)
97 IFLG=IVAL(JFALEG,IRECRF)
98 IF(ISIDE .EQ. IFLG) MSITNF=MSITN
99 IF(MSITN .EQ. IVAL(JSITN,IRECRF)) GO TO 170
100 CALL CHGFLD(JSITN,IRECRF,MSITN,1,NULL)
C
C END LOOP THRL PERCEPTIONS
C
101 170 TPCPRF=NEXTI(IPCLIS,N2)
102 GO TO 115
103 END

FEN MODEL - LDRPLN SUBROUTINE

ISN

```

1      SUBROUTINE LDRPLN
C
C      ROUTINE TO PERFORM LEADER PLANNING AND TO IMPLEMENT A LEADER'S
C      TRANSMISSION OF ORDERS TO HIS SUBORDINATES.
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DTITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NUL1),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /DATAV/
12     DIMENSION ACTTIME(25)
13     EQUIVALENCE (ACTTIME(1),ACTRAT(1))
14     COMMON /RECREF/
15     DIMENSION RECPEO(140)
16     EQUIVALENCE (RECPEO(1),EGOALS)
17     COMMON /NEW/
C
C      ITERATE OVER SIDE
18     DO 100 ISIDE = 1,2
C
C      ITERATE OVER PLAYERS ON EACH SIDE
19     IFLD = JGARDS
20     IF (ISIDE .EQ. 2) IFLD = JADVRS
21     IPSNS = IVAL(IFLD,ISIDE)
22     IPEPSN = IFIRST(IPSNS,KP)
23     10     IF (IPEPSN .EQ. NULL) GO TO 100
24     C      SKIP OVER ANY PERSON WHO IS NOT A LEADER
25     C      IF (IVAL(JSURB,IPEPSN) .EQ. NULL) GO TO 90
C      DETERMINE FORCE WHICH LEADER DIRECTS
          IFOFCE = IVAL(IFORCE,IPEPSN)

```

FSN MODEL - LDRPLN SUBROUTINE

TSN

```
C      -- UPDATE TRUE FORCE RECORD IN WHICH
C          THIS PLAYER IS A LEADER
C
26      CONTT=0.0
27      WHOLE=0.0
28      MSITN=1
C
29      ILLST=IVAL(JCONT,IFORCE)
C
C      -- SEARCH LIST OF PLAYERS ON FORCE
C
30      IPRF=IFIRST(ILLST,KA)
31      15     IF(IPRF .EQ. NULL) GO TO 16
32      CONTT=CONTT+1.0
33      IF(IVAL(JPSTAT,IPRF) .GE. KWOUND)
+          WHOLE=WHOLE+1.0
34      MSITN=MAX0(MSITN,IVAL(JSITN,IPRF))
C
C      -- NEXT PLAYER ON FORCE
35      IPRF=NEXT(ILLST,KA)
36      GO TO 15
C
C      -- UPDATE FORCE RECORD
C
37      15 STATUS=0.0
38      NSIDE=IVAL(JFALEG,IFORCE)
39      IF(NSIDE .EQ. KAFF) GO TO 5
40      IF(MSITN .GE. KSITN3) MSITN=MSITN+1
41      IF(MSITN .EQ. KSITN4) IAIMP(1) = 2
42      IF(MSITN .LT. KSITN4) GO TO 5
43      IAIMP(1) = 3
44      IFCOND(1) = 3
45      5   IF(CONTT .LE. 0.0) GO TO 17
        STATUS=WHOLE/CONTT
C
```

FSN MODEL - LDRPEN SUBROUTINE

```

1SN
      C IF STATUS DROPS, DO PLANNING
      C
      47      IF(VAL(JESTAT,IFORCE) .LE. STATUS) GO TO 17
      48      IF(ISIDE .EQ. KSFF) MSITN=KSITNS
      49      IF(ISIDE .EQ. KAFE) MSITN=KSITN4
      50      17 CALL CHGFLD(JESTAT,IFORCE,STATUS,1,IDUM)
      51      IF(MSITN .EQ. IVAL(JSITN,IFORCE)) GO TO 18
      52      CALL CHGFLD(JSITN,IFORCE,MSITN,1,IDUM)

      C DETERMINE PERCEPTIONS OF CURRENT SITUATION OF THIS FORCE
      53      18      CALL PAPSRF(IPERSN, IDUM, IPLAYR, IDUM)
      C GET LEADER'S CURRENT PERCEPTION OF ALL FORCE SITUATIONS
      54      MSITN = IFCPC(IPLAYR)
      55      MSITNF = MSITN
      C UPDATE LEADER'S PERCEPTION OF FORCES
      56      CALL LDPPER(IPERSN,IFORCE,MSITNF)

      C
      C -- TEST IF RESPONSE FORCE SHOULD BE CALLED
      C
      57      MSITNF=IVAL(JSITN,IFORCE)
      58      IF(MSITNF .LT. 31) GO TO 22

      C -- TEST IF RESPONSE FORCE ALREADY CALLED
      C
      59      IF(TMRFSP .GE. 1.E10) TMPESP=TMIN+RESPTM
      C
      C CHECK WHETHER FORCE IS PERCEIVED TO BE ENGAGED. IF SO DO PLANNING.
      60      22      IF (ISIDE .EQ. KAFE .AND. MSITNF .EQ. KSITN) GO TO 30
      61      IF (ISIDE .EQ. KSFF .AND. MSITNF .EQ. KSITNS) GO TO 30
      C ... CHECK FOR COAL COMPLETION
      62      IF(NEWGOL .NE. NULL) GO TO 30
      C CHECK WHETHER FORCE'S PERCEIVED SITUATION HAS CHANGED. IF SO DO
      C PLANNING
      63      IF (MSITN .EQ. MSITNF) GO TO 90
      64      IFCPC(IPLAYR) = MSITNF

```

FSN MODEL - LDRPLN SUBROUTINE

ISN C CALL LEADER'S PLANNING ROUTINE
65 C CALL PLANAC(1,[PERSON])
C
C GET NEXT PERSON
66 90 IPFPSN = NEXT(IPNSNS,KP)
67 GO TO 10
68 100 CONTINUE
69 RETURN
70 END

FSN MODEL - LOS FUNCTION

ISN

```

1      FUNCTION LOS(IR, JR)
C      FUNCTION TO CHECK WHETHER IR AND JR HAVE LINE-OF-SIGHT.
C      IR AND JR MAY BE ENTITIES, LOCATIONS, OR SITE NODES.
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DT_PNAME, FLONAM, FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      DIMENSION IREF(2), A(3,2), IN(2), LOC(2)
9      IREF(1) = IR
10     IREF(2) = JR
11     DO 50 J = 1, 2
12       LOC(J) = -999
13       CALL PARSRF(IREF(J), ID, IDUM, IDUM)
14       IF (ID .EQ. LLOCN) GO TO 30
15       IF (ID .GE. LYARD .AND. ID .LE. LWIND) GO TO 40
16       IF (ID .GE. LVEHIC .AND. ID .LE. LSENS) GO TO 20
17     10   LOS = -1
18     CALL ERR(12, 21, IREF(J), 0, 0)
19     RETURN
C
C ... ENTITY, RETRIEVE LOCATION
20    20   LOC(J) = IVAL(JLOCN,IREF(J))
21    IREF(J) = IPLACE(LOC(J))
22    GO TO 40
C
C ... HAVE LOCATION RECORD
23    30   LOC(J) = IREF(J)
24    IREF(J) = IVAL(JPLACE,IREF(J))
C   FIND THE INTERIOR STATUS
25    40   IN(J) = INTER(IREF(J)) + 1
26   IF (IN(J) .LT. 1) GO TO 10
27   50 CONTINUE
C

```

FSN MODEL - LOS FUNCTION

ISN

```
C      EXAMINE INTERIOR RELATIONS
C
28      IN1 = IN(1)
29      IN2 = IN(2)
30      IFLAG = 0
31      GO TO (60, 70, 80, 90, 90), IN1
32      60      TO (150, 150, 140, 190, 190), IN2
33      70      J TO (150, 150, 110, 100, 100), IN2
34      80      GO TO (140, 110, 120, 120, 120), IN2
35      90      GO TO (190, 120, 120, 120, 120), IN2
36      100     ITEMP = IREF(1)
37      IREF(1) = IREF(2)
38      IREF(2) = ITEMP
39      IN1 = IN2
40      GO TO 120
41      110     IFLAG = 1
42      120     IPORT = 0
43      125     LOS = INTLOS(IREF(1),IREF(2))
44      IF (LOS .EQ. 1) RETURN
45      IF (IN1 .NE. 4) GO TO 130
46      IF (IPORT .EQ. 1) GO TO 130
47      IF (ISEEPT(IREF(1)) .EQ. 0) GO TO 130
48      IREF(1) = IVAL(JPORT,IREF(1))
49      IPORT = 1
50      GO TO 125
51      130     CONTINUE
52      IF (IFLAG .EQ. 0) RETURN
53      140     CONTINUE
54      IPORT = 0
55      GO TO 160
56      150     IPORT = 1
57      160     CONTINUE
58      DO 180 J = 1, 2
59      IF (LOC(J) .EQ. - 999) GO TO 170
60      CALL COORDS(LOC(J), A(1,J), A(2,J), A(3,J))
61      GO TO 180
```

FSN MODEL - LOS FUNCTION

ISN
62 170 A(1,J) = IVAL(JXCO,IREF(J))
63 A(2,J) = IVAL(JYCO,IREF(J))
64 A(3,J) = IVAL(JZCO,IREF(J))
65 180 CONTINUE
66 LOS = LOSXYZ(IPORT,A(1,1),A(1,2)) - 1
67 LOS = MIN0(1,LOS)
68 RETURN
69 190 LOS = 0
70 RETURN
71 END

FSN MODEL - LOSLIS FUNCTION

ISN

```
1      FUNCTION LOSLIS(IREF,I)
C      FUNCTION TO PROVIDE A LIST OF SIGNIFICANT MODEL ENTITIES (PERSONS,
C      VEHICLES, AND SENSORS) HAVING LINE-OF-SIGHT WITH PERSON I.
C
C      IREF    -> REFERENCE TO PLAYER I, THE OBSERVER
C      I       -> INDEX OF OBSERVER
C      LOSLIS  <- REFERENCE TO LIST OF ENTITIES HAVING LOS WITH I
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /RECREF/
9      DIMENSION RECRFQ(140)
10     EQUIVALENCE (RECRFQ(1,IGOALS)
11     LOSLIS =NULL
C
C      ITERATE THROUGH ENTITIES ON THE SITE (GUARDS, ADVERSARIES,
C      VEHICLES, AND SENSORS, RESPECTIVELY)
12     300 CONTINUE
13     ISEN = 0
14     JL = JGARDS
15     305 LISPN = IVAL(JL,ISITE)
16     310 JREF = IFIRST(LISPN,NRENT)
17     320 IF (JREF .NE. NULL) GO TO 390
18     IF (JL .NE. JGARDS) GO TO 330
19     JL = JADVRS
20     GO TO 305
21     330 IF (JL .NE. JADVRS) GO TO 340
22     JL = JVEHS
23     GO TO 305
24     340 IF (JL .NE. JVEHS) RETURN
C      ITERATE THROUGH SENSORS
```

FSN MODEL - LOSLIS FUNCTION

1SN
25 NS = NRECS0(LSENS)
26 IF (NS.LE.0) RETURN
27 ISEN = 1
28 KS = 0
29 JREF = NEWREF(LSENS,0,0)
30 350 IF (KS.GE. NS) RETURN
31 KS = KS + 1
32 JREF = JREF + 1
33 IF (IVAL(JTYPE,JREF).LT. KXPLSN) GO TO 350
34 IF (IVAL(JTYPE,JREF).GT. KCRDSN) GO TO 350
C PROCESS OBSERVABLE ENTITY J
35 390 CONTINUE
C ELIMINATE SELF REFERENCE BY OBSERVER
36 IF (JREF.EQ. IREF) GO TO 490
C ELIMINATE DUPLICATE ENTRIES
37 391 IF (LSERCH(LOSLIS,JREF,NULL,NULL,NULL).NE. NULL) GO TO 490
38 405 L = LOS(IREF,JREF) + 1
39 GO TO (490,410,410),L
C HAVE LOS -- ADD TO LIST
40 410 LOSLIS = ISTACK(JREF,LOSLIS)
C GET NEXT ENTITY
41 490 IF (ISEN.EQ. 1) GO TO 350
42 JREF = NEXT(LISPNT,NRENT)
43 GO TO 320
44 END

FSN MODEL - LOSXYZ FUNCTION

ISN

```
1      FUNCTION LOSXYZ(IPORT,CI,CJ)
C      FUNCTION TO CHECK IF POINTS CI AND CJ HAVE LOS. RETURNS 1 IF
C      NO, 2 IF YES, 3 IF TOO CLOSE.
C      SHOULD NOT BE CALLED FOR LOCATIONS INTERIOR TO BUILDINGS (EXCEPT
C      FOR EXTERIOR PORTALS). IPORT CONTROLS PORTAL STATUS--IF 1,
C      ALL PORTALS ARE CONSIDERED OPAQUE.
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (FAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /DATAV/
9      DIMENSION ACTTIM(25)
10     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
11     COMMON /RECREF/
12     DIMENSION RECRFQ(140)
13     EQUIVALENCE (RECRFQ(1),IGOALS)
14     DIMENSION A(4),B(4),X(2),CI(1),CJ(1)
15     C   IF THE LOCATIONS ARE TOO CLOSE, INDICATE ERROR
16     40 D = ABS(CI(1)-CJ(1)) + ABS(CI(2)-CJ(2)) + ABS(CI(3)-CJ(3))
17     IF (D .GT. .1) GO TO 50
18     LOSXYZ = 3
19     RETURN
20     C   ITERATE THROUGH ALL BARRIERS ON SITE
21     50 LISBAR = IVAL(JCBARS,ISITE)
22     IBAREF = IFIRST(LISBAR,NOBAR)
23     100 IF (IBAREF .EQ. NULL) GO TO 200
24     C   DISCARD TRANSPARENT BARRIERS
25     IF (IVAL(JVIS,IBAREF) .GE. KTRANP) GO TO 190
26     C   GET COORDINATES OF THE PROJECTION OF THE BARRIER ON THE XY PLANE
27     DO 120 K=1,4
28     120 B(K) = VAL(JXCO1+K-1,IBAREF)
29     C   SEE IF LINE SEGMENT DEFINED BY THE LOCATIONS INTERSECTS THE
30     C   PROJECTION OF THE BARRIER
```

FSN MODEL - LOSXYZ FUNCTION

```

1SN
25      A(1) = CI(1)
26      A(2) = CI(2)
27      A(3) = CJ(1)
28      A(4) = CJ(2)
29      INT = INTSCT(A,B,X)
30      C   IF NO INTERSECTION, GET NEXT BARRIER
31          IF (INT .LE. 0) GO TO 190
32      C   HAVE AN INTERSECTION. IF COLLINEAR ASSUME LOS*****
33          C   IF (INT .EQ. 2) GO TO 190
34      C
35          C   THE PROJECTIONS HAVE A UNIQUE INTERSECTION GIVEN IN X(1), X(2).
36          C   USE THE THIRD DIMENSION TO SEE IF THE LINE AND BARRIER ACTUALLY
37          C   INTERSECT.
38      C
39          C   IF THE LINE IS PARALLEL TO THE Y-AXIS, USE THE PROPORTION OF
40          C   THE Y'S INSTEAD OF THE X'S.
41      32      X1 = CI(1)
42      33      X2 = CJ(1)
43      34      X3 = X(1)
44      35      IF (ABS(CI(1)-CJ(1)) .GT. .1) GO TO 150
45      36      X1 = CI(2)
46      37      X2 = CJ(2)
47      38      X3 = X(2)
48      39      150 ZINT = CI(3) + (CJ(3)-CI(3))*(X3-X1)/(X2-X1)
49      40      C   IF THE LINE CLEARS THE BARRIER, GET NEXT BARRIER
50          C   IF (ZINT .GT. VAL(JHIGH,IBAREF)) GO TO 190
51      41      C   LINE INTERSECTS BARRIER. CHECK FOR PORTALS UNLESS EXCLUDED
52          C   IF (IPOBT.EQ.1) GO TO 165
53      42      LISPOR = IVAL(JPORTS,IBAREF)
54      43      C   IF NO PORTALS, NO LOS
55      44      160 IF (LISPOR .NE. NULL) GO TO 170
56      45      165 LOSXYZ = 1
57          RETURN
58      46      C   ITERATE THROUGH PORTALS
59      47      170 IPOREF = IFIRST(LISPOR,NOPOR)
60          175 IF (IPOREF .EQ. NULL) GO TO 165

```

FSN MODEL - LOSXYZ FUNCTION

ISN C DOES THE LOS PROJECTION INTERSECT THE PORTAL PROJECTION?
48 C DH = VAL(JHORIZ,IPOREF)/2.
49 C DV = VAL(JVERT,IPOREF)
50 C ZP = VAL(JZCO,IPOREF)
51 C ZPT = ZP + DV
52 C IF (ZINT.LT.ZP .OR. ZINT.GT.ZPT) GO TO 180
53 C DS = (X(1)-VAL(JXCO,IPOREF))**2 + (X(2)-VAL(JYCO,IPOREF))**2
54 C DH2 = DH*DH
55 C IF (DS .GT. DH2) GO TO 180
C THE LOS INTERSECTS THE PORTAL. IF THE PORTAL IS TRANSPARENT
C LOS IS STILL POSSIBLE.
56 C IF (IVAL(JVIS,IPOREF) .EQ. KTRANP) GO TO 190
C IF THE PORTAL HAS A STATUS OF OPEN OR DESTROYED, LOS POSSIBLE.
57 C IF (IVAL(JSTAT,IPOREF) .LE. KOPEN) GO TO 190
C NO LOS -- BLOCKED BY PORTAL
58 C GO TO 165
59 C 180 IPOREF = NEXT(LISPOR,NOPOR)
60 C GO TO 175
61 C 190 IBAREF = NEXT(LISBAR,NOBAR)
62 C GO TO 100
C NO LOS INTERRUPTION BY BARRIERS; NOW CHECK ROOFS
63 C 200 CONTINUE
C IF NEITHER LOCATION HAS ELEVATION, NOT NECESSARY
64 C IF (ABS(CI(3)-CJ(3)) .LT. .1) GO TO 300
65 C LISBLD = IVAL(JBLDGS,ISITE)
66 C ZMAX = AMAX1(CI(3),CJ(3))
67 C ZMIN = AMIN1(CI(3),CJ(3))
68 C IBLDGR = IFIRST(LISBLD,NOBLDG)
69 C 210 IF (IBLDGE .EQ. NULL) GO TO 300
70 C LISRUF = IVAL(JROOFS,IBLDGR)
71 C 220 IROOF = IFIRST(LISRUF,NOROOF)
72 C 230 IF (IROOF .EQ. NULL) GO TO 270
73 C Z0 = VAL(JZCO,IROOF)
74 C IF (ZMAX .LT. Z0 .OR. ZMIN .GT. Z0) GO TO 250
C THE PLANE OF THE ROOF INTERSECTS THE LOS. DOES THE ROOF ITSELF
C INTERSECT?

FSN MODEL - LOSXYZ FUNCTION

15N

75 CI(1) = X1
76 CJ(1) = X2
77 XC = VAL(JXCO,IROOF)
78 YC = VAL(JYCO,IROOF)
79 DX = VAL(JXLEN,IROOF)/2.
80 DY = VAL(JYLEN,IROOF)/2.
81 SZ = (Z0-CI(3))/(CJ(3)-CI(3))
C GET THE X,Y COORDINATES OF THE INTERSECTION
82 X0 = CI(1) + (CJ(1)-CI(1))*SZ
83 Y0 = CI(2) + (CJ(2)-CI(2))*SZ
C GET THE EXTREMES OF THE ROOF
84 BL = XC - DX
85 BR = XC + DX
86 BD = YC - DY
87 BU = YC + DY
88 IF (X0.LT.BL .OR. X0.GT.BR .OR. Y0.LT.BD .OR. Y0.GT.BU) GO TO 250
C HAVE INTERSECTION, NO LOS
89 LOSXYZ = 1
90 RETURN
C GET NEXT ROOF ON LIST
91 250 IROOF = NEXT(LISRUF,NOROOF)
92 GO TO 230
C GET NEXT BUILDING
93 270 IBLDGR = NEXT(LISBLD,NOBLDG)
94 GO TO 210
C NO LOS INTERRUPTION BY BARRIERS OR ROOFS
95 300 LOSXYZ = 2
96 RETURN
97 END

PSN MODEL - MDESIR SUBROUTINE

ISN

1

SUBROUTINE MDESIR(IPLARF,LISTRF,TEMP)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. EVALUATE THE DESIRABILITY FOR A PLAYER TO MOVE
C ALONG A SPECIFIED PATH.

C

C

C -- INPUT PARAMETERS

C

C

C IPLARF -- PLAYER REFERENCE FOR PLAYER DECIDING TO MOVE

C

C

C LISTRF -- LIST REFERNCE TO PATH (LIST OF NODE
C REFERENCES)

C

C

C -- OUTPUT VARIABLES

C

C

C TEMP -- VALUE OF DESIRABILITY TO BE COMPARED TO
C MOVEMENT THRESHOLD

C

C

C PROCEDURE

C

C 1. SEARCH LIST OF CASE SUBSETS
C AND ADD VALUES TO DESIRABILITY

C

C

2

COMMON /PARS/

3

EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)

4

REAL*8 DTPNAM,FLDNAM,FORMOT

5

COMMON /PARS1/

6

COMMON /PARS2/

7

COMMON /PARS3/

PSN MODEL - MDESIR SUBROUTINE

```
1SN
 8      COMMON /DATAV/
 9      DIMENSION ACTTIM(25)
10      EQUIVALENCE (ACTTIM(1),ACTRAT(1))
C
11      DATA PI/3.14159/
C      -- SEARCH PATH
C
12      TEMP=1.
13      IF(EVALN(20) .LE. 0.) EVALN(20)=1.0
14      INODE=IFIRST(LISTRP,KA)
15      5      IF(INODE .EQ. NULL) GO TO 100
C
C      -- SEARCH PERCEPTIONS TO DETERMINE WHAT IS AT NODE
C
16      TT=0.
17      IPLT=IVAL(JPRCPS,IPLARF)
18      IPER=IFIRST(IPLT,KB)
19      10     IF(IPER .EQ. NULL) GO TO 90
20      IPERRF=IVAL(JVIEW,IPER)
21      CALL PARSRF(IPERRF, IDYY, IRRN, IDDUM)
22      IF(IDYY .EQ. LFORCE) GO TO 80
C      CHECK PERCIEVED PHYSICAL STATUS
C
23      IF(IVAL(JPSTAT,IPERRF) .EQ. KDEAD) GO TO 80
24      IF(IVAL(JPSTAT,IPERRF) .EQ. KCAPTR) GO TO 80
C
25      IND=1
26      IF(IVAL(JALLEG,IPERRF) .EQ. NULL) GO TO 80
27      IF(IVAL(JALLEG,IPERRF) .EQ. IVAL(JALLEG,IPLARF))
*          IND=11
28      IALT=IVAL(JACTIV,IPERRF)
29      IF(IALT .EQ. NULL) GO TO 12
30      IARF=IFIRST(IALT,KC)
31      GO TO 15
32      12     IARF=NULL
C
```

FSN MODEL - MDESIR SUBROUTINE

ISN

```
C -- CASE SUBSET 1
C PERCIEVED LOCATION
C
33    15 ILOC=IVAL(JLOCN,IPERRF)
34      IF(ILOC .EQ. NULL) GO TO 40
35      IF(IVAL(JSOURC,ILOC) .NE. INODE) GO TO 20
36      IF(VAL(JFRAC,ILOC) .NE. 0.) GO TO 20
37      TT=TT+EVALN(IND)
38      IND=IND+2
39      GO TO 40
C
C -- CASE SUBSET 2
C PERCIEVED MOVING TOWARD NODE
C
40    20 IND=IND+1
41      IF(IVAL(JSINK,ILOC) .NE. INODE) GO TO 30
42      TT=TT+EVALN(IND)
43      IND=IND+1
44      GO TO 40
C
C -- CASE SUBSET 3
C MOVING AWAY FROM NODE
C
45    30 IND=IND+1
46      IF(IVAL(JSOURCE,ILOC) .NE. INODE) GO TO 50
47      TT=TT+EVALN(IND)
C
C -- CASE SUBSET 4
C FIRING IN REGION
C
48    40 IND=IND+1
49      IF(IARF .EQ. NULL) GO TO 80
50      IF(IVAL(JTYPE,IARF) .NE. KFIRNG) GO TO 48
51      TT=TT+EVALN(IND)
52      GO TO 80
53    48 IND=IND+1
```

FSN MODEL - MDESIR SUBROUTINE

```

1SN
54      GO TO 60

C
C   -- CASE SUBSET 5
C       FIRING AT REGION
C
55      50 IND=IND+2
56      IF (IARF .EQ. NULL) GO TO 80
57      IF (IVAL (JTYPE,IARF) .NE. KFIRNG) GO TO 60
58      IF (IVAL (JPAR2,IARF) .NE. INODE) GO TO 60
59      TT=TT+EVALN(IND)
60      GO TO 80

C
C   -- CASE SUBSET 6
C       OBSERVING IN AREA
C
61      60 IND=IND+1
62      IF (IVAL (JTYPE,IARF) .NE. KDETNG) GO TO 80
63      ITAR=IVAL (JID,IPER)
64      CALL DIREC (ITAR,INODE,THET,PHE)
65      IF ((THET+PI/4.) .GT. VAL (JPAR1,IARF)) GO TO 80
66      IF ((THET-PI/4.) .LT. VAL (JPAR1,IARF)) GO TO 80
67      TT=TT+EVALN(IND)

C
C   -- NEXT PERCEPTION
C
68      80 IPER=NEXT (IPLT,KB)
69      GO TO 10

C
C   -- NEXT NODE ON LIST
C
70      90 TEMP=TEMP+TT/EVALN(20)
71      INODE=NEXT (LISTRF,KA)
72      GO TO 5

C
73      100 CONTINUE
74      IF (TEMP .GT. 1.) TEMP=1.0

```

PSN MODEL - MDESIR SUBROUTINE

15N
75
76
END

FSN MODEL - NACTSU FUNCTION

ISN

1

FUNCTION NACTSU(IPLARF,ISUB)

C -- THE PURPOSE OF THIS FUNCTION IS TO:
C 1. DETERMINE THE SUB-ACTIVITY OF A PLAYER

C -- INPUT PARAMETERS

C IPLARF - PLAYER REFERENCE
C ISUB -- ACTIVITY DESIRED

C -- OUTPUT

C THE FUNCTION RETURNS THE ACTIVITY SUBTYPE VALUE

C -- PROCEDURE

C 1. FIND PLAYER ACTIVITY, IF NULL, NO ACTIVITY SUBTYPE

C 2. DETERMINE SUBACTIVITY BASED ON ACTIVITY

2

C COMMON /PARS/

3

EQUIVALENCE (NULL,NULL), (FAIL,FATL)

4

REAL*8 DTPNAM,FLDNAM,FOPNOT

5

COMMON /PARS1/

6

COMMON /PARS2/

C

C -- FIND ACTIVITY

C

7

IACLT=IVAL(JACTIV,IPLARF)

C

8

JACTRF=IFIRST(IACLT,KAI)

9

IF(JACTRF .EQ. NULL) GO TO 10

FSN MODEL - NACTSU FUNCTION

15N
16 IT=IVAL(JTYPE,IACTR)
17 IF(IT .EQ. KMOVNG) GO TO 20
18 IF(IT .EQ. KFIRNG) GO TO 30
19 IF(IT .EQ. KDETNG) GO TO 40
20 IF(IT .EQ. KCAPNG) GO TO 45
C
21 10 TACTS=KREST
22 GO TO 50
C
23 -- MOVING ACTIVITY
C DEFAULT SUB ACTIVITY IS RUNNING
C
24 20 TACTS=KRUN
25 GO TO 50
C
26 -- FIRING ACTIVITY
C DETERMINE SUB-ACTIVITY FROM WEAPON SELECTED
C
27 30 IWPF=IVAL(JPAR3,IACTR)
28 IWPN=IVAL(JTYPE,IWPF)
C
29 -- FIND SUBACTIVITY
C
30 IF(IWPN .EQ. KPISTL) TACTS=KBRRBLD
31 IF(IWPN .EQ. KPIFL) TACTS=KFIPIF
32 IF(IWPN .EQ. KMGRUN) TACTS=KFIRMG
33 IF(IWPN .EQ. KLAWE) TACTS=KFILAW
34 IF(IWPN .EQ. KGRENAD) TACTS=KFIGRN
35 GO TO 50
C
36 -- OBSERVING ACTIVITY
C DEFAULT SUB ACTIVITY IS STATIONARY
C
37 40 TACTS=KDETST
38 GO TO 50
C

FSN MODEL - NACTSU FUNCTION

1SN
C -- CAPTURING
C
29 IACTS=KCPTFG
C
30 IF(IISUB .EQ. NULL) GO TO 100
31 IF(IISUB .EQ. ITI) GO TO 100
32 IF(IISUB .EQ. KMOVNG) GO TO 60
33 IF(IISUB .EQ. KDETNC) GO TO 70
34 IF(IISUB .EQ. KCAPNG) GO TO 80
35 GO TO 100
C
36 IACTS=ARUN
37 GO TO 100
C
38 IACTS=KDETST
39 IF(IT .EQ. KMOVNG) IACTS=KDETMV
40 IF(IT .EQ. KFIRNG) IACTS=KDETFI
41 GO TO 100
C
42 IACTS=KCPTRG
C
43 100 NACTSU=IACTS
44 RETURN
45 END

FSN MODEL - NBRREG FUNCTION

ISN

```
1      FUNCTION NBRREG(IPL)
C      FUNCTION TO SEARCH THROUGH THE NEIGHBORS OF A NODE IPL
C      AND RETURN THE REGION REFERENCE IF A ROOM OR HALL, OR THE
C      NEGATIVE OF THE REGION REFERENCE IF A YARD OR ROOF, OR ZERO
C      IF NO REGION NEIGHBOR.
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      NBRREG = 0
9      NBRLIS = IVAL(JNBRS,IPL)
10     NBR = IFIRST(NBRLIS,INBR)
11     IF (NBR.EQ.NULL) RETURN
12     CALL PARSRF(NBR,IDLTYPE,IDLUM,IDLUM)
13     IF (IDLTYPE.NE.LROOM .AND. IDLTYPE.NE.LHALL) GO TO 150
14     C      HAVE FOUND ROOM OR HALL NEIGHBOR
15     NBRREG = NBR
16     RETURN
16     150    IF (IDLTYPE.NE.LYARD .AND. IDLTYPE.NE.LROOF) GO TO 200
17     C      HAVE FOUND YARD OR ROOF NEIGHBOR
18     NBRREG = -NBR
19     RETURN
19     200    NBR = NEXT(NBRLIS,INBR)
20     GO TO 100
21     FND
```

FSN MODEL - NDRFIR FUNCTION

ISN

1 FUNCTION NDRFIR(IPLARF,ITRG,IACTR,IMNT)
C
C -- THE PURPOSE OF THE FUNCTION IS TO:
C 1. DETERMINE IF IPLARF WILL FIRE
C AT ITRG
C
C
C -- INPUT PARAMETERS
C
C IPLARF -- PLAYER REFERENCE
C ITRC -- TARGET REFERENCE
C IACTR -- ACTIVITY REFERENCE
C IMNT -- FIRE ASSESSMENT
C
C
C -- OUTPUT
C
C THIS FUCTION RETURNS EITHER A 0 FOR UNACCEPABLE
C OR A 1 FOR ACCEPABLE TO FIRE
C
C -- PROCEDURE
C 1. DETERMINE IF TIME REMAINS FOR IPLARF TO FIRE
C 2. DETERMINE IF AMMUNITION AVAILABLE
C
C 3. DETERMINE IF TERMINATION RELATION IS SATISFIED
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PAKS1/
9 COMMON /DATAV/
10 DIMENSION ACTIIM(25)

FSN MODEL - NDRFIR FUNCTION

1 ISN
2 11 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
3 C
4 C -- DETERMINE IF PLAYER TIME REMAINS
5 C
6 12 ISUBA=NACTSU(IPLARF,NULL)
7 13 IF (VAL(J1REQ,IPLARF) .LT. ACTTIM(ISUBA)) GO TO 50
8 C
9 C -- TIME REMAINS
10 C DETERMINE IF AMMUNITION AVAILABLE
11 C
12 14 IWPNRF=IVAL(JPAR3,IACTRFF)
13 15 IF (IVAL(JAMT,IWPNRF) .LE. 0) GO TO 50
14 C
15 C -- TEST IS AREA FIRE WEAPON
16 C DO NOT FIRE AGAIN
17 C
18 16 IWPN=IVAL(JTYPE,IWPNRF)
19 17 IF (IFIRTP(IWPN) .EQ. 2) GO TO 50
20 C
21 C -- TEST FOR RELATION BEING SATISFIED
22 C
23 18 IRLT=IVAL(JMODS,IACTRFF)
24 19 IRF=IFIRST(IRLT,KL)
25 20 30 IF (IRF .EQ. NULL) GO TO 60
26 C
27 C -- TEST RELATION FOR TERMINATION
28 C
29 C
30 C
31 21 IC=NRTEST(IRF)
32 22 IF (IC .EQ. 1) GO TO 50
33 C
34 C -- DECIDES TO FIRE AGAIN
35 C
36 23 NDRFIR=1
37 24 IRF=NEXT(IRLT,KL)

FSN MODEL - NDRFIR FUNCTION

1SN
25 GO TO 30
C
C -- DO NOT FIRE AGAIN
C
26 50 NDRFIR=0
C
27 60 CONTINUE
28 RETURN
29 END

FSN MODEL - NEWLDR FUNCTION

ISN

```
1      FUNCTION NEWLDR(ILLEAD,IFCRCE)
C
C   -- THE PURPOSE OF THIS FUNCTION IS TO:
C       1. FIND A NEW LEADER FOR THE FORCE
C
C   -- INPUT PARAMETERS
C
C       ILLEAD -- OLD LEADER
C
C       IFORCE -- REFERENCE TO FORCE RECORD
C
C
2     COMMON /STATEV/
3     DIMENSION ITEM(41900),DITEM(41900)
4     EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5     COMMON /PARS/
6     EQUIVALENCE (FNULL,NULL),(IPAIL,FAIL)
7     REAL*8 DTPNAM,FLDNAM,FORMOT
8     COMMON /PARS1/
9     COMMON /PARS2/
10    COMMON /PARS3/
C
C   RULE: FIRST SUBORDINATE OF LEADER IS NEW LEADER
C
11    ISUPLT=IVAL(JSUBS,ILLEAD)
12    ITEMP=IPIRST1(ISUPLT,KA)
13    10 IF(ITEMP .EQ. NULL) GO TO 30
14    IF(IVAL(JPSTAT,ITEMP) .GT. KCAPTH) GO TO 25
15    ITEMP=NEXT(ISUPLT,KA)
16    GC TO 10
C
17    25 NEWLDR=ITEMP
```

FSN MODEL - NEWLDB FUNCTION

ISN
18 GO TO 50
19 30 NEWLDB=NULL
C
20 50 CONTINUE
21 RETURN
22 END

FSN MODEL - NEWPCP SUBROUTINE

ISN

```
1      SUBROUTINE NEWPCP(IMSGRF,IPLARF)
C ... THE PURPOSE OF THIS ROUTINE IS TO UPDATE A PLAYER'S
C PERCEPTIONS OF AN OBJECT BASED ON A MESSAGE ABOUT
C THE OBJECT. IT IS ASSUMED THAT THE ROUTINE IS
C CALLED ONLY WHEN IT HAS BEEN DETERMINED THAT THE
C PERCEPTION SHOULD BE UPDATED. CURRENTLY A PLAYER
C MAY HAVE PERCEPTIONS OF VEHICLES, PERSONS AND
C SENSORS.
C
C INPUT PARAMETERS:
C     IMSGRF ... POINTS TO MESSAGE
C     IPLARF ... POINTS TO PLAYER
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCRECT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
C
C ... DETERMINE TYPE OF OBJECT OBSERVED, RETURN
C IF NOT VEHICLE, PERSON OR SENSOR
C
11    IOBSRF=IVAL(JSUBJ,IMSGRF)
12    CALL PARSRF(IOBSRF,IDIYP,IREC,IFLD)
13    IATTR=IVAL(JATTR,IMSGRF)
14    IF(IDTYP.LT.1.VEHIC .OR. IDTYP.GT.1SENS) GO TO 60
C
C ... DETERMINE IF PLAYER CURRENTLY HAS PERCEPTION
C OF THIS OBJECT
C
15    LISPCE=IVAL(JPCES,IPLARF)
```

FSN MODEL - NEWPCP SUBROUTINE

ISM
16 IPCPRF=LSEARCH(LISPCP,NULL,NULL,JID,IOBSRF)
17 IF(IPCPRF.NE.NULL) GC TO 5
C
C ... CREATE A NEW PERCEPTION
C
18 IVWREF=NEWREF(IDTYP,BEUREC(IDTYP),0)
19 IPCPRF = NEWRC4(LPERCP,IOESRF,IVWREF,NULL,TMIN)
C
C ... ADD TO LIST OF PLAYER'S PERCEPTIONS
C
20 LISPCP=ISTACK(IPCPRF,LISPCP)
21 CALL CHGPLD(JPBCPS,IELABF,LISPCP,1,IDUM)
22 GO TO 7
C
C ... PERCEPTION EXISTS AND NEEDS TO BE UPDATED
C POSSIBILITIES ARE:
C 1. MAKE COPY OF MESSAGE CONTENT AND
C REPLACE FIELD IN VIEW RECORD WITH
C REFERENCE (JLCCN OR JACTIV)
C 2. REPLACE FIELD IN VIEW RECORD WITH
C MESSAGE CONTENT (ALL SCALARS, JDRIVER
C CR JIEADR)
C 3. ADD MESSAGE CCNTENT TO LIST IN VIEW
C RECORD. (ALL OTHER ATTRIBUTES)
C
23 5 IVWREF=IVAL(JVIEW,IPCPBF)
24 CALL CHGFLD(JTIME,IPCPBF,TMIN,1,IDUM)
25 7 NBASE=IREP1(NEWREF(IDTYP,BEUCS(IDTYP),0))
26 ITYP=ITEM(NBASE+IATTR)
27 IF(ITYP.GE.3) GC TC 20
C
C ... REPLACE FIELD IN VIEW RECOED WITH MESSAGE CONTENT
C
28 10 CALL CHGFLD(IATTR,IVWREF,IVAI(JCONT,IMSGRF),1,NULL)
29 GC TC 50
C

FSN MODEL - NEWPCP SUBROUTINE

ISM

```
C ... NON-SCALAR FIELD, DETERMINE APPROPRIATE ACTION
C
30  20      IF (IDTYP.EQ.LVEHIC .AND. IATTR.EQ.JDRIVR) GO TO 10
31      IF (IDTYP.EQ.LPEHSK .AND. IATTR.EQ.JLEADR) GO TO 10
32      IF (IATTR.EQ.JLCCN) GO TO 40
33      IF (IDTYP.EQ.LPEESK .AND. IATTR.EQ.JACTIV) GO TO 40
C
C ... IF NOT LIST ADE CCNTENT TO LIST IN VIEW RECORD
C
34      ICONT=IVAL(JCCNT,IMSGRF)
35      CALL FARSREF(ICONT,IE,IR,IF)
36      IF (IE .EQ. 1LIST) GC TO 30
37      ILIS=IVAL(IATTR,IVWREF)
38      ILIS=ISTACK(IVAL(JCCNT,IMSGRF),ILIS)
39      CALL CHGFLD(IATTR,IVWREF,ILIS,1,NULL)
40      GO TO 50
C
C ... CONTENT IS LIST, ADD FIRST "NEW" ITEM
C
41  30      ILIS = IVAL(IATTR,IVWREF)
42      IREF = IPIRST(ICONT,NP)
43  32      IF (IREF .EQ. NULI) GC TO 50
44          ION = LSERCH(ILIS,IREF,NULL,NULL,NULL)
45          IF (ION .NE. NULL) GC TO 35
C
C ... NEW ITEM FOUND, ADD TO LIST IN IVWREF
C
46          CALL IQUEUE (IREF,ILIS)
47          GO TO 50
48  35      IREF = NEXT(ICONT,NP)
49          GC TO 32
C
C ... COPY MESSAGE CONTENT, STORE REFERENCE IN
C     APPROPRIATE FIELD OF VIEW RECORD
C
50  40      IRREF=ICOPY(IVAL(JCCNT,IMSGRF))
```

FSN FCDEL - NEWPCP SUBROUTINE

ISN

51 CALL CHGFLD(IATTR,IVWREF,IRREF,1,NULL)

C

C ... UPDATE SITUATION OF IPLARF IF PERCEPTION IS OF ENEMY

C

52 50 IF(IVAL(JSITN,IPLARF) .GE. KSITN3) GO TO 60

53 IF(IDTYP .NE. IFERSN) GO TO 60

54 IF(IVAL(JALLEG,IPLARF) .EQ. IVAL(JALLEG,IOBSRF)) GO TO 60

C

C ... IF SOURCE OF INFO IS IPLARF, SITN=KSITN3

C IF SOURCE OF INFO NOT IPLARF, SITN=KSITN2

55 MSITN=KSITN2

56 IF(IVAL(JSOURC,IMSGRF) .EQ. IPLARF) MSITN=KSITN3

57 IF(IVAL(JSITN,IPLARF) .GE. MSITN) GO TO 60

58 CALL CHGFLD(JSITE,IPLARF,MSITN,1,NULL)

59 60 RETURN

60 END

FSN MCDEL - NWPNNS SUBROUTINE

158

```
1      SUBROUTINE NWPNNS(IPLARF,ITRG,IWPNRF,NTYP)

C      -- THE PURPOSE OF THIS SUBROUTINE IS TO:
C          1. SELECT THE MOST DESIRABLE WEAPON THAT
C              IS CARRIED BY PLAYER IPLARF TO EMPLOY
C                  AGAINST TARGET TYPE ITRG

C      -- INPUT PARAMETERS
C
C          IPLARF - PLAYER REFERENCE
C
C          ITRG - TARGET REFERENCE

C      -- OUTPUT VARIABLES
C
C          NTYP -- THE WEAPON TYPE SELECTED
C
C          IWPNRF -- WEAPON RECORD REFERENCE

C      -- PROCEDURE
C
C          1. DETERMINE RANGE OF TARGET
C
C          2. SEARCH LIST OF WEAPONS CARRIED BY PLAYER
C              -DETERMINE IF TARGET WITHIN RANGE
C              - SELECT MOST DESIRABLE WITHIN RANGE

C
C          COMMON /PARS/
C          EQUIVALENCE (NULL,NULL), (IPFAIL,FAIL)
C          REAL*8 DTPNAM,FIDNAM,FCBMCT
C          COMMON /PARS1/
C          COMMON /PARS3/
```

FSK MCDEL - NWPN SUBROUTINE

```
1 ISN
2
3
4
5
6
7      COMMON /DATAV/
8      DIMENSION ACTTIM(25)
9      EQUIVALENCE (ACTTIM(1),ACTRAT(1))
C
C -- DETERMINE TARGET TYPE
C
10     CALL FARSRF(ITRG, IDT, IEUP, IDUM)
11     I = IVAL(JTYEE, ITRG)
12     IF(IDT .EQ. LVEHIC) ITART=IARMTP(I)
13     IF(IDT .EQ. LPESNM) ITART=KTGPSN
14     IF(IDT .EQ. LSENS) ITART=KTGSEN
C
C -- DETERMINE DISTANCE OF TARGET
C
15     DT=DIST(IPLARF,ITRG)
C
C -- SEARCH WEAPON LIST FOR FIAYER
C
16     IWPLT=IVAL(JWEAES,IPLARF)
17     NTYP=0
18     IRNK=99999
19     IW=IFIRST(IWPLT,KA)
20     5   IF(IW .EQ. NULL) GO TO 20
C
C -- IGN CBE EQUIPMENT
C
21     CALL PARSRF(IW, IDT, IDUM, IDU)
22     IF(IDT .NE. IWEAE) GO TO 10
23     ITYPE=IVAL(JTYPE,IW)
24     IF(DT .LT. RNGMIN(ITYP)) GO TO 10
25     IF(DT .GT. RNGMAX(ITYP)) GO TO 10
C
C -- DETERMINE IF RANK OF WEAEON GREATER THAN OTHERS
C
26     IF(IRNKWP(ITYP,ITART) .LE. 0) GO TO 10
27     IF(IRNKWP(ITYPE,ITART) .GE. IRNK) GO TO 10
```

FSK MCDEL - NWPNS SUBROUTINE

ISN

C

28 NTYPE=ITYP
29 IWENRF=IW
30 IRNK=IBNKWP(ITYP,ITART)

C

31 10 IW=NEXT(IWPIIT,KA)
32 GO TO 5

C

33 20 CCNTINUE
34 RETURN
35 END

FSN MODEL - OBS SUBROUTINE

ISN

```
1      SUBROUTINE OBS
C
C      ROUTINE TO CONDUCT OBSERVATION ACTIVITIES BY
C      PERSONS -- SPECIFICALLY THOSE WHICH HAVE OBSERVATION
C      ORDERED AS AN ACTIVITY BY THE DECISION MODULE.  THE
C      ROUTINE ALSO CONDUCTS A CHECK AT THE END OF EACH TIME STEP
C      TO ALLOW TARGETS OF FIRE A CHANCE TO SEE THE FIRER.
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IPFAIL,FAIL)
7      REAL*8 DTPNAM,FIDNAME,FCFFECT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /DATAV/
12     DIMENSION ACTTIM(25)
13     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14     COMMON /RECREF/
15     DIMENSION RECBFQ(140)
16     EQUIVALENCE (RECRFQ(1),IGOALS)
C
C      ITERATE THROUGH PERSONS ON SITE - GUARDS AND ADVERSARIES
17     JSIDE=JGARDS
18     50 CALL CHGVAR(4,JSIDE-JGARDS+1)
19     MPNLIS = IVAL(JSIDE,ISITE)
C      GET FIRST PERSON ON LIST
20     MPNREF = IPIRST(MPNLIS,ICURNO)
21     60 IF(MPNREF .EQ. NULL) GO TO 300
22     CALL FARSBF(MPNREF,IDUM,IELAYR,IDUM)
23     CALL CHGVAR(3,IPLAYR)
C      GET ACTION RECORD OF THIS PERSON TO SEE IF HE IS OBSERVING;
C      IF NOT, CHECK TIME REMAINING
```

FSN MODEL - OBS SUBROUTINE

ISN
24 MACIT = IVAL(JACTIV,MPNREF)
25 IF(MACLT .EQ. NULL) GO TO 65
26 MACT=IFIRST(MACIT,KK)
27 MTYEA = IVAL(JTYPE,MACT)
28 TR=DTSEC
29 ANGLE = VAL(JPART1,MACT)
30 IF (MTYPA .EQ. KDETNG) GO TO 70
31 65 TR = VAL(JTREQ,MPNREF)
32 IF (TR.LE.0) GO TO 100
33 ANGLE = PNULL
C PERSON IS OBSERVING FOR THE WHOLE TIME INTERVAL - CALL ROUTINE
C TO PERFCRM THIS ACTIVITY
34 70 CALL CBSERV(MPNREF,TR,KDETNG,ANGLE)
C IF PLAYER IS THE SENSOR MONITOR, CHECK SENSOR ACTIVATIONS
35 IF (IVAL(JTYEE,MPNREF) .EQ. KSNMON) CALL SENSE(M2NREF)
C
C PERFORM END-OF-CYCLE FIRING CHECK
C
C SEE IF CURRENT PEBSCN IS FIRING - IF NOT, DONE WITH HIM
36 100 IF(MTYPA .NE. KFIRNG) GO TO 200
C GET REFERENCE TO TARGET OF FIRER
37 MTGTRF = IVAL(JPART1,MACT)
C CHECK THAT TARGET IS A PEBSCN; IF NOT, DONE
38 CALL FAESRF(MTGTRF,IE,IDUM,IDUM)
39 IF (ID .NE. 1PEFSN) GO TO 200
C SEE IF FIRER IS ALREADY ON OBSERVATION LIST OF TARGET
40 IF(LSEARCH(1OESV(IPLAYR),MIGTRF,NULL,NULL,NULL) .NE. NULL) GO TO 200
C IF NOT, GIVE TARGET A CHANCE TO SEE FIRER
41 CALL CBSERV(MTGTRF,0.,KFIREP,PNULL)
C GET NEXT PERSON
42 200 MPNREF = NEXT(MPNLIS,ICUENO)
43 GO TO 60
44 300 CONTINUE
45 IF (JSIDE .EQ. JADVRS) GO TO 350
46 JSIDE=JADVRS
47 GO TO 50

FSN MODEL - OBS SUBROUTINE

ISN
48 350 CONTINUE
49 RETURN
50 END

PSN MODEL - OBSERV SUBROUTINE

ISN

```
1      SUBROUTINE OSEBVR(IREF,T,JAOF,THETO)
C
C      SUBROUTINE TO CREATE AN OBSERVATION LIST OF PHYSICAL OBJECTS
C      SEEN BY A PERSON IN THE MODEL. CALLED IN TWO MODES:
C          (1) T > 0 FOR NORMAL OBSERVATION ACTIVITY
C          (2) T <= 0 FOR END-OF-CYCLE FIRE CHECK
C
C      IREF    -> RECORD REFERENCE TO OBSERVER
C      T       -> TIME REMAINING FOR OBSERVATION
C      JAOF    -> CODE FOR CURRENT ACTIVITY OF OBSERVER IF T>0;
C                    REFERENCE TO FIRER IF T<=0
C      THETA   -> PRINCIPAL HORIZONTAL ANGLE OF OBSERVATION OF
C                    OBSERVER; IF NULL, UNIFORM SEARCH IS ASSUMED.
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULI),(IPFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FCEMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /DATAV/
9      DIMENSION ACTTIE(25)
10     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
11     COMMON /RECREF/
12     DIMENSION RECBEQ(140)
13     EQUIVALENCE (RECBEQ(1),IGCALS)
14     COMMON/OCTANT(8)
15     C      CHECK THAT IREF IS A PERSON REFERENCE
16     CALL FARSBP(IREF,1D,I,1DCM)
17     IF( ID .EQ. IPERSN) GO TO 50
18     CALL ERR(3,19,IPERSN,IREF,0)
      RETURN
C
C      IS I SUPPRESSED IN OBSERVATION?
```

FSN MODEL - OBSERV SUBROUTINE

15N

```
19      50 IF (ICANOB(IREF) .NE. ICK) RETURN
20          THETA=THETO
21          C   CHECK FOR END-OF-CYCLE CONDITION
22          IF (T .LE. 0) GO TO 500
23          C   DOES ENOUGH TIME REMAIN FOR OBSERVATION?
24          IA = NACTSU(IREF,KDETNG)
25          IF (T .LT. ACTTIM(IA)) RETURN
26          C   PRODUCE ICS LIST FOR I
27          ITLOS = LOSLIS(IREF,I)
28          C   IF AN ERROR IN LCSLIS OR NC LOS LIST, DONE
29          IF (ITLOS .EQ. NULL .OR. ITLOS .EQ. -1) RETURN
30          C   FIND THE PROBABILITY E(K) OF I LOOKING IN OCTANT K
31          CALL FOCT(IREF,THETA)
32          C   ITERATE THROUGH THE LOS LIST FOR I
33          JREF = IFIRST(ITLOS,JNC)
34          C   IF NC MORE PLAYERS, DONE
35          150 IF (JREF .EQ. NULL) RETURN
36          C   IF ALREADY ON OBSERVED LIST, SKIP IT
37          IF (ISERCH(IOBSV(I),JREF,NULL,NULL,NULL) .NE. NULL) GO TO 200
38          C   DETERMINE PROBABILITY OF OBSERVATION
39          P = PCES(IREF,JREF,T)
40          RN = URAND(IRN(1))
41          IF (RN .GT. P) GO TO 200
42          C   ADD J TO LIST OF SIGHTINGS FOR I
43          IOBSV(I) = ISTACK(JREF,ICESV(I))
44          C   GET NEXT PLAYER
45          200 JREF = NEXT(ITLOS,JNC)
46          GO TO 150
47          500 CCNTINUE
48          C   END-OF-CYCLE CHECK FOR BEING UNDER FIRE
49          C   IN THIS MODE THE FIRER IS THE THIRD CALLING PARAMETER
50          JREF = JAOF
51          C   IF FIRER ALREADY ON I'S OBSERVATION LIST, DONE
52          IF (ISERCH(IOBSV(I),JREF,NULL,NULL,NULL) .NE. NULL) RETURN
```

FSN MODEL - OBSERV SUBROUTINE

ISN

C VERIFY THAT JREF IS A PERSON REFERENCE
39 CALL PARSRF(JREF, IDJ, IDUP, IDUM)
40 IF (IDJ .EQ. 1) GO TO 520
41 RETURN

C DETERMINE I'S PRINCIPAL DIRECTION OF OBSERVATION FROM I'S ACTIVITY
42 520 IACREF = IVAL(JACTIV,IREF)
C IF I IS MOVING, FIND I'S PDC FROM I'S LOCATION
43 IV = IVAL(JTYPE,IACREF)
44 IF (IV .NE. KMOVNG) GO TO 550
45 LCCREF = IVAL(JLOCN,IREF)
46 CALL DIREC (IVAL(JSCURC,LCCREF),IVAL(JSINK,LOCREF),THETA,PHI)
47 GO TO 600

C

C IF I IS OBSERVING, FIND I'S PDO FROM I'S ACTION RECORD
48 550 IF (IV .NE. KDETNG) GO TO 570
49 THETA = VAL(JPAR1,IACREF)
50 GO TO 600

C IF I IS FIRING, FIND I'S TARGET AND THEN THE ANGLE TO THE TARGET
51 570 IF (IV .NE. KFIRNG) GO TO 590
52 ITARG = IVAL(JFAR1,IACREF)
53 CALL DIREC(IREF,ITARG,TEETA,PHI)
54 GO TO 600

C IF NONE OF THE ABOVE APPLIES, NULL THE PDO
55 590 THETA = FNULL

C

C FIND THE PROBABILITY THAT I SEES FIRERS, USING THE WHOLE TIME STEP
56 600 CONTINUE
57 CALL ECCT(IREF,THETA)
58 P = POBS(IREF,JREF,DSEC)
59 RN = URAND(IRN(1))

C IF FIRER HAS NOT BEEN SEEN, DONE
60 IF (RN .GT. P) RETURN

C FIRER HAS BEEN SEEN. ADD TO I'S OBSERVATION LIST
61 CALL CHGLST(JREF,1,3,I)
62 IOBSV(I) = ISTACK(JREF,ICBSV(I))
63 RETURN

64 END

FSN MODEL - ODESIR SUBROUTINE

ISM

1

SUBROUTINE ODESIR(IPLAFF,ITRG,ICODE)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. EVALUATE TARGET LIST TO DETERMINE
C WHICH TARGET IS MOST DESIRABLE TO
C OBSERVE.

C 2. IF TARGET LIST IS NULL, RETURN NULL VALUE.

C

C

C -- INPUT PARAMETERS

C

C

C IPRAFF -- PLAYER REFERENCE OF PLAYER TO OBSERVE

C

C

C ITRG -- LIST REFERENCE TO TARGET LIST

C

C

C -- OUTPUT VARIABLES

C

C

C ICODE -- TARGET LOCATION REFERENCE FOR OBSERVATION

C DIRECTION CALCULATION.

C THIS IS NULL WHEN TARGET LIST IS NULL.

C

C

C -- PROCEDURE

C 1. SEARCH CASE SUBSETS TO DETERMINE
C MOST DESIRABLE TARGET FOR OBSERVATION

C

C

2

COMMON /PARS/

3

EQUIVALENCE (FNULL,NULL),(IPFAIL,FAIL)

4

REAL*8 DTPNAM,FLDNAM,FCRMCT

FSN MODEL - ODESIR SUBROUTINE

```
1SN
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /DATAV/
9      DIMENSION ACTITM(25)
10     EQUIVALENCE (ACTITM(1),ACTRAT(1))

C
C
11    ICODE=NULL
12    TMAX=-9999.
13    IPLT=IVAL(JPBCES,IPLABF)

C
C -- SEARCH TARGET IIS1
C
14    ITT=IFIRST(ITRG,KE)
15    5      IF(ITT .EQ. NULL) GO TO 30

C
C SEARCH PERCEPTIONS LIST
C
16    IPEB=IFIRST(IPLT,KC)
17    8      IF(IPEB .EQ. NULL) GO TO 20
18    IF(IVAL(JID,IPEB) .EQ. ITT) GO TO 10
19    9 IPEB=NEXT(IPLT,KC)
20    GC TO 8

C
C -- CASE SUBSET 1
C       ALLEGIENCE KNOWN
C
21    10 IPERRF=IVAL(JVIEW,IPEF)
22          CALL PARSBF(IPERRF, IDYY, IRRN, ISDUM)
23          IF(IDYY .EQ. IFCFCE) GO TO 9
24          TME=0.
25          IF(IVAL(JALLEG,IPERBF) .EQ. NULL) TEMP=TEMP+EVALN(41)

C
C -- CASE SUBSET 2
C       LOCATION KNOWN
```

FSN MODEL - ODESIR SUBROUTINE

ISN

C

26 IF(IVAL(JLOCN,IPERRP) .EQ. NULL) TEMP=TEMP+EVALN(42)

C

C -- CASE SUBSET 3

C PHYSICAL STATUS KNOWN

C

27 IF(IVAL(JPSTAT,IPERRP) .EQ. NULL) TEMP=TEMP+EVALN(43)

C

C -- CASE SUBSET 4

C ACTIVITY KNOWN

C

28 IF(IVAL(JACTIV,IPERRP) .EQ. NULL) TEMP=TEMP+EVALN(44)

C

C -- CASE SUBSET 5

C EQUIPMENT AND WEAPONS KNOWN

C

29 IF(IVAL(JWEAPS,IPERRP) .EQ. NULL) TEMP=TEMP+EVALN(45)

C

30 IF(TEMP .LE. TMAX) GC TO 20

31 TMAX=TEMP

32 ICCDE=ITT

33 20 ITT=NEXT(ITRG,KE)

34 GC TO 5

C

35 30 CCNTINUE

36 RETURN

37 END

FSN MODEL - PEQUAL FUNCTION

1 FUNCTION PEQUAL(IPCPRF,IMSGRF)
C ... FUNCTION CALLED BY CCMEEE TO DETERMINE IF MESSAGE
C CONTENT IS EQUIVALENT TO PLAYER'S CURRENT PERCEPTION
C OF THE SUBJECT OF THE MESSAGE.
C ... INPUT PARAMETERS:
C IPCPRF ... ECINTEE TO PERCEPTION RECORD
C IMSGRF ... POINTER TO MESSAGE RECORD
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DITEM,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCRECT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 LOGICAL PEQUAL
C
C ... SET UP FOR PROCESSING
C
11 PEQUAL=.TRUE.
12 IVWREF=IVAL(JVIEW,IPCPRF)
13 ICBSRF=IVAL(JSUBJ,IMSGRF)
14 CALL PARSRF(IOESRF,ITYPE,IRECNO,IFLD)
15 IATTR=IVAL(JATTR,IMSGRF)
C
C ... IF NO PERCEPTION OF THIS ATTRIBUTE RETURN .FALSE.
C
16 IF(IVAL(IATTR,IVWREF).EQ.NULL) GO TO 50
C
C GET PCINTER TO GENERIC RECORD TO GET ATTRIBUTE TYPE
C
17 NBASE=IREFI(NEWREF(IDTYPE,NRECS(IDTYPE),0))
18 ITYP=ITEM(NBASE+IATTR)
19 IF(ITYP.GE.3) GO TO 20

PSN MODEL - PEQUAL FUNCTION

ISN

```

C
C ... ATTRIBUTE IS SCALAR OR SINGLE ENTRY FIELD
C     AND DIRECT COMPARISON IS APPROPRIATE
C
20   10    IF(IVAL(IATTR,IVWREF) .EQ. IVAL(JCONT,IMSGRF)) RETURN
21
22   20    GC TC 50
C
C ... NOT SCALAR, DETERMINE PROCESSING REQUIRED
C
23   20    IF(IDTYP.EQ.IVHEIC .AND. IATTR.EQ.JDRIVE) GO TO 10
24    IF(IDTYP.EQ.LPERSN .AND. IATTR.EQ.JLEADR) GO TO 10
25    IF(IATTR.EQ.JLCCN) GC TC 40
26    IF(IATTR.EQ.JACTIV .AND. IDTYP.EQ.LPERSN) GO TO 30
C
C ... SIMPLE LIST SEARCH REQUIRED
C
27    LISBEF=IVAL(IATTR,IVWREF)
28    ION=LSEARCH(LISBEF,IVAL(JCCNT,IMSGRF),NULL,NULL,NULL)
29    IF(ION.NE.NULL) RETURN
      GO TO 50
C
C ... MESSAGE SUBJECT IS PERSON'S ACTIVITY
C     SPECIAL SEARCH REQUIRED.
30   30    LISREF=IVAL(IATTR,IVWBKF)
31    IACTBF=IVAL(JCCNT,IMSGRF)
32    IACTYP=IVAL(JTYEE,IACTBF)
33    MACTRF=IFIRST(LISREF,LPTB)
34   35    IF(MACTRF.EQ.NULL) GC TC 50
35    MACTYP=IVAL(JTYPE,MACTR)
36    IF(MACTYP.NE.IACTYP) GO TO 38
C
C     ACTIVITY TYPE MATCH, COMPARE PARAMETERS
C
37    DO 37 I=JPAB1,JDOER
38    IF(IVAL(I,IACTRF) .NE. IVAL(I,MACTR)) GO TO 38
39   37    CONTINUE

```

FSN MODEL - PEQUAL FUNCTION

ISN

```
C
C ... IF WE GET HERE, MATCH
C
40      RETURN
41      MACTRF=NEXT(LISREF,LFTE)
42      GO TO 35
C
C ... LOCATION, COMPARE PLACE FIELD ONLY
C
43 40    LOCBF1=IVAL(IATTR,IVWREF)
44    LCCBF2=IVAL(JCCBT,IMSGEF)
45    IF(IVAL(JPLACE,LOCRF1) .EQ .IVAL(JPLACE,LOCRF2)) RETURN
C
C ... SET FUNCTION VALUE TO .FAISE.
C
46 50    PEQUAL=.FALSE.
47    RETURN
48    END
```

FSN MODEL - PLANAC SUBROUTINE

ISN
1 SUBROUTINE PLANAC(IFLAG,LEADER)

C A ROUTINE TO IMPLEMENT A LEADER'S PLANNING OF ACTIONS. THE
C ROUTINE SELECTS CERTAIN TYPES OF SITUATIONS TO MONITOR AND SELECTS
C ACTIONS ACCORDING TO THESE SITUATIONS.
C THE CURRENT SET OF RULES MAKE SOME PROBABILISTIC SELECTIONS FROM
C A SET OF ALTERNATIVE ACTIONS, GIVEN A PARTICULAR SITUATION.

C INPUT PARAMETERS:
C IFLAG INDICATOR OF WHICH CONDITION CAUSED THE ROUTINE TO BE
C CALLED: (1) A CHANGE IN A FORCE'S SITUATION, (2) NON-
C REPORTING OF FORCE MEMBERS AT EXPECTED TIMES.
C LEADER REFERENCE TO LEADER DOING THE PLANNING.

C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DITEM,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCBECT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
11 COMMON /RECREF/
12 DIMENSION RECREF(140)
13 EQUIVALENCE (RECREF(1),IGCALS)
14 COMMON /NEW/
C
15 DIMENSION PROBS(40),NCETS(5,2),IPTR(5,2)
16 DATA PROBS/.3,.4,.3,.1,.3,.6,.1,.4,.5,.1,.2,.3,.4,
+ .2,.8,.6,.4,.8,.2,.9,.1,
+ .2,.8,.6,.4,.8,.2,.9,.1/
17 DATA NCETS/0,3,3,3,2,2,2,2,2,2/
18 DATA IPTR/0,1,4,7,14,22,22,22,22,22/

FSN MODEL - PLANAC SUBROUTINE

ISN C GET POINTER TO FORCE THAT LEADER COMMANDS
19 IFCECE = IVAI(JFCECE,LEADER)
C GET CURRENT FORCE SITUATION
20 ISITN = IVAL(JSITN,IPCBCE)
C GET SIDE OF LEADER
21 ISIDE = IVAL(JFALEG,IFORCE)
C SEPARATE LIST OF PERSONS IN FORCE INTO THOSE WHO ARE DEAD OR CAPTURED
C AND THOSE WHO ARE WHOLE OR JUST WOUNDED.
22 IPSNOK = NULL
23 IPSNX = NULL
24 ISNMON = NULL
25 PSNCK = 0.
26 PSNX = 0.
27 SNMON = 0.
C
28 IPSNS = IVAL(JCCNTS,IFORCE)
29 IPERSN = IFIRST(IPSNNS,RP)
30 2 IF (IPERSN .EQ. NULL) GO TO 8
31 IPSTAT = IVAL(JPSTAT,IPERSN)
32 IF (IPSTAT .GE. KWCUND) GO TO 4
C PLAYER IS DEAD OR CAPTURED
33 PSNX = PSNK + 1.
34 IPSNX = ISTACK(IPEESN,IPSNX)
35 GO TO 6
C AGGREGATE WOUNDED OR WHOLE PERSONS ASSIGNED AS SENSOR MONITORS
36 4 IF (IVAL(JTYEE,IPEESB) .NE. KSNMON) GO TO 5
37 SNMON = SNMON + 1.
38 ISNMON = ISTACK(IEEEESN,ISNMON)
39 GO TO 6
C PLAYER IS WHOLE OR WOUNDED & NOT SENSOR MONITOR
40 5 PSNCK = PSNOK + 1.
41 IPSNCK = ISTACK(IPERSN,IPSNOK)
C CONSIDER NEXT PERSON
42 6 IPERSN = NEXT(IPSNNS,RP)
43 GO TO 2
C

FSM MODEL - PLANAC SUBROUTINE

ISN

C SET UP TO SELECT ALTERNATIVE ACTIONS

44 8 IPT = IPTR(ISITN,ISIDE)

45 N = NOPTS(ISITN,ISIDE)

C

C

C BRANCH ACCORDING TO FLAG

46 GO TO (10,200), IFLAG

C

C BRANCH ACCORDING TO SIDE

C

47 10 GO TO (12,100), ISIDE

C

C *** SFE PLAYER

C

C HAVE A CHANGE IN THE FORCE'S SITUATION CODE. BRANCH ACCORDING

C TO SITUATION CODE'S CURRENT VALUE.

48 12 GO TO (15,20,40,60,80), ISITN

C

49 15 RETURN

C

C SFE ALERTED, BEING RESOLVED BY ANOTHER FORCE

C SELECT ALTERNATIVE ACTION

50 20 IOPT = ICHOOS(N,PROBS(IPT),IRN(2))

C BRANCH BASED ON ALTERNATIVE SELECTED

51 GO TO (24,28,32), IOPT

C

C OPTION 1: DO NOTHING

52 24 RETURN

C

C OPTION2: GUARD SFE ROCE WITH UP TO THREE GUARDS

53 28 MFORCE = NITEMS(3,IPSNOR,IEEM)

54 CALL SECURE(ISNBERE,LEADER,MFORCE)

55 RETURN

C

C OPTION 3: DIRECT PERSONS WHO CAN SEE REGION OF ALERT TO LOOK

C IN THAT DIRECTION

FSN MODEL - PLANAC SUBROUTINE

ISN

56 32 CALL FLNOBS(LOCATE,LEADER,IPSNOK)
57 RETURN
C
C SPE ALERTED, BEING RESCIVED BY OWN FORCE
C SELECT ALTERNATIVE ACTION
58 40 ICPT = ICHOOS(N,PROBS(IET),IRN(2))
C BRANCH BASED ON ALTERNATIVE SELECTED
59 GO TO (42,46,50), ICPT
C
C OPTION 1: DO NCTHING
60 42 RETURN
C
C OPTION 2: SEND UP TO THREE PLAYERS TO SECURE THE SNM ROOM
61 46 MFORCE = NITEMS(3,IPSNOK,IREM)
62 CALL SECURE(ISNMRM,LEADER,MFORCE)
63 RETURN
C
C OPTION 3: SEND TWO PLAYERS TO CHECK OUT THE REGION OF THE ALARM
64 50 MFORCE = NITEMS(2,IPSNOK,IREM)
65 CALL CHKOUT(NULL,LCCAFE,LEADER,MFORCE)
66 RETURN
C
C AFE DETECTED; NO ENGAGEMENT YET.
C
C BRANCH ACCORDING TO WHETHER FREE FIRING IS ALLOWED
67 60 IF (IFCOND(ISIDE) .GE. 3) GO TO 70
C SELECT ALTERNATIVE ACTION IF NO FIRING YET PERMITTED
68 ICPT = ICHOOS(N,FBCE(S,IET),IRN(2))
C BRANCH ACCORDING TO OPTION SELECTED
69 GO TO (62,65,68), ICPT
C
C OPTION 1: DO NCTHING
70 62 RETURN
C
C OPTION 2: SET FIRING CCNDITION TO "DON'T FIRE UNLESS FIRED UPON".
C ALSO DISPATCH THREE PLAYERS TC BLOCK ADVERSARY FORCE'S MOVEMENT TO

FSN MODEL - PLANAC SUBROUTINE

ISN

C THE SNN BOOM.

71 65 IFCOND(ISIDE) = MAX0(IFCOND(ISIDE),2)

72 66 MFORCE = NITEMS(3,IPSNOK,IREM)

73 CALL PLNBLK(NULL,LCCAFE,ISNMRR,LEADER,MFORCE)

74 RETURN

C

C OPTION 3: SET FIRING CCNDITION TO "DON'T FIRE UNLESS FIRED UPON".

C ALSO DISPATCH THREE PLAYERS TOWARDS PRESUMED ADVERSARY FORCE LOCN.

75 68 IFCCND(ISIDE) = MAX0(IFCOND(ISIDE),2)

76 69 MFORCE = NITEMS(3,IPSNOK,IREM)

77 CALL CHKOUT(NULL,LCCAFE,IFADER,MFORCE)

78 RETURN

C

C AFE DETECTED, NO ENGAGEMENT YET; FIRING ALLOWED.

79 70 N = 4

80 IPT = IPT + 3

C SELECT ALTERNATIVE ACTION

81 IOPT = ICHOOS(N,PROBS(IPT),IRN(2))

C BRANCH ACCORDING TO CPTION SELECTED

82 GO TO (62,65,76,69), ICPT

C

C OPTION 3: SET AIM PCINT CONDITION TO "FIRE TO WOUND". THEN

C BLOCK THE ADVERSARY'S PCMENT.

83 76 IAMPT(ISIDE) = 2

84 GO TO 66

C

C ENGAGEMENT UNDERWAY. DETERMINE PERCENTAGE OF FORCE REMAINING.

85 80 TOT = PSNOK + FSNX

86 IF (TOT .LE. 0.) GO TO 84

87 PCENT = PSNOK / TOT

88 ISTATE = PCENT*4.999 + 1.0

89 GO TO (84,84,88,92,96), ISTATE

C

C 0 - 40% EFFECTIVE STRENGTH

90 84 IOPT = ICHOOS(N,PROBS(IPT),IRN(2))

91 IF (ICPT .EQ. 1) RETURN

FSN MODEL - PLANAC SUBROUTINE

ISN

92 GO TO 98

C

C 40 - 60% EFFECTIVE STRENGTH

93 88 IOPT = ICHOOS(N,PRCES(IET+2),IRN(2))

94 IF (IOPT .EQ. 1) RETURN

95 GO TO 98

C

C 60 - 80% EFFECTIVE STRENGTH

96 92 IOPT = ICHOOS(N,PROBS(IET+4),IRN(2))

97 IF (IOPT .EQ. 1) RETURN

98 GO TO 98

C

C 80 - 100% EFFECTIVE STRENGTE

99 96 IOPT = ICHOOS(N,PRCBS(IET+6),IRN(2))

100 IF (IOPT .EQ. 1) RETURN

C

C DISENGAGE: IF ADVERSARY, TRY TO ESCAPE FROM SITE. IF SECURITY FORCE,
C RETREAT TO SNM BOCP.

101 98 IF (ISIDE .EQ. 1) CALL SECURE(IMONRM,LEADER,IPSNOK)

102 IF (ISIDE .EQ. 2) CALL ESCAPE(IEXITS,LEADER,IPSNOK)

103 RETURN

C

C

C *** ADVERSARY PLANS

C

C IF HAVE JUST ACHIEVED A GOAL AND ARE TO BEGIN WORKING ON A NEW GOAL,
C ASSIGN ALL OF AFE FORCE TO MOVE TO IT.

104 100 IF (NEWGOL .EQ. NULL) GC TC 110

105 IGOAL = IFIRST(IGOALS,KG)

106 LCC = IVAL(JID,IGCAL)

107 CALL CHKOUT(NULL,LCC,LEADER,IPSNOK)

108 RETURN

C

C IF NO NEW GOAL, BUT IF THERE IS AN ENGAGEMENT GOING ON, TEST
C WHETHER WISH TO BREAK OFF.

C TEST FOR HOW MANY PLAYER CURRENTLY RESPONDING TO GOAL

FSN MODEL - PLANAC SUBROUTINE

ISN C
109 110 INCEL=NULL
110 NUM=0
111 IGCAL=IFIRST(IGCAIS,KH)
112 LOC=IVAL(JID,IGOAL)
C
C COUNT NUMBER OF PLAYERS AT ICC OR MOVING TO LOC
C
113 IPSNS=IVAL(JCONTS,IFORCE)
114 IESN=IFIRST(IPSNS,KL)
115 120 IF(IPSN .EQ. NULL) GC TO 190
C
C TEST PHYSICAL STATUS
C
116 IF(IVAL(JPSTAT,IPSN) .LE. KCAPTE) GO TO 180
C
C TEST LOCATION
C
117 ILOC0=PLACE(IVAL(JLCCN,IPSN))
118 IF(ILCCC .EQ. ICC) GC TO 160
C
C TEST IF MOVING TO ICC
C
119 IPLANS=IVAL(JPIANS,IESN)
120 IPLAN=IFIRST(IPLANS,KN)
121 140 IF(IPLAN .EQ. NULL) GC TO 170
122 IF(IVAL(JTYPE,IPLAN) .NE. KMOVNG) GO TO 150
123 IF(IVAL(JPAR1,IELAN) .EQ. LCC) GO TO 160
C
124 150 IELAN=NEXT(IPLANS,KN)
125 GO TO 140
C
C COUNT PLAYER AS MOVING TOWARD GOAL
C
126 160 NUM=NUM+1
127 GO TO 180

FSK MODEL - PLANAC SUBROUTINE

ISN

```
C
C     PLAYER NOT ACTIVE FOR GOAL
C
128    170 INCEL=ISTACK(IESN,INCEI)
C
C     NEXT PLAYER ON FORCE
C
129    180 IPSN=NEXT(IPSN,KT)
130    GO TO 120
C
C     TEST IF GOAL BEING SATISFIED
C
131    190 IF(IVAL(JADVRQ,IGOAL) .LE. NUM) GO TO 200
C
C     ADDITIONAL PLAYERS NEED TO BE SENT
C
132    NDIFF=IVAL(JADVRQ,IGOAL) - NUM
133    IPSN=IFIRST(TNCPI,KT)
134    191 IF(IPSN .EQ. NULL) GO TO 192
C
135    IACT=NEWRC6(LACTN,KMCVNG,LOC,NULL,NULL,NULL,IPSN)
136    IMSG=NEWRC5(LMESS,LEADER,IPSN,JPLANS,TMIN,IACT)
137    CALL COMMO(LEADER,IPSN,IMSG)
138    NDIFF=NDIFF-1
139    IF(NDIFF .EQ. 0) GO TO 200
140    IPSN=NEXT(INOPI,KT)
141    GO TO 191
C
C     CAN NOT COMPLETE GOAL
C
142    192 IF(ISITN .NE. 4) GO TO 200
143    GO TO 80
C
C     ACTIONS OCCASIONED BY A NCN-REPORTING OF FORCE ARE NOT IMPLEMENTED
C     IN THIS VERSION
144    200 RETURN
```

FSN MODEL - PLANAC SUBROUTINE

ISBN
145

END

FSN MODEL - PLNELK SUBROUTINE

TSN

1

SUBROUTINE PLNELK(IPC5CE,LOCNOW,LOCTO,LEADER,IFORC2)

C
C ROUTINE FOR A LEADER TO ELAN AND ACTIVATE THE DEPLOYMENT OF HIS
C FORCES TO BLOCK AN ENEMY FORCE MOVING FROM A CURRENT POSITION
C TOWARDS A GOAL POSITION. THE ROUTINE DETERMINES THE LOCATION OR
C LOCATIONS TO WHICH TO DISPATCH THE PERSONS SPECIFIED AND THEN
C GENERATES ORDERS FOR THESE PERSONS TO MOVE TO THESE LOCATIONS.
C THE CURRENT VERSION OF THE ROUTINE USES THE VERY SIMPLIFIED
C STRATEGY OF DISPATCHING EACH OF THE INTERCEPTORS TO THE PRESUMED
C GOAL OF THE TO-BE-INTERCEPTED FORCE.

C

C INPUT PARAMETERS:

C IFORCE PERSON OR LIST OF PERSONS OR VEHICLES TO BE INTERCEPTED
C LOCNOW LATEST KNOWN LOCATION OF IFORCE
C LOCTO PRESUMED DESTINATION OF IFORCE
C LEADER REFERENCE TO LEADER PLANNING THE BLOCKING ACTION
C IFORC2 LIST OF PERSONS OR VEHICLES TO BE USED FOR THE
C BLOCKING ACTION

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),EITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FIDNAM,FC6POT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/

C

C ITERATE THROUGH EACH OF THE PERSONS TO BE DISPATCHED.

C

11 IPERSN = IFIRST(IFORC2,KP)
12 10 IF (IPERSN .EQ. NULL) RETURN

C

C SET UP INSTRUCTIONS FOR THIS PERSON TO MOVE TO THE OPPONENTS*

FSN MODEL - PLNBLK SUBROUTINE

ISN

C PRESUMED GOAL
13 IACT = NEWRC6(IACTB,RECVNG,LOCTO,NULL,NULL,NULL,IPERSN)
14 IMSG = NEWRC5(IMESS,LEADER,IPERSN,JPLANS,TMIN,IACT)
C SEND MESSAGE TO THE PLAYEE
15 CALL COMMO(LEADER,IPERSN,IMSG)
C GET THE NEXT PERSON IN THE FORCE TO BE DISPATCHED
16 IPERSN = NEXT(IFORC2,EP)
17 GO TO 10
18 END

PSN MODEL - PLNOBS SUBROUTINE

1 SUBROUTINE PLNCES(LOCS,LEADER,IFORCE)
C
C ROUTINE FOR A LEADER TO CALL TO INSTRUCT EACH OF A GROUP OF
C SUBORDINATES WHO HAVE LINE-OF-SIGHT TO A GIVEN LOCATION TO OBSERVE
C IN THAT DIRECTION.
C
C INPUT PARAMETERS:
C LCCS NODES AT WHICH TO LOOK
C LEADER REFERENCE TO LEADER GIVING ORDERS
C IFORCE LIST OF PERSONS WHO ORDERS ARE TO BE GIVEN TO
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IPAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCRPT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
C
C ITERATE OVER PERSONS TO GIVE ORDERS TO
11 IPERSN = IPIRST(IFORCE,KP)
12 10 IF (IPERSN .EQ. NULL) RETURN
C
C ITERATE OVER LOCATIONS AT WHICH TO OBSERVE
13 LOC = IPIRST(LOCS,KL)
14 12 IF (LOC .EQ. NULL) GO TO 20
C DOES PERSON HAVE LINE-OF-SIGHT TO LOCATION ?
15 IF (LCS(IPERSN,LOC) .EQ. 0) GO TO 15
C
C IF SO, DIRECT HIM TO OBSERVE IN THAT DIRECTION
16 CALL DIREC(IPERSN,LOC,THETA,PHI)
17 IACT = NEWRC6(LACT,KDETNG,THETA,NULL,NULL,NULL,IPERSN)
18 IMSG = NEWRC5(LMESS,LEADER,IPERSN,JPLANS,TMIN,IACT)

FSN MODEL - PLNOBS SUBROUTINE

18N
19 CALL CCMEC(LEADEE,IPERSN,IMSG)
20 GO TO 20
C
C GET NEXT LOCATION
21 15 LCC = NEXT(LCCS,KI)
22 GO TO 12
C
C GET NEXT PLAYER
23 20 IPERSN = NEXT(IFCBCE,KP)
24 GO TO 10
25 END

FSN MCDEL - POBS FUNCTION

ISN

```
1      FUNCTION POBS(IREF,JREF,T)
C      FUNCTION TO CALCULATE THE PROBABILITY I SEES J WITHIN TIME T
C
C      IREF  -> REFERENCE TO PERSON I, THE OBSERVER
C      JREF  -> REFERENCE TO ENTITY J, THE OBSERVATION TARGET
C      T      -> TIME AVAILABLE FOR OBSERVATION
C
2      COMMON /PARS/
3      EQUIVALENCE (PNULL,NULI),(IPFAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FCRMOT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      COMMON /DATAV/
9      DIMENSION ACTTIM(25)
10     EQUIVALENCE (ACTTIM(1),ACTRAT(1))
11     CCHECN//OCTANT(8)
C
12     DATA PI/3.14159/
C
13     POBS = 0.
C      DETERMINE THE OCTANT OF J RELATIVE TO I
14     CALL DIBEC(IREF,JREF,THETA,PHI)
15     JCCT = INT((4.*THETA)/PI) + 1
C      FIND THE EXPOSED AREA OF J
16     CALL PARSBF(JREF,ICHTY,I,IDUM)
17     ITYPE = IVAL(JITYPE,JREF)
18     EXPSR=1.
19     IF (IDTYP .NE. IPERSN) GO TO 50
20     ITYPE = 1
C      GET POSTURE AND COVER OF TARGET
21     COVER = COVB(JREF,THETA + PI)
22     POST = VAL(JECSTR,JREF)
C      FIND THE EXPOSURE OF THE TARGET
23     EXPSR = AMIN1(1.,PCST - CCVLT)
```

FSK MODEL - POBS FUNCTION

```

ISN
24      IF (EXPSR .LE. 0.) RETURN
25  50  IPCBTF = IPHSOE(IDTYE-IWIND) + ITYPE
26  60  A = AREA(IPOETP)*EXPSR
      C FIND THE RANGE OF J
27      R = DIST(IREF,JREF)
28      IF(R .GT. .1) GC TO 98
29      POES=1.0
30      RETURN

      C
      C DETERMINE J'S ACTIVITY REFERENCE
31      98 NACT = 1
      C***IF NOT A PERSON ASSUME STATIONARY
32      IF (IDTYP .NE. IPERSN) GC TO 150
33      100 JACREF = IVAL(JACTIV,JBEF)
34      IF(JACREF .EQ. NULL) GO TO 150
      C FIND THE ACTIVITY OF J
35      NACT = IVAL(JPAH1,JACREF)
36      IF(NACT .NE. NULL) GC TO 200
37      150 NACT=3
38      200 NVDCI=1
39      IF(NACT .EQ. 2) NVDCI=2
40      IF(NACT .EQ. 1) NVDCI=3
      C GET DETECTION RATE BASED ON J'S VISUAL DETECTION CLASS
41      DETR = DETRAT(NVDCI)*550.04
      C CALCULATE BASIC DETECTION RATE
42      IPSTAT = IVAL(IESTAT,IREF)
43      IF (IPSTAT .GT. 4 .OR. IESTAT .LT. 1) IPSTAT = 4
44      AL = -(DETR/DETEHI)*(1/(E*R))*SKILL(KDETSK,IPSTAT,I)
45      AL = AMAX1(AL,-10.)
      C CALCULATE PROBABILITY OF OBSERVATION
46      POES = 1. - EXP(AL*OCTAM(JOCT)*T)
47      RETURN
48      END

```

FSN MODEL - POCT SUBROUTINE

1 SUBROUTINE POCT(IREF,THETA)

C SUBROUTINE TO DETERMINE THE PROBABILITY OF AN OBSERVER SCANNING
C IN EACH STANDARD 45 DEGREE OCTANT, MEASURED FROM THE X-AXIS
C IREF -> REFERENCE TO CBSEVER
C THETA -> PRINCIPAL DIRECTION OF OBSERVATION; IF NULL, UNIFORM
C OCTANT<- ARRAY OF PROBABILITIES OF SCAN FOR EACH OCTANT

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NOLL),(IFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FCBNCT
5 COMMON /PARS1/
6 COMMON /DATAV/
7 DIMENSION ACTTIP(25)
8 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
9 COMMON//OCTANT(8)

C DIMENSION Q(4)
10 DATA PI/3.14159/

C IF NO PDO INDICATED, EACH OCTANT HAS EQUAL PROBABILITY OF SCAN
12 IF (THETA .NE. FNULL) GO TO 200
13 DO 100 J=1,8
14 100 OCTANT(J) = .125
15 GO TO 550

C DETERMINE THE PRIMARY SCAN OCTANT
16 200 THETA = AMOD(THETA,2.*PI)
17 IF (THETA .LT. 0.) THETA=THETA + 2. * PI
18 IOCT = INT((4.*THETA)/PI) + 1

C DISTRIBUTE THE PROBABILITIES AMONG THE OCTANTS
19 DO 300 K=1,8
20 J = MOD(IOCT+K-2,8) + 1
21 300 OCTANT(J) = DETANG(K)

22 400 CONTINUE

C

FSN MODEI - POCT SUBROUTINE

ISN

```

C ADJUST SCAN PROBABILITIES BASED ON ACTIVITY OF OBSERVER
C
23      IA = NACTSU(IREF,NULL)
24      NCCT = NDTSEC(IA)
25      IF (NOCT .GT. 0) GO TO 450
26 420  CCNTINUE
27      DO 430 J=1,8
28 430  OCTANT(J) = 0.
29      RETURN
C
30 450  IF (NOCT .GE. 8) GO TO 550
C DETERMINE THE INDICES OF THE EXTREME OCTANTS TO BE INCLUDED
C IN THE SCAN, SYMMETRICALLY ABOUT THE PRIMARY OCTANT
31      MDISP = NOCT/2
32      LMIN = IOCT - MDISP
33      LMAX = IOCT + MDISP
34      IF (LMIN .LT. 1) LMIN = IMIN + 8
35      LMAX = MOD(LMAX-1,8)+1
C DETERMINE THE SUM OF THE PROBABILITIES OF THE SCANNED OCTANTS
36      DENCM = 0.
37      L = LMIN
38 470  L = MOD(L-1,8) + 1
39      IF (L .EQ. LMAX) GO TO 490
40      DENCM = DENCM + OCTANT(L)
41      L = L + 1
42      GO TO 470
C INCREASE THE PROBABILITIES IN THE SCANNED OCTANTS BY A FACTOR
C OF 1/SUM OF SCANNED OCTANTS (= 1/1 - SUM OF EXCLUDED OCTANTS)
43 490  L = LMIN
44 500  L = MOD(L-1,8) + 1
45      IF (L .EQ. LMAX) GO TO 510
46      OCTANT(L) = OCTANT(L)/DENCM
47      L = L + 1
48      GO TO 500
C ZERO OUT THE REMAINING PROBABILITIES
49 510  L = LMAX + 1

```

FSN MODEL - POCT SUBROUTINE

ISN
50 520 L = MOD(L - 1, E) + 1
51 IF (L .EQ. LMIN) GC TC 550
52 OCTANT(L) = 0.
53 L = L + 1
54 GO TO 520
55 550 CONTINUE
C
C ADJUST SCAN PROBABILITIES BASED ON LOCAL COVER OF OBSERVER
C
C GET OBSERVER'S PCSTUBE
56 POST = VAL(JPOSTR,IREF)
57 IF (PCST .LT. 0. .OR. PCST .GT. 1.) POST = 1.
C DETERMINE THE OBSERVER'S EXPOSURE IN EACH QUADRANT
58 DO 600 J=1,4
59 600 Q(J) = POST - COVR(IREF,FLOAT(2*j-1)*PI/4.)
C ITERATE THROUGH THE QUADRANTS, REDISTRIBUTING SCAN PROBABILITIES
60 PSUM = 0.
61 DO 700 J=1,4
62 IF (Q(J) .GE. .1) GC TC 700
63 PSUM = PSUM + OCTANT(2*j) + OCTANT(2*j-1)
64 OCTANT(2*j) = 0.
65 OCTANT(2*j-1) = 0.
66 700 CONTINUE
C IF THE EXCLUDED OCTANTS HAVE NO PROBABILITY, DONE
67 IF (PSUM .LE. 0.) RETURN
C OTHERWISE, INCREASE REMAINING PROBABILITIES BY 1/1-SUM OF EX-
C CLUSED PROBABILITIES
68 IF (PSUM .GT. .99) GC TC 420
69 DO 750 J=1,8
70 750 OCTANT(J) = OCTANT(J)/(1. - PSUM)
71 RETURN
72 END

PSM MODEL - PORTST SUBROUTINE

ISN

1

SUBROUTINE PORTST(IPLABF,IOLDRF)

C

-- THE PURPOSE OF THIS SUBROUTINE IS TO:
1. DETERMINE THE STATES OF A PORTAL
ONCE A PLAYER HAS PASSED THROUGH IT.

C

-- INPUT PARAMETERS

C

IPLABF -- PLAYER REFERENCE
IOLDRF -- CID LOCATION REFERENCE

C

C

-- OUTPUT

C

THIS SUBROUTINE ADJUSTS THE STATUS OF THE
PORTAL AFTER IT HAS BEEN PENETRATED

C

C

-- PROCEDURE

C

1. DETERMINE IF PLAYER HAS PASSED THROUGH PORTAL
2. IF SO, DETERMINE WHAT THE STATUS OF THE
PORTAL IS.
3. IF LOCKABLE, DETERMINE IF PAYER HAS A KEY TO
LOCK THE PORTAL
4. IF SECURED, DETERMINE IF PLAYER HAS EQUIPMENT TO
DESTROY THE PORTAL

C

C

COMMON /PARS/
EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
REAL*8 DTPNAM,FIDNAM,FCFPCT
COMMON /PARS1/
COMMON /PARS2/
COMMON /RECREF/

2

3

4

5

6

7

FSM MODEL - PORTST SUBROUTINE

```
1 ISN
2      DIMENSION RECBFG(140)
3      EQUIVALENCE (RECRFO(1),IGOALS)
4      DIMENSION INSTAT(2,5)
5      EQUIVALENCE (IOESV(41),INSTAT(1,1))
6
7      C
8      NINSDE=5
9      KINSDE=4
10
11
12      C
13      -- FINE LOCATION OF PLAYER
14      C
15      ILCC=IVAL(JLCCN,IPLARP)
16      ISOU=IVAL(JSCUBC,ICLDF)
17      ISIN=IVAL(JSINK,ILOC)
18
19      C
20      -- DETERMINE IF PASSED THROUGH PORTAL
21      C
22      IPOSUE=NULL
23      ATIM=TRAVEL(IPLARP,ISCU,ISIN,KBUN,IPOSUB)
24      IF(IPOSUE .EQ. NULL) GO TO 30
25
26      C
27      -- PASSED THROUGH PORTAL
28
29      C
30      IF(IPOSUB .EQ. KPPWEP) GO TO 20
31
32      C
33      DETERMINE IF THIS PLAYER OPENED PORTAL
34
35      C
36      IPLNS=IVAL(JPLANS,IPLARP)
37      IPLN=IFIRST(IPLNS,KA)
38      IF(IVAL(JPAR3,IPLN) .EQ. NULL) GO TO 30
39
40      C
41      DETERMINE IF CAPABLE OF LOCKING PORTAL
42
43      C
44      10 IEQRF=NULL
45      CALL KEYS(IPLARP,ISIN,IEQRF)
46      IF(IEQRF .EQ. NULL) GO TO 15
```

PSM MODEL - PORTST SUBROUTINE

ISN

```
C
C -- CAPABLE OF LOCKING PORTAL
C
27     CALL CHGFLD(JSTAT,ISOU,KICCKD,1,IDUM)
28     CALL CHGFLD(JSTAT,ISIS,KICCKD,1,IDUM)
29     GO TO 30
C
C -- NOT CAPABLE OF LOCKING PORTAL
C
30     15 CALL CHGFLD(JSTAT,ISOU,KCLOSD,1,IDU)
31     CALL CHGFLD(JSTAT,ISIS,KCLOSD,1,IDU)
32     GO TO 30
C
C -- WEAPON USED AGAINST PORTAL
C     STATUS IS DESTROYED
C
33     20 CALL CHGFLD(JSTAT,ISOU,KDESTR,1,IDU)
34     CALL CHGFLD(JSTAT,ISIS,KDESTR,1,IDU)
C
C -- PORTAL DESTROYED, TEST IF INSIDER DID IT
C
35     IF(IVAL(JTYPE,IPLARF) .NE. KINSDB) GO TO 30
C
C -- SET TRIGGER
C
36     DO 25 IK=1,NINSDR
37     IF(INSTAT(1,IK) .EQ. IPLARF) GO TO 26
38     25 CONTINUE
39     GO TO 30
C
C -- TRIGGER VALUE OF 5
C
40     26 INSTAT(2,IK)=5
C
41     30 CONTINUE
42     RETURN
```

PSN MODEL - PORTIST SUBROUTINE

PSN
43

END

FSN MODEL - POROPN SUBROUTINE

1 SUBROUTINE POROPN(IPLABF,ISOU,ISIN,ISUM)
C
C -- THE PURPOSE OF THIS SUBROUTINE IS TO OPEN A
C PORTAL BEFORE A PLAYER TRAVELS THROUGH IT
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DITEM,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCREPOT
8 COMMON /PARS1/
9 COMMON /PARS2/
C COMMON /PARS3/
C
C TEST IF BREECHING A BARRIER
C
11 IF (ISUM .EQ. KEEWEE) GO TO 50
C
C TEST IF SCALING FCRTAL
C
12 IF (ISUM .EQ. KSCALE) GO TO 50
C
C SAVE ORIGINAL STATUS OF FCRTAL
C
13 ISTT=IVAL(JSTAT,ISCU)
14 IF (ISTT .LE. KOPEN) GO TO 50
15 IPLNS=IVAL(JPLANS,IPLABF)
16 IPLN=IFIRST(IPLNS,KA)
17 IPTR=IREP1(IPLN)
18 ITEM(IPTR+JPARE)=ISIT
C PENETRATING A PORTAL SET STATUS TO OPEN
C
19 CALL CHGFLD(JSTAT,ISCU,KCPEN,1,IDU)
20 CALL CHGFLD(JSTAT,ISIN,KCEEN,1,IDU)

FSN MODEL - POROPN SUBROUTINE

ISN

C

21 50 CONTINUE

C

22 RETURN

23 END

FSN MCDEL - SCALE SUBROUTINE

1 SUBROUTINE SCALE(IPLARF,IEARR,IEQRF,AHGT)

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. DETERMINE IF A PLAYER HAS EQUIPMENT
C NECESSARY TO SCALE A BARRIER

C -- INPUT PARAMETERS

C

C IPLARF -- PLAYER REFERENCE

C IBARR -- BARRIER REFERENCE

C

C -- OUTPUT VARIABLES

C

C IEQRF -- EQUIPMENT DIFFERENCE

C AHGT -- HEIGHT OF BARRIER

C

C -- PROCEDURE

C 1. DETERMINE HEIGHT OF BARRIER

C 2. DETERMINE IF ANY EQUIPMENT IS NEEDED

C 3. DETERMINE IF PLAYER HAS EQUIPMENT TO SCALE
C BARRIER

C

C

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FCRPT
5 COMMON /PARS1/
6 CCMMCN /PARS2/
7 CCMMCN /PARS3/

C

8 IEQRF=NULL

C -- HEIGHT OF BARRIER

C

9 AHGT=VAL(JHIGH,IPARR)

FSN MODEL - SCALE SUBROUTINE

ISN

```
C
C -- RULE
C     IF HEIGHT IS LESS THAN 1 METER, NO EQUIPMENT
C     IS NEEDED TO SCALE FABRIER
C
10    IF(AHGT .GT. 1.0) GO TO 10
11    IEQRF=0
12    GO TO 40
C
C -- SEARCH EQUIPMENT LIST OF PLAYER
C
13    10 IEQLT=IVAL(JWEAPS,IPLABF)
14    IEQUIF=IPIRST(IEQLT,KA)
15    15 IF(IEQUIP .EQ. NULL) GO TO 40
C
C -- IGNORE WEAPONS
C
16    CALL PARSBF(IEQUIP, IDY, IR, ID)
17    IF(IDY .NE. IEQUIF) GO TO 20
C
C -- FIND SCALING EQUIPMENT
C
18    IF(IVAL(JTYPE,IEQUIP) .NE. KSCALR) GO TO 20
C
C -- TEST HEIGHT OF SCALING EQUIPMENT
C
19    IF(PLCAT(IVAL(JEAR,IEQUIF)) .GE. AHGT) GO TO 30
C
20    20 IEQUIZ=NEXT(IEQLT,KA)
21    GO TO 15
C
22    30 IEQRF=IEQUIP
C
23    40 CONTINUE
24    RETURN
25    END
```

FSN MODEL - SDESIR SUBROUTINE

1 SUBROUTINE SDESIR(IPLARF,TEMP)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. EVALUATE THE DESIRABILITY FOR A PLAYER TO

C SURRENDER.

C

C

C -- INPUT PARAMETERS

C

C IPLARF -- PLAYER PREFERENCE FOR SURRENDER

C DECISION

C

C

C -- OUTPUT PARAMETERS

C

C TEMP -- VALUE OF DESIRABILITY TO BE COMPARED

C TO THE SURRENDER THRESHOLD.

C

C

C

C -- PROCEDURE

C 1. SEARCH PERCEPTIONS OF PLAYER

C TO DETERMINE WHICH CASE

C SUBSETS APPLY AND ADD THE

C APPROPRIATE VALUES

C

2 COMMON /PARS/

3 EQUIVALENCE (FNUIL,BUII),(IFAIL,FAIL)

4 REAL*8 DTPNAM,FLENAM,FCRMCT

5 COMMON /PARS1/

6 COMMON /PARS2/

7 COMMON /PARS3/

8 COMMON /DATAV/

9 DIMENSION ACTTIME(25)

FSN MODEL - SDESIR SUBROUTINE

10 EQUIVALENCE (ACTTIM(1),ACTBAT(1))
C
C
11 TME=0.
12 IF(EVALN(53) .IE. 0) EVALN(53)=1.0
C
C -- CASE SUBSET 1
C PHYSICAL STATUS WOUNDED
C
13 IF(IVAL(JPSTAT,IPLARF) .EQ. KWOUND)
* TEMP=TME+EVALN(51)
C
C -- CASE SUBSET 2
C NUMBER OF THREATS TO BE UNDER FIRE FROM
C
14 IPLT=IVAL(JPRCPS,IPLARF)
15 IPER=IFIRST(IPIT,KA)
16 21 IF(IPER .EQ. NULL) GO TO 28
17 IPEBRF=IVAL(JVIEW,IEER)
18 CALL PARSBF(IPERRF, IDYY, IRBN, IDDUM)
19 IF(IDYY .NE. IPEERN) GO TO 25
20 IACLT=IVAL(JACTIV,IPERRF)
21 IF(IACLT .EQ. NULL) GC TO 25
22 IACRF=IFIRST(IACLT,KE)
23 22 IF(IACRF .EQ. NULL) GO TO 25
24 IF(IVAL(JTYPE,IACRF) .NE. KFIERN) GO TO 24
25 IF(IVAL(JPAH1,IACRF) .EQ. IPLARF) TEMP=TEMP+EVALN(52)
C
26 24 IACRF=NEXT(IACLT,KE)
27 GC TO 22
C
28 25 IPEB=NEXT(IPIT,KA)
29 GC TO 21
C
30 28 TME=TEMP/EVALN(53)
31 IF(TME .GT. 1.) TME=1.

FSM MODEL - SDEGIR SUBROUTINE

ISN
32
33

RETURN
END

PSN MODEL - SECURE SUBROUTINE

ISN

1

SUBROUTINE SECURE(IRCCE,LEADER,IPFORCE)

C
C ROUTINE BY WHICH A LEADEE CAN PLAN AND GIVE INSTRUCTIONS TO
C SUBORDINATES TO SECURE A ROOM. SECURING A ROOM WILL BE ACCOMPLISHED
C BY POSTING ONE OR MORE GUARDS AT EACH OF THE PORTALS OF THE ROOM,
C AND HAVING ONE OR MORE GUARDS TAKE UP POSITIONS IN THE ROOM. GUARDS
C WILL BE ASSIGNED ONE AT A TIME TO POSITIONS TO BE PROTECTED UNTIL NO
C MORE OF THE GUARDS ALLOCATED TO THIS MISSION ARE LEFT.

C

C INPUT PARAMETERS:

C IRCCE ROOM TO BE SECURED

C LEADER REFERENCE TO LEADER DOING THE PLANNING

C IPFORCE A LIST OF GUARDS TO BE ASSIGNED TO SECURE THE ROOM

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (PNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCBECT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/

C

C INITIALIZE INDEX OF ITERATION

11 ITER = 1

C INITIALIZE NEIGHBOR LIST INDEX

12 KN = NULL

C GET FIRST PERSON TO BE ASSIGNED A POSITION

13 IPLJSN = IFIRST(IPFORCE,KP)

C GET LIST OF PORTALS TO BCCM. (THE LIST OF THE ROOM'S NEIGHBORS
C WILL BE USED AS A LIST OF ITS PORTALS. STRICTLY SPEAKING A NEIGHBOR
C DOES NOT HAVE TO BE A PORTAL, BUT IF IT ISN'T, IT IS PROBABLY STILL
C WORTH GUARDEING.)

14 5 NBES = IVAL(JNEBS,IRCCE)

FSN MODEL - SECURE SUBROUTINE

ISN

```

C TEST WHETHER HAVE RUN OUT OF GUARDS TO ASSIGN
15 10      IF (IPZBSN .EQ. NULL) RETURN
16          IF (KN .NE. NULL) GO TO 12
C POSITION THE FIRST GUARD (OF EACH CYCLE OVER POSNS) TO BE INSIDE THE
C ROOM
17          IPOSN = IROCM
C GET NEXT PORTAL
18          NBR = IPIFIRST(NEFS, KN)
19          GO TO 15
12          CALL PARSEBP(NEB, IDTYP, IDUM, IDUM)
C ONLY PROCESS DOORS ON THE FIRST ITERATION, TO GIVE THEM HIGHER
C PRIORITY.
21          IF (IDTYP .NE. LDOOR .AND. ITER .LE. 1) GO TO 30
C STATION GUARD AT NEIGHBOR, BUT OUTSIDE OF PORTAL, IF NEIGHBOR
C IS A DOOR
22          IPCSM = NER
23          IF (IDTYP .EQ. LDOOR .OR. IDTYP .EQ. LWIND) IPOSN =
+            IVAL(JECBT, NEF)
C
C CREATE INSTRUCTIONS TO MOVE TO THE LOCATION IDENTIFIED
24 15          IACT = NEWRC6(LACTB, KMOVNG, IPOSN, NULL, NULL, NULL, IPERSN)
25          IMSG = NEWRC5(LMESS, LEADER, IPERSN, JPLANS, TMIN, IACT)
C SEND THE INSTRUCTIONS TO THE PLAYER
26          CALL COMM(LEADER, IEEEFSN, IMSG)
C GET NEXT PLAYER
27          IPERSN = NEXT(IFORCE, KP)
C GET ROOM'S NEXT NEIGHBOR
28 30          NBR = NEXT(NEFS, KN)
29          IF (NBR .NE. NULL) GO TO 10
C HAVE RUN OUT OF NEIGHBORS OF THE ROOM BUT NOT OUT OF PERSONS;
C MAKE ANOTHER ITERATION OVER NEIGHBORS.
30          ITER = ITER + 1
31          GO TO 5
32          END

```

FSR MODEL - SENSE SUBROUTINE

ISN

1

SUBROUTINE SENSE(IREF)

C
C ROUTINE TO EXAMINE RECENT SENSOR STIMULI AND CONVERT THEM TO
C A LIST OF ENTITIES OBSERVED BY THE SENSOR MONITOR. THE ROUTINE
C UPDATES OLD OBSERVATIONS BY RECORDING SENSORS AND
C EXAMINES NEW SENSOR DETECTIONS TO FORM THE LIST. SOME IMPORTANT
C VARIABLES ARE:

C

IREF -> REFERENCE TO THE SENSOR MONITOR

C

C
C NWSNLT -- REFERENCE TO THE LIST OF NEW SENSOR ACTIVATIONS
C (FROM SNSACT)C
C NOOLDN -- REFERENCE TO THE LIST OF OLD (RECORDED) ACTIVATIONS

C

2

COMMON /STATEV/
DIMENSION ITEM(41900),DITEM(41900)
EQUIVALENCE (DTMIN,IREF(1),DITEM(1))
COMMON /PARS/
EQUIVALENCE (FNOLL,NOLL),(IFAIL,FAIL)

3

REAL*8 DTPNAM,FLDNAM,FCBNM

4

COMMON /PARS1/

5

COMMON /PARS2/

6

COMMON /PARS3/

7

COMMON /DATAV/

8

DIMENSION ACTTIM(25)

9

EQUIVALENCE (ACTTIM(1),ACTRAT(1))

10

COMMON /RECREF/

11

DIMENSION RECBFQ(140)

12

EQUIVALENCE (RECBFQ(1),ICCALLS)

13

NDET = 0

C

C PROCESS THE NEW SENSOR ACTIVATIONS

C

C GET THE PLACE OF THE SENSOR MONITOR

14

IPLC = IVAL(JPLACE,IVAL(JLCNN,IREF))

FSN MODEL - SENSE SUBROUTINE

19 MSGREF = IFIRST(NWSNLT,INW)
20 100 IF (MSGREF .EQ. NULL) GO TO 300
C IF A RECORDING SENSOR, ADD THE MESSAGE TO THE OLD SENSOR ACTIVATION
C LIST
21 NSEN = IVAL(JSCOURC,MSGREF)
22 NRPTYP = IVAL(JRTYPE,NSEN)
23 IF (NRPTYP .NE. KEINRC .AND. NRPTYP .NE. KPICRC) GO TO 200
24 CALL CHGLST(MSGREF,1,11,0)
25 NOLDSN = ISTACK(MSGREF,NCLDSN)
C MAKE SURE THE SENSOR MONITOR IS AT THE DESTINATION OF THE MESSAGE
26 200 IF (IPIC .NE. IVAL(JINCCE,NSEN)) GO TO 290
C ADD THE NEW SENSOR DETECTION TO THE LIST OF DETECTIONS BY THE
C SENSOR MONITOR,
C INDEXED BY THE TYPE OF THE SENSOR
27 NDET = NDET + 1
28 CALL CHGLST(MSGREF,1,4,NEPTYP)
29 INNCBS(NRPTYP) = ISTACK(MSGREF,INNOBS(NRPTYP))
C GET NEXT SENSOR ACTIVATION MESSAGE
30 290 MSGREF = NEXT(NWSNLT,INW)
31 GO TO 100
C
C COMPARE THE NUMBER OF DETECTIONS BY THE SENSOR MONITOR WITH THE
C MAXIMUM ALLOWED
C
C IF THE NUMBER OF ENTITIES DETECTED IS AT LEAST THE MAXIMUM
C NUMBER THE SENSOR MONITOR CAN ASSIMILATE, DONE.
32 300 NIDET = DTSEC * HELIPS(KIIM2,ISIDE)
33 IF (NDET .GE. NIDET) GO TO 900
C
C PROCESS OLD DETECTIONS
C
34 500 CONTINUE
35 MSGREF = IFIRST(NCLDSN,ICLD)
36 600 IF (MSGREF .EQ. NULL) GO TO 900
C MAKE SURE THE SENSOR MONITOR IS AT THE DESTINATION OF THE MESSAGE
37 IF (IPIC .NE. IVAL(JCCNT, MSGREF)) GO TO 690

FSN MODEL - SENSE SUBROUTINE

ISN

```
C     IF THE MESSAGE HAS THE CURRENT TIME IT IS ALREADY AN OBSERVATION OF
C     THE SENSOR MONITOR
38    TIME = VAL(JTIME,MSGREF)
39    AGEM = TMIN - TIME
40    IF (AGEM .LT. .1) GC TO 690
C     IF THE MESSAGE IS TOO OLD, ELIMINATE IT FROM THE LIST
41    IF (AGEM/60. .LT. TSTALE) GO TO 605
42    CALL CHGLST(IOLD,3,11,0)
43    CALL DELIST(IOLD,0,NCLDSN)
44    GC TO 690
C     IF THE SENSOR IS A FIXTURE SENSOR, SEE IF THE SUBJECT IS STILL
C     WITHIN RANGE
45    605 NSEN = IVAL(JSURC,MSGREF)
46    IF (IVAL(JRTYPE,NSEN) .NE. KPICRC) GO TO 670
C     GET THE PLAYER DETECTED AND HIS CURRENT PLACE
47    NPLARP = IVAL(JSUBJ,MSGREF)
48    NPLAC = IVAL(JPLACE,IVAI(JLCCN,NPLARP))
C     ITERATE THROUGH THE LIST OF SENSORS COVERING THE PLACE, CHECKING
C     FOR COVERAGE BY THIS SENSOR
49    LSNS = IVAL(JSENS,NPLAC)
50    NSN = IFIRST(LSNS,ISN)
51    610 IF (NSN .EQ. NULL) GC TO 630
52    IF (NSN .NE. NSEN) GO TO 630
C     CHECK TO SEE IF PLAYER IS CLOSE ENOUGH TO BE DETECTED
53    IF (DIST(NSEN,NPLARP) .GT. SENRNG(IVAL(JTYPE,NSEN))) GO TO 690
C     THE PLAYER IS STILL COVERED BY THE SENSOR; UPDATE THE TIME OF
C     OBSERVATION TO CURRENT TIME
54    CALL CHGFLD(JTIME,MSGREF,TSEC,1,NULL)
55    GO TO 670
56    630 NSN = NEXT(LSNS,ISN)
57    GC TO 610
C     ADD THE MESSAGE TO THE SM'S OBSERVATION LIST
58    670 NRPTYP = IVAL(JRTYPE,NSEN)
C     IF THE PLAYER IS ALREADY ON THE OBS LIST, SKIP IT
59    IF (LSEARCH(IMNOES(NRPTYP),NULL,NULL,JSUBJ,NULL) .NE. NULL) GO TO 690
60    CALL CHGLST(MSGREF,1,4,NEPTYP)
```

FSM MODEL - SENSE SUBROUTINE

ISN

61 IMNCBS(NRPTYP) = ISTACK(ESGREP,IMNOBS(NRPTYP))
62 690 MSGREF = NEXT(NCLDSN,ICLD)
63 GO TO 600

C

C NULL THE NEW MESSAGE LIST REFERENCE
64 900 NWSNLT = NULL
65 CALL CHGVAR(15,NWSNLT)
66 RETURN
67 END

FSN MODEL - SNSACT SUBROUTINE

1 ISN
1 SUBROUTINE SNSACT(IPLARF,IOLDLC)
C
C SUBROUTINE TO CREATE SENSOR ACTIVATION MESSAGES TO THE SENSOR
C MONITOR. THE ROUTINE IS CALLED FROM THE MOVEMENT MODULE WHENEVER
C A MOVING PLAYER CROSSES A NEW NODE.
C IPLARF -> REFERENCE TO PLAYER MOVING
C IOLDLC -> REFERENCE TO NODE (PLACE) PLAYER HAS JUST LEFT
C (USEFUL FOR CHECKING PORTAL SENSOR ACTIVATION)
C
C THE ROUTINE PLACES THE SENSOR ACTIVATIONS IN A MESSAGE LIST,
C REFERRED TO BY NMSLT. THIS LIST IS LATER PROCESSED IN THE
C ROUTINE SENSE TO CREATE A LIST OF OBSERVATIONS BY THE SENSOR
C MONITOR.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNOLI,NOLI),(IPAIL,FAIL)
7 REAL*8 DTPNAM,FDNAM,FCMCT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
11 COMMON /DATAV/
12 DIMENSION ACTTIM(25)
13 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
14 COMMON /RECREF/
15 DIMENSION RECBFQ(140)
16 EQUIVALENCE (RECBFQ(1),IGCALS)
C
C GET I'S PLACE
17 IPLAC = IVAL(JPLACE,IVAL(JLCCN,IPLARF))
C SEE IF COVERED BY SENSORS - IF NOT, DONE
18 MSENS = IVAL(JSENS,IEIAC)
19 IF (MSENS .EQ. NULL) RETURN

FSN MODEL - SNSACT SUBROUTINE

ISN C ITERATE THROUGH THE LIST OF SENSORS COVERING THIS NODE
 20 NSEN = IFIRST(MSENS,INC)
 21 100 IF (NSEN .EQ. NULL) RETURN
 C CHECK TO SEE IF THIS IS INDEED A SENSOR
 22 CALL PARSBP(NSEN,IDS,IDEU,IDUM)
 23 IF (IDS .EQ. LSENS) GO TO 110
 24 CALL EBR(3,29,ISENS,BSEN,0)
 25 GO TO 200
 C IF THE SENSOR IS NOT OPERATIONAL DONE WITH THIS SENSOR
 26 110 IF (IVAL(JPSTAT,NSEN) .NE. KOPERL) GO TO 200
 C FIND THE TYPE OF THE SENSOR AND CHECK TO SEE IF THE PLAYER IS
 C WITHIN RANGE OF THE SENSOR (IF NOT A PORTAL SENSOR)
 27 NSTYP = IVAL(JTYPE,NSEN)
 28 IF (NSTYP .EQ. KPORSN) GC TO 40
 29 R = DIST(NSEN,IPLARE)
 30 IF (R .GT. SENENG(NSTYP)) GO TO 200
 C BRANCH TO THE CHECK FOR THIS TYPE OF SENSOR
 31 GO TO (10,20,30,40,50,60,70),NSTYP
 C EXPLOSIVES SENSORS - CURRENTLY NOT IMPLEMENTED
 32 10 CONTINUE
 33 GO TO 200
 C METAL SENSOR - CHECK PLAYF'S EQUIPMENT FOR WEAPONS
 34 IEQULT = IVAL(JWEAPS,IPL&P)
 35 IEQUIP = IFIRST(IEQULT,IKUM)
 36 25 IF (IEQUIP .EQ. NULL) GO TO 200
 37 CALL PARSRP(IEQUIP,IDDD,IRRR,IDUM)
 38 IF (IDDD .NE. IWEAP) GC TO 29
 C CHECK TO SEE IF EQUIPMENT IS METAL
 39 IF (IVAL(JTYPE,IEQUIP) .LE. KGRENAD) GO TO 29
 C THE MATERIAL IS METAL- IF THE SENSOR
 40 GO TO 150
 41 29 IEQUIP = NEXT(IEQULT,IKUM)
 42 GO TO 25
 C RADIOACTIVITY SENSOR - ITERATE THROUGH THE PLAYER'S EQUIPMENT
 43 IEQULT = IVAL(JWEAPS,IPL&P)
 44 IEQUIP = IFIRST(IEQULT,IKUM)

FSN MODEL -- SNSACT SUBROUTINE

15N
45 35 IF (IEQUIP .EQ. NULL) GO TO 200
46 CALL FARSRP(IEQUIP,IEEDD,IBRR,IDUM)
47 IF (IEEDD .NE. LEQUIP) GO TO 39
 C CHECK IC SEE IF EQUIPMENT IS NUCLEAR MATERIAL
48 IF (IVAL(JTYPE,IEQUIP) .NE. KSNM) GO TO 39
 C THE MATERIAL IS NUCLEAR- TRIP THE SENSOR
49 GO TO 150
50 39 IEQUIE = NEXT(IEQUIT,ISUE)
51 GO TO 35
 C PORTAL STATUS SENSOR - SEE IF I HAS PASSED THROUGH
52 40 CONTINUE
 C IF I'S CURRENT LOCATION IS NOT A PORTAL, NO ACTIVATION
53 CALL FARSRF(IPLAC,IDLTP, IDUM, IDUM)
54 IF (IDLTP .NE. IDCOR .AND. IDLTP .NE. LWIND) GO TO 200
 C IF HIS PREVIOUS LOCATION IS A PORTAL PAIRED WITH THIS ONE,
 C TRIP THE SENSOR
55 ITM=IVAL(JSOURC,ICIDIC)
56 IF (IVAL(JPORT,IPLAC) .NE. ITM) GO TO 200
57 GO TO 150
 C PERSON-VEHICLE SENSOR 1
58 50 CONTINUE
59 GO TO 150
 C PERSON-VEHICLE SENSOR 2
60 60 CONTINUE
61 GO TO 150
 C CREDENTIALS SENSOR - ACTIVATE ONLY IF PLAYER LACKS PROPER
 C CREDENTIALS
62 70 CONTINUE
63 IF (IVAL(JCRED,IPLARP) .NE. NULL) GO TO 200
64 150 CONTINUE
 C THE SENSOR IS TRIPPED - CREATE A MESSAGE
65 IPLAC = ICOPY(IVAL(JLCRN,IPLARP))
66 MSGRF = NEWRCS(LMESS,NSEN,IPLARP,JIOCN,TMIN,IPLAC)
 C SAVE THE REFERENCE TO THE NEW MESSAGE
67 CALL CHGLST(MSGRF,1,12,0)
68 NWSNLT = ISTACK(MSGRF,BWSNLT)

FSN MODEL - SNSACT SUBROUTINE

ISN

C GET NEXT SENSOR COVERING THIS NODE
69 200 NSEN = NEXT(MSENS,INC)
70 GC TO 100
71 END

FSN MODEL - SURFAC SUBROUTINE

ISN

```
1      SUBROUTINE SURFAC(IPLARP)
C ... THIS SUBROUTINE IS CALLED TO CHANGE AN INSIDER TO A
C     NOEMAL AFE COMEATANT.
C
C           IPLARP >-- EEFERENCE TO PLAYER TO SURFACE
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      CCMMCN /PARS/
6      EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FC6PCT
8      COMMON /PARS1/
9      COMMNCN /PARS2/
10     COMMNCN /PARS3/
11     CCMECN /RECREF/
12     COMMON /NEW/
13     DIMENSION INSTAT(2,5)
14     EQUIVALENCE (IOESV(41),INSTAT(1,1))
15     LCGICAL INTCPT
C
C ... TEMEOFARY INITIALIZATION, REMOVE WHEN PARSN UPDATED
C
16     NJNSDR = 5
17     KINSDR = 4
C
C ... IF EAYER NOT INSIDER, RETURN
C
18     IF(IVAL(JTYPE,IPLARP) .EQ. KINSDR) GO TO 10
19     CALL ERB(41,37,IPLARP,NULL,NULL)
20     BETJEN
C
C ... REMOVE PLAYER FROM INSIDER ARRAY
C
21     10 DO 20 I=1,NINSDR
```

FSN MODEL - SURFAC SUBROUTINE

1 SN

22 IF(INSTAT(1,I) .NE. IPLARF) GO TO 20
23 INSTAT(1,I) = NULL
24 INSTAT(2,I) = 0
25 GO TO 30
26 20 CCNTINUE

C

C ... ALLOW INSIDER TO EXIT ON ANY APE NETS ON WHICH HE
C CAN CURRENTLY RECEIVE.

C

27 30 NNET = NRCS0(LCOMM1)
28 DO 40 INET = 1,BNET
29 NETREF=NEWREF(LCOMM1,INET,0)
C SEE IF PLAYER ON NET

C

30 ION=LSEBCH(IVAL(JRCVES,NETREF),IPLARF,NULL,NULL,NJLL)
31 IF(ION.EQ.NULL) GC TC 40

C

C ... SEE IF THIS IS APE NET

C

32 CALL DRCOM3(IPLARF,NETREF,INTCPT)
33 IF(INTCPT) GC TC 40

C

C ... ADD NET TO PLAYE'S LIST

C

34 NETLIS = IVAL(JCMNTS,IPLARF)
35 NETLIS = ISTACK(NETREF,NETLIS)
36 CALL CHGFLD(JCMNTS,IPLARF,NETLIS,1,1DUM)

37 40 CONTINUE

C

C ... FIND A FORCE FOR THE INSIDER

C

38 NFORCE = NRCS0(LFORCE)
39 DO 50 I=1,NFCBCE
40 IFREF = NEWREF(LFCRCE,I,0)
41 IF(IVAL(JFALEG,IFREF) .EQ. KAFE) GO TO 60

42 50 CONTINUE

FSN MODEL - SURFAC SUBROUTINE

ISN

```
C
C ... NO AFE FORCE FOUND --CREATE FORCE WITH ONLY INSIDER ON IT
C AND SET GOALS FLAG FOR PLANNING
C
43      NEWF = NEWREC(IFCRCE)
44      IFREF = NEWREF(LFORCE,NEWF,0)
45      IPTR = IREP1(IFREF)
46      ITEM(IPTR+JTYPE) = KEC1
47      ITEM(IPTR+JFALEG) = KAFE
48      ITEM(IPTR+JFLDR) = IFIARE
49      ITEM(IPTR+JCCNIS) = IELABF
50      ITEM(IPTR+JSITN) = KSITN3
51      DITEM(IPTR+JPSTAT) = 1.0
52      ITEM(IPTR+JPLANS) = IVAL(JPLANS,IPLABF)
53      NEWGOL = 1
C      MAKE SUPERORDINATES FIELD OF INSIDER NON-NULL, FOR PLANNING PURPOSES
C      AND ADD THE FORCE REFERENCE TO HIS FORCE FIELD
54      CALL CHGFLD(JSUBS,IPLABF,IPLABF,1,NULL)
55      CALL CHGFLD(JFCRCE,IFIARE,IFREF,1,NULL)
56      GO TO 70
C
C ... UPDATE FORCE RECORD AND IPLABF TO SHOW FORCE
C MEMBERSHIP
57      60 LISMEM = IVAL(JCONTS,IFREF)
58      LISMEM = ISTACK(IPLABF,LISMEM)
59      CALL CHGFLD(JCONTS,IFREF,LISMEM,1,IDLUM)
60      CALL CHGFLD(JFCRCE,IFIARE,IPREF,1,IDLUM)
61      LDRREF = IVAL(JFLDR,IFREF)
62      CALL CHGFLD(JLEADR,IFIARE,LDRREF,1,IDLUM)
C
C      ADD INSIDER TO SUPERORDINATES FIELD OF FORCE LEADER
C
63      LISSUE = IVAL(JSUBS,IDLREF)
64      LISSUE = ISTACK(IPLABF,LISSUB)
65      CALL CHGFLD(JSUBS,LDREF,LISSUB,1,IDLUM)
C
```

PSM MODEL - SURFAC SUBROUTINE

ISN

C ... CHANGE TYPE TO COMBATANT
C
66 70 CALL CHGFLD(JTYPE,IPLABF,KCOMBT,1, IDUM)
67 RETURN
68 END

```
FSN MODEL - TRAVEL FUNCTION
ISN
1      FUNCTION TRAVEL(IPLARF,ISOU,ISIN,IACSUB,IPOSUB)
C
C
C -- THE PURPOSE OF THIS FUNCTION IS :::
C     1. EVALUATE A MOVEEMENT LINK FOR THE TRAVEL TIME
C        IN SECCNDS.
C
C
C -- INPUT PARAMETERS
C
C     IPLARF -- PLAYER REFERENCE
C     ISOU   -- SOURCE NODE
C     ISIN   -- SINK NCDE
C     IACSOE -- ACTIVITY SUBTYPE
C
C
C -- OUTPUT
C
C
C     FUNCTION RETURNS THE TIME IN SECONDS NEEDED
C     TO TEAVERSE A GIVEN LINE FOR PLAYER IPLARF
C     IF PLAYER IS PENETRATING A PORTAL IPOSUB IS THE
C        SUE ACTIVITY
C{
C
C -- PROCEDURE
C
C     1. DETERMINE ACTIVITY RATE FOR ACTIVITY SUB-TYPE
C
C     2. ADJUST RATE ACCORDING TO WOUND, SKILL.
C
C     3. DETERMINE DISTANCE IN METERS
C
C     4. TRAVEL=DISTANCE/RATE
C
```

FSN MCDEL - TRAVEL FUNCTION

ISN

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (IFAIL,FAIL)
4 REAL*8 DTPNAM,FIDNAM,ECENCT
5 COMMON /PARS1/
6 COMMON /PARS2/
7 COMMON /PARS3/
8 COMMON /DATAV/
9 DIMENSION ACTTIM(25)
10 EQUIVALENCE (ACTTIM(1),ACTRAT(1))

C -- FIND SKILL FOR PLAYER
C -- SKILL TYPE = KMOVSK
C -- FIND PHYSICAL STATUS
C

11 ITEMP=KEWHOL
12 IF(IVAL(JPSTAT,IPLARF) .EQ. KWOUND) ITEMP=KPWGUN

C -- FIND PLAYER NUMBER
C

13 CALL PARSBF(IPLARF, IDY, IFL'NO, IDUM)

C CHECK IF PASSING THROUGH PORTAL
C

14 CALL PARSBF(ISOU, IDY, IRR, IDU)
15 IF(IDY .EQ. IDCCR) GO TO 10
16 IF(IDY .EQ. LWIND) GO TO 10
17 GO TO 100

C

18 10 CALL FARSRF(ISIN, IDY, IFF, IDU)
19 IF(IFY .EQ. LDOOR) GO TO 12
20 IF(IFY .EQ. LWIND) GO TO 12
21 GO TO 100

C -- DETERMINE IF PORTAL

PSN MODEI - TRAVEL FUNCTION

ISN

```
C
22      12 IF{ISOU .NE. IVAL(JPORT,ISIN)} GO TO 100
C
C -- PASSING THROUGH PORTAL
C     DETERMINE TIME REQUIRED
C
23      IF{IVAL(JSTAT,ISCU) .IE. KOPEN} GO TO 15
24      IF{IVAL(JSTAT,ISOU) .EQ. KCLOSD} GO TO 20
25      IF{IVAL(JSTAT,ISOU) .EQ. KLCCKD} GO TO 30
26      GO TO 40
C
C -- PORTAL OPEN
C
27      15 TEM=ACTITIM(KPPOPE)
28      IF{IACTTP(KPPOPE) .EQ. 1} TEM=1./ACTRAT(KPPOPE)
29      IPOSUE=KPPOPE
30      GO TO 70
C
C -- PORTAL CLOSED
C
31      20 TEM=ACTITIM(KPPUNL)
32      IF{IACTTP(KPPUNL) .EQ. 1} TEM=1./ACTRAT(KPPUNL)
33      IPOSUE=KPPUNL
34      GO TO 70
C
C -- PORTAL LOCKED
C     USE KEY
C
35      30 CALL KEYS(IPLABF,ISCU,IE(BF)
36      IF{IEQBF .EQ. NULL} GO TO 40
C
37      TEM=ACTITIM(KPPKEY)
38      IF{IACTTP(KPPKEY) .EQ. 1} TEM=1./ACTRAT(KPPKEY)
39      IPOSUE=KPPKEY
40      GO TO 70
C
```

FSN MODEL - TRAVEL FUNCTION

1 ISN

2 C -- SCALE PORTAL

3 C

4 41 40 IF(IDY .EQ. LWIND) GO TO 50

5 42 IF(IVAL(JBABIN,ISCU) .EQ. NULL) GO TO 50

6 43 IEABR=IVAL(JBABIN,ISCU)

7 44 CALL SCALE(IPLABF,IEABR,IEQBF,AHEGT)

8 45 IF(IEQBF .EQ. NULL) GO TO 50

9 C

10 46 TEM=AHEGT/ACTRAT(KSCALE)

11 47 IF(IACTTP(KSCALE) .EQ. 1) TEM=AHEGT*ACCTIM(KSCALE)

12 48 IPOSUE=KSCALE

13 49 GO TO 70

14 C

15 C -- PORTAL SECURED

16 C TEST FOR EQUIPMENT TO BREACH

17 C

18 50 CALL BRECH(IPLABF,ISOU,IEQBF)

19 51 IF(IEQBF .EQ. NULL) GC TO 60

20 C

21 52 TEM=ACCTIM(KPPWEP)

22 53 IF(IACTTP(KPPWEP) .EQ. 1) TEM=1./ACTRAT(KPPWEP)

23 54 IPOSUE=KPPWEP

24 55 GO TO 70

25 C

26 C -- IMPOSSIBLE TO PENETRATE FORTAL

27 C

28 56 60 TRAVEL=9999.0

29 57 GO TO 200

30 C

31 58 70 TRAVEL=TEM*SKILL(KMCVSK,ITEMP,IELANO)

32 59 GO TO 200

33 C

34 C -- FIND ACTIVITY RATE FOR MCVING

35 C

36 60 100 RATE=ACTRAT(IACSUE)

37 61 IF(IACTTP(IACSUE) .EQ. 2) RATE=1./RATE

FSN MCDEL - TRAVEL FUNCTION

```
1 SN
2
3 RATE=RATE*SKILL(KMOVE,ITEMP,IFLANDO)
C
C -- CALCULATE DISTANCE
C
4 X1=(VAL(JXCO,ISIN)-VAL(JXCO,ISOU))**2
5 X2=(VAL(JYCO,ISIN)-VAL(JYCO,ISOU))**2
6 X3=(VAL(JZCO,ISIN)-VAL(JZCO,ISOU))**2
7 DIST=SQRT(X1+X2+X3)
C
C -- FIND TRAVEL TIME
C
8 TRAVEL=DIST/RATE
C
9 200 CONTINUE
10 RETURN
11 END
```

FSN MODEL - TRGLST SUBROUTINE

ISN

1

SUBROUTINE TRGLST(IPLARF,ILCCN,ITARLT)

C

C

C -- THE PURPOSE OF THIS SUBROUTINE IS TO:

C 1. GENERATE A LIST OF POTENTIAL TARGETS FOR PLAYER
C IPLARF.

C

C

C -- INPUT PARAMETERS

C

C

C IPLARF - PLAYER PREFERENCE

C

C ILCCN

C IF ILCCN IS NULL, TARGETS LOCATION IS
C ANYWHERE WITH LINE OF SIGHT

C IF ILOCN IS NOT-NUL, TARGETS MUST BE IN
C THIS DEFINED REGION

C

C -- OUTPUT VARIABLES

C

C ITARLT - LIST PREFERENCE FOR TARGET LIST

C

C

C TARGET LIST GENERATION

C ANY PLAYER PERCEIVED OF OPPOSITE ALLEGIANCE, AND
C THE PERCEIVED LOCATION IS IN THE REGION OF INTEREST
C IS DEFINED AS A POTENTIAL TARGET. ONLY IF THE
C PERCEIVED LOCATION IS THE SAME AS THE TRUE LOCATION
C OF THE PERCEIVED PLAYER AND LINE OF SIGHT EXISTS
C WILL THIS PLAYER BE ADDED TO THE TARGET LIST.

C

C

C

FSN MODEL - TRGLST SUBROUTINE

1SN

```
2      CCMECN /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,XTEP(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCRMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11
12     C   -- INITIALIZE TARGET LIST
13     C
14     11 ITAFLT=NULL
15
16     C   FIN'L LINE OF SIGHT LIST
17     C
18     12 CALL PARSRP(IPLABF, IDY, IRE, IDUM)
19     13 ILOSL=LOSLIS(I$LABF,IRE)
20
21     C   -- FIND RECEIVED LIST OF PLAYERS
22     C
23     14 IPEBLT=IVAL(JPRCES,IFLAGF)
24     15 ISIDE=IVAL(JALLEG,IPLABF)
25
26     C   -- FIND FIRST PERCEPTION RECORD REFERENCE
27     C   ON LIST OF PERCEPTIONS
28     C
29     16 IPEBRF=IPIRST(IPERIT,KT)
30     17      5 IF(IPEBRF .EQ. NULL) GO TO 30
31
32     C   -- FIND PLAYER REFERENCE OF PERCEPTION
33     C
34     18 ICURRF=IVAL(JVIEW,IFERBF)
35     19 CALL PARSEF(ICURRF, IDYY, IRRN, IDDUM)
36     20 IF(IDYY .EQ. LFCRCE) GO TO 20
37
38     C
```

FSN FCDL - TRGLST SUBROUTINE

ISN

```

C -- TEST IF ICURRF IS OF OPPCSITE ALLEGIANCE
C
21      IF(IVAL(JALLEG,ICURRF) .EQ. NULL) GO TO 20
22      IF(IVAL(JALLEG,ICURRF) .EQ. ISIDE) GO TO 20
C
C -- PLAYER PERCEIVED CP CEECSITE ALLEGIANCE, TEST IF
C     PLAYER IS IN REGION OF INTEREST
C
23      IFLOC=IVAL(JLOCN,ICURRF)
24      IF(IPLOC .EQ. NULL) GO TO 20
25      IF(ILOCN .EQ. NULL) GO TO 10
26      IF(IVAL(JPLACE,IPLOC) .NE. ILOCN) GO TO 20
C
C -- PLAYER IN REGION OF INTEREST , TEST TRUE LOCATION
C
27      ITBURF=IVAL(JIC,IPERRF)
28      ITLCC=IVAL(JLCNN,ITBURF)
29      ITEMPI=IVAL(JPLACE,ITLOC)
30      IF(IVAL(JPLACE,IEICC) .NE. ITEMPI) GO TO 20
C
C TEST IF PLAYER IS BEING CAPTURED
C
31      IACTL=IVAL(JACTIV,ITRUFF)
32      IAT=IFIRST(IACTL,KR)
33      IF(IAT .EQ. NULL) GO TO 17
34      IF(IVAL(JTYPE,IAT) .NE. ECAPNG) GO TO 15
35      IF(IVAL(JPAR1,IAT) .EQ. ITRURF) GO TO 20
C
36      IAT=NEXT(IACTL,KR)
37      GO TO 12
C
C
C -- PLAYER IS IN SAME REGION AS PERCEPTION
C     TEST FOR LINE OF SIGHT
C
C SEARCH LCS LIST

```

FSN MODEL - TRGLST SUBROUTINE

1SN

C

38 17 ICOD=IFIRST(ILOSL,KS)
39 18 IF(ICOD .EQ. NULL) GO TO 20
40 IF(ICOD .EQ. ITBURF) GO TO 19
41 ICOD=NEXT(ILCSL,KS)
42 GO TO 18

C

C -- ADD PLAYER TO TARGET LIST
C LINE OF SIGHT EXISTS

C

43 19 ITARLT=ISTACK(ITBURF,ITARLT)

C

C -- NEXT PERCEPTION

C

44 20 IPERLT=NEXT(IPERLT,KT)
45 GO TO 5

C

C

C

46 30 CONTINUE
47 RETURN
48 END

FSN MODEL - UPACTS SUBROUTINE

ISM

1 SUBROUTINE UPACTS

C
C -- THE PURPOSE OF THIS SUBROUTINE IS TO:
C 1. REMOVE ALL ACTIVITIES FROM EACH PLAYER
C LIST EXCEPT CAPTURING
C 2. AND, REMOVE PLANS THAT HAVE BEEN COMPLETED
C
C

C -- OUTPUT
C THIS SUBROUTINE ADJUSTS THE ACTIVITY AND PLANS
C LIST OF PLAYERS
C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNUII,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FDNAM,FCRRECT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /DATAV/
11 DIMENSION ACTTIM(25)
12 EQUIVALENCE (ACTTIM(1),ACTRAT(1))
13 COMMON /RECREF/
14 DIMENSION RECREFQ(140)
15 EQUIVALENCE (RECREFQ(1),IGCALS)

C

C

16 JL=JGARDS
17 5 IPBLT=IVAL(JL,ISITE)
18 IPLARF=IPIRST(IERIT,KA)
19 10 IF(IPLARF .EQ. NULL) GO TO 40
20 INEWLT=NULL
21 IACTLT=IVAL(JACTIV,IPLARF)
22 IACTRF=IFIRST(IACTIT,KB)

FSN MODEL - UPACTS SUBROUTINE

ISN
23 20 IF(IACTRF .EQ. NULL) GO TO 30
24 IF(IVAL(JTYPE,IACTRF) .NE. KCAPNG) GO TO 25
C
C TEST IF EITHER PLAYER IS DEAD OR CAPTURED
C
25 ISUR=IVAL(JPAR1,IACTRF)
26 IF(IVAL(JPSTAT,ISUR) .LE. KCAPTE) GO TO 25
27 ICAP=IVAL(JDCER,IACTRF)
28 IF(IVAL(JPSTAT,ICAP) .LE. KCAPTR) GO TO 25
29 INEWLT=IQUEUE(IACTRF,INEWLT)
30 25 IACTRF=NEXT(IACTLT,KE)
31 GC TO 20
32 30 IPTB=IBEF1(IPLARF)
33 ITEM(IETR+JACTIV)=INEWLT
C
C -- CHECK PLANS
C
34 ILST=IVAL(JPLANS,IPLARF)
35 IPLAN=IFIRST(ILST,KP)
36 IF(IPLAN .EQ. NULL) GO TO 38
37 IF(IVAL(JTYPE,IPLAN) .EQ. KMOVNG) GO TO 35
38 IF(IVAL(JTYPE,IPLAN) .EQ. KDETNG) GO TO 32
C
C -- TEST FOR TERMINATION CONDITION
C
39 IRF=IVAL(JMCDS,IEIAN)
40 IF(IRF .EQ. NULL) GO TO 38
41 ICCD=NBTTEST(IRF)
42 IF(ICCD .NE. 1) GO TO 38
43 CALL DE_TST(KP,JEIABS,IPLARF)
44 ITT=IVAL(JPLANS,IELARF)
45 CALL CHGFID(JPLANS,IELARF,ITT,1,1DUM)
46 GO TO 38
C
C OBSERVATION PLAN , CHECK DURATION TIME
C

PSM MODEL - UPACTS SUBROUTINE

ISN
47 32 IF(IVAL(JPAR3,IPLAN) .EQ. NULL) GO TO 38
48 TT=VAL(JPAR3,IELAN)
49 TT=TT-DTSEC
50 IF(TT .LE. 0.0) GO TO 37
51 IPTR=IREF1(IPLAN)
52 DITEM(IPTR+JPAB3)=TT
53 GO TO 38

C
C -- CHECK LOCATION OF PLAYER
C
54 IDES=IVAL(JPAR1,IPLAN)
55 ILCC=IVAL(JLCCN,IELARF)
56 IPTRA=IREF1(IPLAN)
57 ITEM(IPTRA+JPAB2)=NULL
58 IF(IVAL(JSCURC,ILOC) .NE. IDES) GO TO 38
59 37 CALL DELIST(KP,JPLANS,IPLARF)
60 ITT=IVAL(JPLANS,IPLARF)
61 CALL CHGFID(JPLANS,IELARF,ITT,1,IDUM)
62 38 IPLARF=NEXT(IPBLT,KA)
63 GO TO 10
64 40 IF(JL .EQ. JADVRS) GO TO 50
65 JI=JADVES
66 GO TO 5
67 50 CONTINUE
68 RETURN
69 END

This subsection contains listing for the utility routines used by the Plex Preprocessor and the Fixed Site Neutralization Model. For convenience, they are presented in alphabetical order. For the most part, they are liberally commented. The reader may wish to consult section 3.3, in volume 1 of this manual, for definitions and discussions that are related to this logic. Basically, the utility routines provide record management, reference formation and analysis, storage and retrieval, list processing, and input/output functions that are used in connection with the plex data structure.

UTILITIES - CHGFLD SUBROUTINE

ISN

1

SUBROUTINE CHGFLD(IFIELD,IRECFL,IVALUE,IFLAG,ICHGLS)

C
C ROUTINE TO MAKE AND REPORT A CHANGE TO A RECORD FIELD'S
C VALUE. THE ROUTINE HAS SEVERAL ALTERNATIVE MODES OF ACTION WHICH
C DEPEND ON THE IFLAG PARAMETER. IT MAY EITHER MAKE THE CHANGE
C RIGHT AWAY OR DEFER IT UNTIL LATER, SAVING THE CHANGE SPECIFICATION
C IN A CHANGE RECORD. IT MAY EITHER OUTPUT A REPORT OF THE CHANGE
C OR NOT.

C
C I"PUT PARAMETERS:
C IFIELD INDEX OF FIELD WHICH IS TO BE CHANGED.
C IRECFL REFERENCE TO RECORD CONTAINING THE CHANGED FIELD
C IVALUE NEW VALUE WHICH THE FIELD IS ASSIGNED (MAY BE INTEGER,
C REAL, OR RECORD REFERENCE)
C IFLAG A FLAG INDICATING WHICH OPTION THE ROUTINE IS TO FOLLOW:
C -2 DEFER THE CHANGE, DON'T OUTPUT A REPORT
C -1 MAKE THE CHANGE IMMEDIATELY, DON'T OUTPUT A REPORT
C 1 MAKE THE CHANGE IMMEDIATELY, OUTPUT A REPORT
C 2 DEFER THE CHANGE, OUTPUT A REPORT
C ICHGLS IF THE CHANGE IS TO BE DEFERRED, ICHGLS GIVES THE
C RECORD INDEX OF THE FIRST CHANGE RECORD ON A LIST IN
C WHICH THE NEWLY CREATED CHANGE RECORD IS TO BE STACKED
C (OR HAS THE VALUE NULL IF THERE IS NO PRIOR CHANGE LIST).
C NOTE THAT CHANGE RECORDS HAVE A "NEXT" FIELD AND CAN BE
C STACKED VIA THAT FIELD WITHOUT RESORTING TO LISTENDS.

C
C OUTPUT PARAMETERS:
C ICHGLS (SEE ABOVE)

C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (PNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT

UTILITIES - CHGFLD SUBROUTINE

ISN

8 COMMON /PARS1/
9 COMMON /PARS3/

C

10 DIMENSION MCHNG(4,1)
11 EQUIVALENCE (VALUE,NEWVAL), (MCHNG(1),CHANGE(1))
12 DATA BLS/* */

C

C IF CHANGE IS TO BE OUTPUT, DO SO

C

13 IF (IFLAG .LT. 0) GO TO 40
C GET THE FIELD TYPE FOR THE FIELD
14 CALL PARSRF(IRECRF, IDTYP, IRECNO, IDUM)
C GET INDEX OF FIELD NAME
15 IFLDNM = IPDNPT(IDTYP) + IFIELD
C BRANCH ACCORDING TO FIELD TYPE
16 IPTYPE = IVAL(IFIELD, NEWREF(IDTYP, NRECS(IDTYP), 0))
17 GO TO (5,10,15,20), IPTYPE

C

C INTEGER FIELD
C (THE CURRENT VERSION OF THIS ROUTINE DOES NOT HANDLE NULL INTEGER
C FIELDS SPECIALLY).

C

18 5 IOLDVL = IVAL(IFIELD, IRECRF)
19 WRITE (6,100) FLDNAM(IFLDNM), DTPABR(IDTYP), IRECNO, IOLDVL, IVALUE
20 100 FORMAT ('FIELD ',A8,' OF ',A3,I4,' CHANGED FROM ',I8,' TO ',I8)
21 GO TO 40

C

C REAL FIELD
C (THE CURRENT VERSION OF THIS ROUTINE DOES NOT HANDLE NULL REAL
C FIELDS SPECIALLY).

C

22 10 OLDVAL = VAL(IFIELD, IRECFF)
23 IF (OLDVAL .EQ. FNNULL) CIDVAL = NULL
24 NEWVAL = IVALUE
25 IF (NEWVAL .EQ. NULL) VALUE = NULL
26 WRITE (6,105) FLDNAM(IFLDNM), DTPABR(IDTYP), IRECNO, OLDVAL, VALUE

UTILITIES - CHGFLD SUBROUTINE

ISN

```
27 105 FORMAT ('FIELD ',A8,' OF ',A3,I4,' CHANGED FROM ',F8.1,' TO ',
+           F8.1)
28      GO TO 40
C
C   REFERENCE FIELD
C
29 15   IOLDVL = IVAL(IFIELD,IRECNP)
30   IF (ICLDVL .NE. NULL) CALL PARSRP(IOLDVL, IDTPOV,IRECOV, IDUM)
31   IF (IVALE .NE. NULL) CALL PARSRP(IVALE, IDTPNV,IRECNV, IDUM)
32   IF (IOLDVL .EQ. NULL) GO TO 16
33   IF (IVALE .EQ. NULL) GO TO 18
34   WRITE (6,110) FLDNM(IFLDNM), DTPABR(IDTYP),IRECNO,DTPABR(IDTPOV),
+                 IRECOV,DTPABR(IDTPNV),IRECNV
35 110 FORMAT ('FIELD ',A8,' OF ',A3,I4,' CHANGED FROM ',A3,I4,1X,' TO ',
+           A3,I4)
36      GO TO 40
C
C   OLD VALUE IS NULL
37 16   IF (IVALE .EQ. NULL) GO TO 19
38   WRITE (6,111) FLDNM(IFLDNM), DTPABR(IDTYP),IRECNO,BLS,
+                 DTPABR(IDTENV),IRECNV
39 111 FORMAT ('FIELD ',A8,' OF ',A3,I4,' CHANGED FROM ',A8,' TO ',A3,I4)
40      GO TO 40
C
C   NEW VALUE IS NULL
41 18   WRITE (6,113) FLDNM(IFLDNM), DTPABR(IDTYP),IRECNO,DTPABR(IDTPOV),
+                 IRECOV,BLS
42 113 FORMAT ('FIELD ',A8,' OF ',A3,I4,' CHANGED FROM ',A3,I4,' TO ',A4)
43      GO TO 40
C
C   BOTH OLD AND NEW VALUES ARE NULL
44 19   WRITE (6,111) FLDNM(IFLDNM), DTPABR(IDTYP),IRECNO,BLS,BLS
45      GO TO 40
C
C   FIELD CAN HAVE ANY OF SEVERAL TYPES AS VALUE
C
```

UTILITIES - CHGFLD SUBROUTINE

ISN

```
C IS THIS A MESSAGE RECORD ?
46 20 IF (IDTYP .NE. 1MESS) GO TO 25
C IF SO, IT MUST BE THE CONTENT FIELD WHICH IS BEING CHANGED
47 IF (IFIELD .NE. JCONT) CALL ERR(20,15,IFIELD,IRECNO,IDTYP)
C GET THE DATA TYPE OF THE SUBJECT OF THE MESSAGE
48 CALL PARSRP(IVAL(JSUBJ,IRECRF),IDTPSB,1DUM,1DUM)
C GET THE INDEX OF THE ATTRIBUTE FIELD BEING UPDATED BY THE MESSAGE
49 IATTR = IVAL(JATTR,IRECRF)
C GET THE CONTENT DATA TYPE CORRESPONDING TO THE TYPE OF ATTRIBUTE
C FIELD
50 ICTYPE = IVAL(IATTR,NEWREF(IDTPSB,NRECS(IDTPSB),0))
C BRANCH ACCORDING TO THIS TYPE
51 GO TO (5,10,15),ICTYPE
C
C IS THIS AN ACTION RECORD ?
52 25 IF (IDTYP .NE. 1ACTN) GO TO 32
C IF SO, IS IT ONE OF THE PARAMETER FIELDS WHICH IS TO BE CHANGED ?
53 IF (IFIELD .LT. JPAR1 .OR. IFIELD .GT. JPAR3)
+ CALL ERR(20,15,IFIELD,IRECNO,IDTYP)
C DETERMINE THE TYPE OF THE PARAMETER CORRESPONDING TO THE ACTION
54 IACTYP = IVAL(JTYPE,IRECRF)
55 IF (IACTTP .LE. NACTTP) GO TO 30
56 CALL ERR(21,15,1,IACTTP,0)
57 GO TO 38
C
C GET DATA TYPE OF THE APPROPRIATE FIELD AND BRANCH ACCORDINGLY
58 30 IPTYPE = MACTTP(IFIELD-1,IACTTP)
59 GO TO (5,10,15,38,38),IPTYPE
C
C IS THIS A RELATION RECORD ? IS IT ONE OF ITS PARAMETER FIELDS
C WHICH IS TO BE CHANGED ?
60 32 IF (IDTYP .NE. 1RELN .OR. (IFIELD .NE. JPAR1 .AND. IFIELD .NE.
+ JPAR2)) CALL ERR(20,15,IFIELD,IRECNO,IDTYP)
C DETERMINE THE TYPE OF THE PARAMETER CORRESPONDING TO THE RELATION
61 IRELTP = IVAL(JTYPE,IRECRF)
62 IF (IRELTP .LE. NRELTP) GO TO 36
```

UTILITIES - CHGFLD SUBROUTINE

ISN

63 CALL ERB(21, 15, 2, IRELTP, 0)

64 GO TO 36

C

C GET DATATYPE OF THE APPROPRIATE FIELD AND BRANCH ACCORDINGLY

65 36 IFTYPE = IRELTP(IFIELD-1,IRELTP)

66 GO TO (5,10,15,38,38),IFTYPE

C

C IF CAN'T RECOGNIZE THE TYPE OF THE FIELD BASED ON ITS CONTEXT,
C BASE DECISION ON ITS VALUE

67 38 IFTYPE = 1

68 IF (IVALUE .LE. 1009999999 .AND. IVALUE .GE. 1000010000) + IFTYPE = 3

69 IF (IVALUE .GT. 1009999999 .OR. IVALUE .LT. -100) IFTYPE = 2

70 GO TO (5,10,15),IFTYPE

C

C IF A CHANGE IS TO BE MADE RIGHT AWAY, DO SO.

C

71 40 IF (IABS(IFLAG) .EQ. 2) GO TO 50

72 I = IREF(IFIELD,IRECFF)

73 ITEM(I) = IVALUE

74 RETURN

C

C DEFER MAKING THE CHANGE; STORE CHANGE IN A CHANGE RECORD

C

75 50 ICHG = NEWREC(LCHNG)

76 MCHNG(JSUBJ,ICHG) = IRECFF

77 MCHNG(JATTR,ICHG) = IFIELD

78 MCHNG(JNVAL,ICHG) = IVALUE

79 MCHNG(JNXT,ICHG) = ICHGLS

80 ICHGLS = ICHG

81 RETURN

82 END

UTILITIES - CHGLST SUBROUTINE

1 SUBROUTINE CHGLST(IELEM,IADD,ILIST,ISUB)
C
C ROUTINE TO REPORT A RECORD BEING ADDED OR DELETED FROM A LIST.
C
C INPUT PARAMETERS:
C IELEM REFERENCE TO THE RECORD BEING ADDED TO LIST
C IADD A FLAG INDICATING HOW THE RECORD IS BEING ADDED:
C (1) = STACKED, (2) = QUEUED, (3) = DELETED
C ILIST AN INDEX REFERRING TO THE LIST THAT THE RECORD IS BEING
C ADDED TO. THE INDEX IS USED TO DETERMINE THE LIST'S
C FORTRAN NAME AND ITS NUMBER OF DIMENSIONS (ASSUMED TO
C BE AT MOST 1). SEE VARIABLES VARNAM AND NSUBS.
C ISUB THE SUBSCRIPT OF THE VARIABLE DESIGNATED BY "ILIST", IF
C THERE IS ONE. (OTHERWISE EQUAL ZERO)
C
2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL),(IPFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FOBMOT
C
5 REAL*8 ADDTYP(3),VARNAM(15)
6 DIMENSION NSUBS(15)
7 DATA VARNAM//IGOALS','INEWLT','IOBSV','IMNOBS','IPATH','ITARLT',
+ 'LASMNT','LINTCE','LNODIN','LOSLIS','NOLDSN','NWSNLT',/
8 DATA NSUBS/0,0,1,1,0,0,1,1,0,0,0/
9 DATA ADDTYP/"STACKED","QUEUED","DELETED"/
C
C GET PARAMETERS OF BEFORE ADDED TO LIST.
C (CURRENT SUBROUTINE CAN'T HANDLE A NON-RECORD BEING ADDED TO LIST)
10 IF (IELEM .LT. 1000000000 .OR. IELEM .GT. 1009999999) RETURN
11 CALL FARSRF(IELEM, IDTYP,IRECNO, IDUM)
C
C DETERMINE NUMBER OF DIMENSIONS OF VARIABLE AND BRANCH ACCORDINGLY
12 IF (NSUBS(ILIST) .GE. 1) GO TO 20
C LIST IS A SCALAR. REPORT IT.
13 WRITE (6,10) DTPABR(IDTYP),IRECNO,ADDTYP(IADD),VARNAM(ILIST)

UTILITIES - CHGLST SUBROUTINE

ISN
14 10 FORMAT ("RECORD ",A3,I4,1X,A8,"ON LIST ",A7)
15 RETURN
C
C LIST IS A 1-DIMENSIONAL ARRAY. REPORT IT.
16 20 WRITE (6,25) DTPABR(IDTYP),IRECNO,ADDTYP(IADD),VARNAME(ILIST),ISUB
17 25 FORMAT ("RECORD ",A3,I4,1X,A8,"ON LIST ",A7,"(',I3,'")
18 RETURN
19 END

UTILITIES - CHGVAL SUBROUTINE

ISN

1 SUBROUTINE CHGVAR(IVAR,VALUE)
2 ENTRY CHGVR1(IVAR,VALUE,ISUB1)
3 ENTRY CHGVR2(IVAR,VALUE,ISUB1,ISUB2)
4 ENTRY CHGVR3(IVAR,VALUE,ISUB1,ISUB2,ISUB3)

C
C ROUTINE TO REPORT THE CHANGE IN VALUE OF A VARIABLE.

C
C INPUT PARAMETERS:
C IVAR A UNIQUE INDEX IDENTIFYING THE VARIABLE WHOSE VALUE IS
C TO BE REPORTED BY CHGVAR. THIS INDEX IS USED TO LOOK UP
C THE PROPERTIES OF THE VARIABLE: ITS NAME, TYPE, AND
C NUMBER OF DIMENSIONS. SEE ARRAYS VARNAM, ITYPE, & NSUBS.
C VALUE THE VARIABLE'S NEW VALUE TO BE REPORTED. IF THE VARIABLE
C IS OF TYPE INTEGER, THIS VALUE SHOULD BE PASSED AS AN
C INTEGER TO THE ROUTINE, EVEN THOUGH THE DUMMY VARIABLE
C "VALUE" IS REAL.
C ISUB1 | THE FIRST, SECND AND THIRD SUBSCRIPTS OF THE
C ISUB2 | VARIABLE, IF IT HAS THESE. A VALUE OF A SUBSCRIPT
C ISUB3 | GIVEN FOR A VARIABLE NOT HAVING SUCH A SUBSCRIPT
C | WILL BE IGNORED.

C
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT

C
8 EQUIVALENCE (IVALUE,FVAIUE)
9 DIMENSION ITYPE(20),NSUBS(20)
10 REAL*8 VARNAM(20)
11 DATA VARNAM/'TMIN','BMSTAT','IPLAYER','ISIDE','FORCES','TMRESP',
+ 'ISECPL','IGOALS','IOBSV','IMNOBS','LISMNT','LINTCP',
+ 'LNODIN','NOIDSK','NEWSNLT'/
12 DATA ITYPE/2,1,1,1,2,2,1,3,3,3,3,3,3,3,3/
13 DATA NSUBS/0,0,0,0,2,0,1,0,1,1,0,1,0,0/

C
C DETERMINE THE NUMBER OF DIMENSIONS THE VARIABLE HAS AND BRANCH

UTILITIES - CHGVAR SUBROUTINE

ISN

```

C ACCORDINGLY TO SECTION OF CODE TO HANDLE IT.
14      NS = NSUBS(IVAR) + 1
15      GO TO (10,30,50,70),NS
C
C VARIABLE IS A SCALAR. REPORT ITS VALUE.
16      10 IF (ITYPE(IVAR) .EQ. 2) GO TO 20
C INTEGER VALUE.
17      FVALUE = VALUE
18      IF (ITYPE(IVAR) .EQ. 3 .AND. VALUE .NE. FNULL) GO TO 27
19      WRITE (6,15) VARNAM(IVAR),IVALUE
20      15 FORMAT (A6,10X,' =',I7)
21      RETURN
C
C REAL VALUE
22      20 WRITE (6,25) VARNAM(IVAR),VALUE
23      25 FORMAT (A6,10X,' =',F10.2)
24      RETURN
C
C RECORD REFERENCE VALUE
25      27 CALL PARSRF(IVALUE, IDTYP,IRECNO, IDUM)
26      WRITE (6,28) VARNAM(IVAR),DTPABR(IDTYP),IRECNO
27      28 FORMAT (A6,10X,' =',A3,I4)
28      RETURN
C
C VARIABLE HAS ONE DIMENSION. REPORT ITS VALUE.
29      30 IF (ITYPE(IVAR) .EQ. 2) GO TO 40
C INTEGER VALUE
30      FVALUE = VALUE
31      IF (ITYPE(IVAR) .EQ. 3 .AND. VALUE .NE. FNULL) GO TO 47
32      WRITE (6,35) VARNAM(IVAR),ISUB1,IVALUE
33      35 FORMAT (A6,'(,I2,)',6X,' =',I7)
34      RETURN
C
C REAL VALUE
35      40 WRITE (6,45) VARNAM(IVAR),ISUB1,VALUE
36      45 FORMAT (A6,'(,I2,)',6X,' =',F10.2)

```

UTILITIES - CHGVAR SUBROUTINE

1SN

37 RETURN

C

C RECORD REFERENCE VALUE

38 47 CALL PARSR' (TVALUE, IDTYP,IRECNO, IDUM)

39 WRITE (6, 6) VARNAM(IVAR),ISUB1,DTPABR(IDTYP),IRECNO

40 48 FORMAT (A6,'(*,I2,*)',6X,' =',A3,I4)

41 RETURN

C

C VARIABLE HAS TWO DIMENSIONS. REPORT ITS VALUE.

42 50 IF (ITYPE(IVAR) .EQ. 2) GO TO 60

C INTEGER VALUE

43 FVALUE = VALUE

44 IF (ITYPE(IVAR) .EQ. 3 .AND. VALUE .NE. FNULL) GO TO 67

45 WRITE (6,55) VARNAM(IVAR),ISUB1,ISUB2,IVALUE

46 55 FORMAT (A6,'(*,I2,*,*,I2,*)',3X,' =',I7)

47 RETURN

C

C REAL VALUE

48 60 WRITE (6,65) VARNAM(IVAR),ISUB1,ISUB2,VALUE

49 65 FORMAT (A6,'(*,I2,*,*,I2,*)',3X,' =',F10.2)

50 RETURN

C

C RECORD REFERENCE VALUE.

51 67 CALL PARSRP(IVALUE, IDTYP,IRECNO, IDUM)

52 WRITE (6,68) VARNAM(IVAR),ISUB1,ISUB2,DTPABR(IDTYP),IRECNO

53 68 FORMAT (A6,'(*,I2,*,*,I2,*)',3X,' =',A3,I4)

54 RETURN

C

C VARIABLE HAS THREE DIMENSIONS. REPORT ITS VALUE

55 70 IF (ITYPE(IVAR) .EQ. 2) GO TO 80

C INTEGER VALUE

56 FVALUE = VALUE

57 IF (ITYPE(IVAR) .EQ. 3 .AND. VALUE .NE. FNULL) GO TO 87

58 WRITE (6,75) VARNAM(IVAR),ISUB1,ISUB2,ISUB3,IVALUE

59 75 FORMAT (A6,'(*,I2,*,*,I2,*,*,I2,*) =',I7)

60 RETURN

UTILITIES - CHGVAR SUBROUTINE

ISN

```
C
C REAL VALUE
61 80      WRITE (6,85) VARNAM(IVAR),ISUB1,ISUB2,ISUB3,VALUE
62 85      FORMAT (A6,*(*,I2,*,*,I2,*,*,I2,*) =*,F10.2)
63      RETURN
C
C RECORD BEFERENCE VALUE
64 87      CALL PARSHP(IVALUE, IDTYP,IRECNO, IDUM)
65      WRITE (6,88) VARNAM(IVAR),ISUB1,ISUB2,ISUB3,DTPABR(IDTYP),IRECNO
66 88      FCERMAT (A6,*(*,I2,*,*,I2,*,*,I2,*) =*,A3,I4)
67      RETURN
68      END
```

UTILITIES - COLECT SUBROUTINE

ISN

1

SUBROUTINE COLECT

C ROUTINE TO PERFORM GARBAGE COLLECTION OF PLEX RECORDS.
C INSTEAD OF A PROGRAMMER'S EXPLICITLY CALLING A DELETE ROUTINE
C WHENEVER A RECORD IS NO LONGER NEEDED, OR INSTEAD OF HIS CALLING
C A SPECIAL BOOKKEEPING ROUTINE WHENEVER HE ADDS, DELETES, OR
C CHANGES A REFERENCE TO A BECORD OR LIST, IT SEEMS EASIEST AND
C SAFEST FOR THE PROGRAMMER TO HAVE PROVIDED AN EXPLICIT GARBAGE
C COLLECTION ROUTINE. THEN WHENEVER A PROGRAMMER WISHES TO DELETE
C A RECORD OR LIST, HE NEED ONLY SET A REFERENCE TO THAT RECORD OR
C LIST TO SOME OTHER VALUE (E.G., A NEW RECORD OR LIST OR TO NULL).
C IF THE REFERENCE BEING MODIFIED WAS THE LAST ONE TO THIS RECORD
C OR LIST, THE RECORD OR LIST WILL BECOME GARBAGE - PHYSICALLY
C ALLOCATED SPACE BUT INACCESSIBLE. IT WILL REMAIN GARBAGE UNTIL THIS
C ROUTINE, THE GARBAGE COLLECTOR, IS CALLED TO FIND AND RECLAIM
C THESE RECORDS.

C WHEN THE GARBAGE COLLECTOR IS CALLED IT INITIALIZES ALL RECORDS
C IN A COLLECTABLE TYPE FILE TO "COLLECTABLE". IT THEN ITERATES
C THROUGH EXTERNAL REFERENCES TO PLEX RECORDS, PLACING THEM ON A STACK.
C THE EXTERNAL REFERENCES WILL BE CONTAINED IN (1) THE SCALAR PORTION
C OF /STATEV/, (2) THE NON-COLLECTABLE FILES OF /STATEV/, OR (3) A
C SPECIAL COMMON BLOCK /RECBFF/. AFTER STACKING ALL EXTERNAL
C REFERENCES TO RECORDS, THE ROUTINE TAKES THE FIRST ONE OFF THE
C STACK, MARKS THE CORRESPONDING RECORD ACTIVE, AND ADDS ANY STACK
C REFERENCES TO OTHER RECORDS CONTAINED IN THIS RECORD TO THE TOP
C OF THE STACK. PROCESSING THEN CONTINUES WITH THE NEXT ITEM ON
C THE STACK IN THE SAME MANNER. WHEN THE STACK HAS BEEN COMPLETELY
C PROCESSED AND NO RECORDS REMAIN, THE RECORD WHICH ARE STILL MARKED
C "COLLECTABLE" ARE RECLAIMED.

C THE GARBAGE COLLECTOR IS CALLED BY NEWREC WHEN IT TRIES TO
C ALLOCATE RECORDS AND NONE ARE AVAILABLE OF THE GIVEN TYPE. SINCE A
C CALL TO THE GARBAGE COLLECTOR CAN OCCUR IN MANY DIFFERENT CONTEXTS
C IN THE PROGRAM, A PROGRAMMER MUST MAKE SURE THAT ANY LIST OR RECORD
C THAT HE DOES NOT WISH RECLAIMED IS ALWAYS REFERENCED, EITHER
C DIRECTLY OR INDIRECTLY, BY AN EXTERNAL RECORD REFERENCE ACCESSIBLE

UTILITIES - COLECT SUBROUTINE

ISN

```

C BY THE GARBAGE COLLECTOR - EITHER IN /STATEV/ OR IN /BECREP/.
C IF INSUFFICIENT SPACE IS INITIALLY SET ASIDE IN ANY DYNAMICALLY
C ALLOCATED FILE FOR RECORDS, THE GARBAGE COLLECTOR WILL GET CALLED
C MORE AND MORE FREQUENTLY AS STORAGE NEARS EXHAUSTION. AT SUCH A
C POINT IT MIGHT BE BETTER TO STOP AND START OVER AGAIN WITH MORE
C ROOM ALLOWED. EVENTUALLY IT MAY BE REASONABLE TO PUT IN AN
C AUTOMATIC CHECK FOR THIS SITUATION. THIS VERSION OF THE ROUTINE,
C HOWEVER, WILL JUST PRINT OUT A STATUS REPORT EVERY TIME THE
C GARBAGE COLLECTOR IS CALLED, INDICATING TO THE USER WHEN IT IS
C CALLED, HOW MANY RECORDS OF EACH TYPE IT COLLECTS, AND HOW MANY OF
C EACH TYPE HAVE NEVER BEEN ALLOCATED. IT IS UP TO THE USER TO USE
C THIS INFORMATION TO MODIFY HOW MANY RECORDS OF A GIVEN TYPE TO
C ALLOW ON SUBSEQUENT RUNS, IF NEEDED.
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IPAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FORECT
8      COMMON /PARS3/
9      COMMON /RECREP/
10     DIMENSION RECREP(140)
11     EQUIVALENCE (RECREP(1),IGCALS)
12     COMMON /GARCOL/
13     LOGICAL*1 ACTIVE
C
C     INITIALIZE RECORDS TO "COLLECTABLE"
C
14     DO 10 IR=1,NCOICT
15   10     ACTIVE(IR) = .FALSE.
16     IQ = 0
C
C     ... INITIALIZE IAVAIL TO NULL
C
17     CALL NJLIPY(IAVAIL,NDTYPE)

```

UTILITIES - COLECT SUBROUTINE

ISN

```
C
C ADD EXTERNAL REFERENCES TO RECORDS AND LISTS TO STACK; MARK RECORDS
C ACTIVE
18    CALL QXREFS(RECRFQ,1,NBFCRF,4)
19    CALL QXREFS(ITEM,1,NFIDS(1),4)
20    DO 20 IDTYP=LSITE,LACTD
21        IOFF = IDTCFF(IDTYP) + 1
22        IOFFG = IDTOFF(IDTYP) + NPLDS(IDTYP)*(NRECS(IDTYP) - 1) + 1
23        CALL QXREFS(ITEM(ICFF),NPLDS(IDTYP),NRECS0(IDTYP),ITEM(IOFFG))
24    20    CONTINUE
C
C GET NEXT ITEM OFF STACK
C
25    30    IQ = IQ - 1
26    IF (IQ .LT. 0) GO TO 40
27    IRECRF = IRECQ(IQ+1)
C
C ADD ANY REFERENCES TO RECORDS AND LISTS NOT ALREADY ON THE STACK TO
C THE STACK; MARK THE RECCBD AS ACTIVE
C
28    CALL PABSRF(IRECRF,IDTYP,IRECNO,IDLJN)
29    IF(IDTYP.EQ.LLIST) GO TO 35
30    IOFF = IREF1(IRECBF)+1
31    IOFFG = IDTOFF(IDTYP) + NPLDS(IDTYP)*(NRECS(IDTYP) - 1) + 1
32    CALL QXREFS(ITEM(IOFF),NPLDS(IDTYP),1,ITEM(IOFFG))
33    GO TO 30
C
C IRECRF IS A LIST
C
34    35    MREF=IFIRST(IRECRF,LPTR)
35    36    IF(MREF.EQ.NULL) GO TO 30
36    IF(LPTR.EQ.NULL) GO TO 30
37    CALL QXREFS(LLISTND(2,LPTR),1,1,3)
38    IB=IRCOFF(LLIST)+LPTR
39    ACTIVE(IR) = .TRUE.
40    MREF=NEXT(IRECBF,L PTR)
```

UTILITIES - COLECT SUBROUTINE

1SN
41 GO TO 36

C
C ITERATE THROUGH FILE TYPES, SKIPPING OVER THOSE KNOWN TO HAVE NO
C COLLECTABLE RECORDS.

C
42 40 CALL INITVL(ICRECS,NDTYPE,0)
43 DO 60 IDTYP=LLIST,NDTYEE
44 IF (IDTYP .GE. LSITE .AND. IDTYP .LE. LWIND) GO TO 60

C
C DETERMINE THE LOWER AND UPPER INDEXES OF THE ACTIVE VECTOR IN
C WHICH TO LOOK FOR INACTIVE RECORDS.

45 IRO = IRCOFF(IDTYP)
46 NGAP = IRCOFF(LWEAP) - IRCCFF(LSITE)
47 IF (IDTYP .GE. LWEAP) IRO = IRO - NGAP
48 IRNN = IRO + LSTREC(IDTYP)
49 IR1 = IRO + 1
50 IF (IDTYP .GE. LWEAP .AND. IDTYP .LE. LACTD) IR1 = IRO +
+ NRECS0(IDTYP) + 1
51 IF (IR1 .GT. IRNN) GC TO 60

C
C ITERATE OVER THE POTENTIALLY INACTIVE RECORDS OF THE CURRENT FILE,
C COLLECTING THOSE WHICH ARE GARBAGE.

52 DO 50 IR=IR1,IRNN
53 IF (ACTIVE(IR)) GO TO 50
54 IRECRF = NEWREF(IDTYP,IR-IRO,0)
55 CALL DELREC(IRECRF)
56 ICRECS(IDTYP) = ICRECS(IDTYP) + 1
57 50 CONTINUE
58 60 CONTINUE

C
C OUTPUT REPORT OF GARBAGE COLLECTION

C
59 DO 70 I=1,NDTYEE
60 70 IVRECS(I) = NRECS(I) - LSTREC(I) - NGENRC
61 I1 = 1
62 IN = 18

UTILITIES - COLECT SUBROUTINE

1SN

63 WRITE (7,100) TMIN

64 100 FORMAT ('-GARBAGE COLLECTOR CALLED AT TIME ',F8.2//
* 40X,'COLLECTION RESULTS:')

65 80 WRITE(7,105) (DTPABB(I),I=I1,IN)

66 WRITE(7,106) (ICRECS(I),I=I1,IN)

67 WRITE(7,107) (BRECS(I),I=I1,IN)

68 WRITE(7,108) (IVRECS(I), I=I1,IN)

69 105 FORMAT ("RECORD TYPE: ",18(3X,A2))

70 106 FORMAT ("NUMBER COLLECTED: ",18I5)

71 107 FORMAT ("TOTAL NUMBER RECORDS: ",18I5)

72 108 FORMAT ("NUMBER NEVER ALLOCATED: ",18I5,//)

C

73 IF (IN .GE. NDTYPE) RETURN

74 I1 = I1 + 18

75 IN = IN + 18

76 IF (IN .GT. NDTYPE) IB = NDTYPE

77 GO TO 80

78 END

UTILITIES - CONDAC SUBROUTINE

ISN

1

SUBROUTINE CONDAC(IPLARF)

C
C ROUTINE INTERPRETS CONDITIONAL ACTIONS ASSOCIATED WITH PERSON
C IPLARF. A CONDITIONAL ACTION HAS THE FORM: WHEN(BOOLEAN EXPRESSION).
C BOTH THE WHEN(,) AND THE BOOLEAN EXPRESSION ARE REPRESENTED BY
C RELATION RECORDS. THE "ACT" PART IS REPRESENTED BY EITHER AN
C ACTION OR A FUNCTION RECORD.
C THIS ROUTINE SEQUENTIALLY TESTS ALL CONDITIONAL ACTIONS
C ASSOCIATED WITH THE DESIGNATED PERSON. IF IT ENCOUNTERS A CONDITION
C WHICH IS SATISFIED, IT ACTIVATES THE ASSOCIATED ACT. "ACTIVATE"
C MEANS EITHER (1) STACKING THE ACT ON THE PLAYER'S ACTIVITY LIST, IF
C THE ACT IS AN ACTION RECORD, OR (2) EVALUATING THE ACT, IF IT IS
C A FUNCTION RECORD. EVALUATION OF THE FUNCTION CAN RESULT IN AN
C ACTION RECORD, IN WHICH CASE ALTERNATIVE (1) IS FOLLOWED FOR THE
C RESULT.
C THIS ROUTINE RETURNS WHEN ALL OF THE PLAYER'S CONDITIONAL ACTIONS
C HAVE BEEN PROCESSED. THERE MAY ALSO BE CONDITIONAL ACTIONS PERTINENT
C TO A PLAYER STORED WITH THE RECORD DESCRIBING THE FORCE HE BELONGS
C TO.
C
C INPUT PARAMETERS:
C IPLARF RECORD REFERENCE TO THE PERSON BEING PROCESSED.
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900), DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 CCMNCN /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FIDNAM,FOBJECT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 COMMON /NEW/
C
11 IFLAG = 0
12 IACTIV = IREF(JACTIV,IPLARF)

UTILITIES - CONDAC SUBROUTINE

1 ISN

13 ICONDS = IVAL(JSOFS,IFLABF)

14 8 ICONE = IPIRST(ICCND\$,\$C)

15 10 IF (ICOND .EQ. NULL) GO TO 50

16 C TEST WHETHER CONDITIONAL ACTION IS SATISFIED.

17 IACT = NRTEST(ICOND)

18 C IF NOT, GO TRY NEXT CONDITIONAL ACTION

19 15 IF (IACT .EQ. 0) GO TO 40

20 C IF SO, IS RESULT AN ACTION RECORD ?

21 CALL PARSRP(IACT, IDTYP,IRECNO, IDUM)

22 IF (IDTYP .NE. IACTN) GO TO 20

23 ITEM(IACTIV) = ISTACK(IACT,ITEM(IACTIV))

24 GO TO 40

25 C IF RESULT IS A RELATION RECORD, TEST ITS TRUTH

26 20 IF (IDTYP .NE. IREIN) GO TO 30

27 IACT = NRTEST(IACT)

28 GO TO 15

29 C IF RESULT IS A FUNCTION RECORD, EVALUATE IT.

30 30 IF (IDTYP .NE. IFUNC) CALL ERR(39,35, IDTYP,0,0)

31 IACT = IFNCVL(IACT)

32 GO TO 15

33 C GET NEXT CONDITIONAL ACTION

34 40 ICOND = NEXT(ICCND\$,KC)

35 GO TO 10

36 C

37 C HAVE ANY CONDITIONAL ACTIONS ASSOCIATED WITH THE FORCE THAT THE

38 C BELONGS TO BEEN PROCESSED ?

39 50 IF (IFLAG .NE. 0) RETURN

40 IFLAG = 1

41 IFORCE = IVAL(JFOBCE,IPLRPF)

42 IF (IFORCE .EQ. NULL) RETURN

43 CALL FARSRP(IFOBCE, IDTYP, IDUM, IDUM)

44 IF (IDTYP .NE. LFORCE) RETURN

45 ICONDS = IVAL(JFSOPS,IFCFC)

46 GO TO 8

47 END

UTILITIES - COORDS SUBROUTINE

15N

1 SUBROUTINE COORDS(LOC,X,Y,Z)

C ROUTINE TO CALCULATE THE X, Y, AND Z COORDINATES CORRESPONDING TO
C A LOCATION SPECIFIED IN TERMS OF A SOURCE, A SINK, AND A FRACTIONAL
C DISTANCE BETWEEN.

C INPUT PARAMETERS:
C LOC A REFERENCE TO A LOCATION RECORD

C OUTPUT PARAMETERS:
C X X COORDINATE OF LOCATION
C Y Y COORDINATE OF LOCATION
C Z Z COORDINATE OF LOCATION

2 COMMON /PARS1/

C

3 ISOURC = IVAL(JSOURC,LOC)
4 ISINK = IVAL(JSINK,LOC)
5 FRAC = VAL(JFRAC,LOC)
6 XSOURC = VAL(JXCO,ISOURC)
7 YSOURC = VAL(JYCO,ISOURC)
8 ZSOURC = VAL(JZCO,ISOURC)
9 X = XSOURC + FRAC*(VAL(JXCO,ISINK) - XSOURC)
10 Y = YSOURC + FRAC*(VAL(JYCO,ISINK) - YSOURC)
11 Z = ZSOURC + FRAC*(VAL(JZCO,ISINK) - ZSOURC)
12 RETURN
13 END

UTILITIES - COPY SUBROUTINE

ISN

1

SUBROUTINE COPY(X,Y,N)

C

C ROUTINE TO COPY VECTOR X TO VECTOR Y.

C

C INPUT PARAMETERS:

C X VECTOR TO BE COPIED

C Y VECTOR TO RECEIVE CONTENTS OF X

C N SIZE OF VECTORS X AND Y

C

2

DIMENSION X(N),Y(N)

C

3

DO 10 I=1,N

4

Y(I) = X(I)

5

RETURN

6

END

UTILITIES - DELIST SUBROUTINE

ISN

1

SUBROUTINE DELIST(IELEM,IFIELD,IREFCRF)

C ROUTINE TO DELETE AN ELEMENT FROM A LIST. IF THERE IS A
C PRIOR ELEMENT ON THE LIST, ITS NEXT FIELD IS SET TO THE SAME
C VALUE AS THE NEXT FIELD OF THE ITEM DELETED. IF THERE IS NO
C PRIOR ITEM ON THE LIST, THE HEAD OF THE LIST IS INSTEAD SET TO
C THE VALUE IN THIS FIELD. IF THE ELEMENT DESIGNATED TO BE DELETED
C CAN'T BE FOUND, AN ERROR COMMENT IS GENERATED.

C

C INPUT PARAMETERS:

C IELEM INDEX OF LIST NODE REFERRING TO THE ELEMENT TO BE
C DELETED FROM THE LIST, OR NULL IF THE ELEMENT IS ON
C A ONE-ELEMENT LIST NOT ENCASED IN A LISTND.
C (IFLEM IS THE VALUE RETURNED AS THE SECOND PARAMETER
C OF THE FUNCTIONS IFIRST OR NEXT).

C IFIELD THE PARAMETERS IFIELD AND IREFCRF TOGETHER IDENTIFY THE
C & STORAGE LOCATION AT WHICH A REFERENCE TO THE HEAD OF THE
C IREFCRF LIST IS STORED. THERE ARE TWO OPTIONS:

C (1) IF IFIELD > 0, IREFCRF IS A REFERENCE TO THE RECORD IN
C WHICH IT IS STORED; IFIELD IS AN INDEX OF THE FIELD OF
C THIS RECORD IN WHICH THE LIST REFERENCE IS STORED.
C (2) IF IFIELD <= 0, THEN IREFCRF IS ITSELF A VARIABLE
C REFERRING TO THE HEAD OF THE LIST. IT MAY THEN BE
C UPDATED.

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900), DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULI),(IPAIL,FAIL)
7 REAL*8 DTPNAM,FIDNAM,FCFMCT
8 COMMON /PARS1/
9 COMMON /PARS3/

C

C IF THIS IS A ONE ELEMENT LIST, TREAT SPECIALLY

UTILITIES - DELIST SUBROUTINE

ISM

```
10      IF (IELEM .NE. NULL) GO TO 3
11      2      I = IREF(IPFIELD,IRECRF)
12      IF (IFIELD .GT. 0) ITEM(I) = NULL
13      IF (IFIELD .LE. 0) IRECRF = NULL
14      RETURN
15      3      ILLIST = IVAL(IFIELD,IRECRF)
16      IF (IFIELD .LE. 0) ILIST = IRECRF
17      IF (ILIST .NE. NULL) GO TO 10
C     ELEMENT COULDN'T BE LOCATED ON LIST
18      5      CALL ERR(22,16,IELEM,IFIELD,IRECRF)
19      RETURN
20      10     IDUM = IPIRST(ILIST,ILSTND)
21      IF (ILSTND .NE. IELEM) GO TO 20
C     FIRST ELEMENT ON LIST IS TO BE DELETED
22      IF (IFIELD .GT. 0) IR = IREF(IPFIELD,IRECRF)
23      IF (IFIELD .LE. 0) IR = IREF1(IRECRF)
24      IF (ILSTND .NE. NULL) GO TO 15
25      ITEM(IR) = NULL
26      RETURN
27      15     IDUM = NEXT(ILIST,ILSTND)
28      IF (IDUM .EQ. NULL) GO TO 2
29      ITEM(IR) = NEWREF(LLIST,ILSTND,0)
30      RETURN
C     SAVE INDEX OF LIST NODE
31      20     IPREV = ILSTND
C     GET NEXT LIST NODE
32      IDUM = NEXT(ILIST,ILSTND)
33      IF (ILSTND .EQ. NULL) GO TO 5
34      IF (ILSTND .NE. IELEM) GO TO 20
C     HAVE FOUND LIST NODE OF INTEREST; MAKE PRIOR LIST NODE POINT
C     TO ITS SUCCESSOR.
35      LISTND(JNXT,IPREV) = LISTND(JNXT,ILSTND)
36      RETURN
37      END
```

UTILITIES - DELREC SUBROUTINE

ISN

```
1      SUBROUTINE DELREQ(IRECBF)
C
C      ROUTINE TO RECLAIM A RECCRD WHICH IS NO LONGER NEEDED AND
C      RETURN IT TO AVAILABLE STORAGE FOR LATER REUSE.  ROUTINE NULLS
C      OUT THE FIELDS OF THE RECCRD.
C
C      INPUT PARAMETERS:
C      IRECBF    A RECORD REFERENCE IDENTIFYING THE RECORD TO BE DELETED.
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (NULL,NOLL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FIDNAM,FCRECT
C
C      BREAK THE RECORD REFERENCE INTO ITS COMPONENTS
8      CALL PARSBF(IRECRF,IDLTYPE,IRECNO,IDLUM)
C      NULL OUT THE CONTENTS OF THE RECORD
9      IRECPT = IREF(1,IRECRF)
10     CALL NULIFY(ITEM(IRECPT),NPLDS(IDLTYPE))
C      PUSH THE EMPTY RECCRD ONTO THE AVAILABLE SPACE LIST.
C      (IRECPT IDENTIFIES THE JNEXT POSITION OF THE RECORD TO BE RECLAIMED)-
11     ITEM(IRECPT) = IAVAIL(IDLTYPE)
12     IAVAIL(IDLTYPE) = IBECKNO
13     RETURN
14     END
```

UTILITIES - DIREC SUBROUTINE

ISN

1

SUBROUTINE DIREC(I,J,THETA,PHI)

C
C ROUTINE TO CALCULATE THE ANGULAR ORIENTATION OF A RAY EXTENDING
C FROM ITEM I TO ITEM J. THE ANGULAR ORIENTATION IS MEASURED IN
C RADIANS, USING THE STANDARD POLAR COORDINATE SYSTEM NOTATION,
C VIEWING I AS THE CRIGIN. THE ARGUMENTS I AND J MAY BE EITHER
C SITE NODES OR PHYSICAL OBJECTS WHICH HAVE A LOCATION FIELD (VEHICLES,
C PERSONS, SENSORS, OR EFFECTORS (ACTIVATED DELAYS)).
C

C INPUT PARAMETERS:

C I A REFERENCE TO A SITE NODE OR A PHYSICAL OBJECT (WITH
C LOCATION FIELD).
C J A REFERENCE TO A SITE NODE OR A PHYSICAL OBJECT (WITH
C LOCATION FIELD).

C OUTPUT PARAMETERS:

C THETA THE ANGLE DESCRIBING THE ORIENTATION OF THE RAY I-J
C IN THE X-Y PLANE, IN RADIANS. (0. <= THETA <= 2*PI)
C PHI THE ANGLE DESCRIBING THE ORIENTATION OF THE RAY I-J
C IN THE PLANE VERTICAL TO THE X-Y PLANE. MEASURED IN
C RADIANS. -PI/2. <= PHI <= PI/2..
C

2

COMMON /PARS/
EQUIVALENCE (FNULL,NULL), (IPAIL,FAIL)
REAL*8 DTPNAM,FIDNAM,FCRFCT
COMMON /PARS1/

C

6 DATA PI/3.14159/,PI2/1.570795/

C

C CHECK WHETHER I IS A VALID ARGUMENT AND IF SO GET ITS COORDINATES.
7 CALL PALSBF(I, IDTYP1, IDUP, IDUM)
8 IF (IDTPOK (IDTYP1) .EQ. 2) GO TO 10
9 IF (IDTPOK (IDTYP1) .NE. 0) CALL ERR(17,13,I, IDTYP1,0)
C ARGUMENT I IS A SITE NODE
10 X1 = VAL(JXCC,I)

UTILITIES - DIREC SUBROUTINE

ISN

```
11      Y1 = VAL(JYCO,I)
12      Z1 = VAL(JZCC,I)
13      GO TO 20
C   ARGUMENT I IS A PHYSICAL OBJECT WITH A LOCATION FIELD
14      10 CALL COORDS(IVAL(JLOCN,I),X1,Y1,Z1)
C
C   CHECK WHETHER J IS A VALID ARGUMENT AND IF SO GET ITS COORDINATES.
15      20 CALL FARSRP(J, IDTYPJ, IDUM, IDUM)
16      IF (IDTPOK(IDTYPJ) .EQ. 2) GO TO 30
17      IF (IDTPOK(IDTYPJ) .LE. 0) CALL ERR(17,13,J, IDTYPJ,0)
C   ARGUMENT J IS A SITE NODE
18      X2 = VAL(JJKCO,J)
19      Y2 = VAL(JYCO,J)
20      Z2 = VAL(JZCC,J)
21      GO TO 40
C   ARGUMENT J IS A PHYSICAL OBJECT WITH LOCATION FIELD
22      30 CALL COORDS(IVAL(JLOCN,J),X2,Y2,Z2)
C
C   DETERMINE THETA
23      40 DX = X2 - X1
24      DY = Y2 - Y1
25      DZ = Z2 - Z1
26      IF (ABS(DX) .GT. 1.E-6) GO TO 45
27      THETA = PI2
28      IF (DY .LT. 0.) THETA = PI + PI2
29      GO TO 50
30      45 THETA = ATAN(DY/DX)
31      IF (DX .LT. 0.) THETA = PI + THETA
32      IF (DX .GT. 0. AND. DY .LT. 0.) THETA = 2.*PI + THETA
C
C   DETERMINE PHI
33      50 DB = SQRT(DX**2 + DY**2)
34      IF (DB .GT. 1.E-6) GO TO 55
35      PHI = PI2
36      IF (DZ .LT. 0.) PHI = -PI2
37      RETURN
```

UTILITIES - DIREC SUBROUTINE

ISN
38 55 PHI = ATAN(DZ/DB)
39 RETURN
40 END

UTILITIES - DIREC2 SUBROUTINE

ISN

1

SUBROUTINE DIREC2(X1,Y1,Z1,X2,Y2,Z2,THETA,PHI)

C
C ROUTINE TO CALCULATE THE ANGULAR ORIENTATION OF A RAY EXTENDING
C FROM POINT (X1,Y1,Z1) TO POINT (X2,Y2,Z2). THE ANGULAR ORIENTATION
C IS MEASURED IN RADIANS, USING THE STANDARD POLAR COORDINATE SYSTEM
C NOTATION, VIEWING (X1,Y1,Z1) AS THE ORIGIN.

C

C INPUT PARAMETERS:

C X1 THE X COORDINATE OF THE ORIGIN
C Y1 THE Y COORDINATE OF THE ORIGIN
C Z1 THE Z COORDINATE OF THE ORIGIN
C X2 THE X COORDINATE OF THE 2ND POINT
C Y2 THE Y COORDINATE OF THE 2ND POINT
C Z2 THE Z COORDINATE OF THE 2ND POINT

C

C OUTPUT PARAMETERS:

C THETA THE ANGLE DESCRIBING THE ORIENTATION OF THE RAY
C IN THE X-Y PLANE, IN RADIANS. (0. <= THETA <= 2*PI)
C PHI THE ANGLE DESCRIBING THE ORIENTATION OF THE RAY
C IN THE PLANE VERTICAL TO THE X-Y PLANE. MEASURED IN
C RADIANS. -PI/2. <= PHI <= PI/2..

C

C

2 DATA TWOPI/6.28318/

C

C

C DETERMINE THETA

3 40 DX = X2 - X1
4 DY = Y2 - Y1
5 DZ = Z2 - Z1
6 45 THETA = ATAN2(DY,DX)
7 IF (THETA .LT. 0.0) THETA = THETA + TWOPI

C

C DETERMINE PHI

8 50 DR = SQRT(DX**2 + DY**2)

UTILITIES - DIREC2 SUBROUTINE

ISN
9 55 PHI = ATAN2(DZ,DR)
10 RETURN
11 END

UTILITIES - DIST FUNCTION

ISN

```
1      FUNCTION DIST(I,J)
C
C      ROUTINE TO RETURN THE DISTANCE BETWEEN TWO ITEMS. EITHER ITEM
C      MAY BE A REGION OF THE SITE (I.E., NODE), A PERSON, A VEHICLE,
C      OR A SENSOR. THE FUNCTION RETURNS THE DISTANCE BETWEEN THEM.
C
C      INPUT PARAMETERS
C      I          REFERENCE TO NODE OF SITE, PERSON, SENSOR, OR VEHICLE
C      J          REFERENCE TO NODE OF SITE, PERSON, SENSOR, OR VEHICLE
C
2      CCMCN /PARS/
3      EQUIVALENCE (PNULL,NULL),(FAIL,FAIL)
4      REAL*8 DTPNAM,FLDNAM,FORMOT
5      COMMcn /PARS1/
C
C
6      CALL FABSRF(I, IDTYPi, IDUM, IDUM)
7      CALL FABSRP(J, IDTYPj, IDUM, IDUM)
8      IF (IDTPOK(IDTYPi) .LE. 0) CALL ERR(17,11,I, IDTYPi,0)
9      IF (IDTPOK(IDTYPj) .LE. 0) CALL ERR(17,11,J, IDTYPj,0)
C
C      GET COORDINATES OF I
C
10     IF (IDTPOK(IDTYEI) .EQ. 2) GO TO 10
11     XI = VAL(JXCO,I)
12     YI = VAL(JYCC,I)
13     ZI = VAL(JZCO,I)
14     GO TO 20
15 10     CALL COORDS(IVAL(JLOCN,I),XI,YI,ZI)
C
C      GET COORDINATES OF J
C
16 20     IF (IDTPOK(IDTYPj) .EQ. 2) GO TO 30
17     XJ = VAL(JXCC,J)
18     YJ = VAL(JYCC,J)
```

UTILITIES - DIST FUNCTION

ISN
19 ZJ = VAL(JZCO,J)
20 GO TO 40
21 30 CALL COORDS(IVAI(JLCCN,J),XJ,YJ,ZJ)
C
22 40 DIST = SQRT((XI-XJ)**2 + (YI-YJ)**2 + (ZI-ZJ)**2)
23 RETURN
24 END

UTILITIES - ERR SUBROUTINE

ISN

1

SUBROUTINE EBB(IERR,ISUBR,IPAR1,IPAR2,IPAR3)

C ROUTINE TO FIELD ALL ERRORS AND TO WRITE OUT ERROR MESSAGES TO
C THE USER. THE ROUTINE FIRST PRINTS OUT GLOBAL STATUS VARIABLES
C IDENTIFYING THE GENERAL CONDITIONS UNDER WHICH THE ERROR OCCURRED.
C INCLUDED IS THE NAME OF THE ROUTINE IN WHICH THE ERROR OCCURRED.
C THEN A BRANCH IS TAKEN TO A SPECIFIC SET OF STATEMENTS TO WRITE
C OUT A SPECIFIC ERROR MESSAGE AND TO TAKE ANY CORRECTIVE ACTION
C DEEMED NECESSARY.

C

C INPUT PARAMETERS:

C IERR A POSITIVE INTEGER IDENTIFYING THE ERROR CONDITION
C ISUBR A POSITIVE INTEGER IDENTIFYING THE SUBROUTINE IN WHICH
C THE ERROR OCCURRED
C IPAR1 THE FIRST PARAMETER OF THE ERROR CONDITION
C IPAR2 THE SECOND PARAMETER OF THE ERROR CONDITION
C IPAR3 THE THIRD PARAMETER OF THE ERROR CONDITION

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMOT
8 COMMON /PARS1/
9 COMMON /PARS2/
10 COMMON /PARS3/
11 C
12 REAL*8 SUBRTN(40),DTYP(2),DTABLE(2),ARRAY(10)
13 DIMENSION SIDE(2)
14 EQUIVALENCE (PAR1,MPAR1),(PAR2,MPAR2),(PAR3,MPAR3)
DATA SUERTN/ 'NEWREC','PABSRF','IREF','IVAL','NEXT','ISTACK',
+ 'IQUEUE','ICOPY','INFLEX','IVALRF','DIST','LOS','DIREC',
+ 'IPATH','CHGFLD','DELIST','QKREPS','OBS','OBSERV','SENSE',
+ 'LOSLIS','POCT','POES','LISTOB','CHKOBS','DECIDE','ACTTY',

UTILITIES - ERR SUBROUTINE

15 ISN
16 DATA SIDE/*SPE*,*AEE*/
17 DATA DTYP/*ACTION*,*RELATION*/,DTABLE/*MACCTP*,*MRELTP*/
17 DATA ARRAY/*IACTIV*/
C
C EQUATE ARGS SO THAT FLOATING POINT "PLRS" ARE DEFINED
18 MPAR1 = IPAR1
19 MPAR2 = IPAR2
20 MPAR3 = IPAR3
21 WRITE (7,5) IEBR,SUBRIN(ISUBR)
22 5 FORMAT (*- ***** EROR *,I3,* IN ROUTINE *,A8)
23 WRITE (7,6) TMIN,SIDE(ISIDE),IPLAYER
24 6 FORMAT (* TIME=*,F6.2,* SIDE=*,I4,* LAST REFERENCED PLAYER=*,I3)
C
25 GO TO {10,20,30,40,50,60,70,80,90,100,110,120,130,140,150,160,
+ 170,180,190,200,210,220,230,240,250,260,270,280,290,300,310,
+ 320,330,340,350,360,370,380,390,400,410,420
+),IERR
C
26 10 WRITE (7,15) IFAIL
27 15 FORMAT (* THE PARAMETER IDENTIFYING A RECORD DATATYPE HAS THE*/
+ * 'VALUE',I5,* AND IS THUS OUT OF RANGE. FURTHER USE OF THIS*/
+ * PARAMETER WILL BE SUPPRESSED. *)
28 RETURN
C
29 20 WRITE (7,25) IPAR1
30 25 FORMAT (* A PARAMETER WHICH IS SUPPOSED TO BE A FIELD OR RECORD*/
+ * REFERENCE HAD THE VALUE ',I5,', AND HENCE IS INCORRECT. *)
31 RETURN
C
32 30 WRITE (7,35) DTEKAM(IFAIL),IPAR2
33 35 FORMAT (* A RECORD INDEX USED TO REFER TO A RECORD OF TYPE*,A8*/
+ * HAD THE VALUE ',I5,', AND HENCE WAS OUT OF BOUNDS. *)
34 RETURN
C

UTILITIES - ERR SUBROUTINE

ISN

35 40 WRITE (7,45) IPAR2,DTENAME(IPAR1),IPAR3
36 45 FCRRMAT (" A FIELD INDEX REFERENCING A FIELD FROM RECORD NUMBER",
+ " I6/" OF DATATYPE ',A8,' AND THE VALUE ',I5,' AND HENCE'
+ " WAS CUT OF BOUNDS.")

37 RETURN

C

38 50 WRITE (7,55) IEAR1
39 55 FORMAT (" A RECORD INDEX FOR A LISTND RECORD HAD THE VALUE ',I5/
+ " AND WAS THUS CUT OF BOUNDS.")

40 RETURN

C

41 60 WRITE (7,65) IPAR1,DTENAME(IPAR2)
42 65 FORMAT (" A RECCRD REFERENCE THAT WAS SUPPOSED TO IDENTIFY A"/
+ " LISTND HAD THE FAULTY VALUE ',I5,'. IT REFERS TO A RECORD"/
+ " OF TYPE ',A8)

43 RETURN

C

44 70 WRITE (7,75) IPAR1,IPAR2
45 75 FCRRMAT (" THE CURRENT POINTER WHICH WAS SUPPOSED TO"/
+ " IDENTIFY THE CURRENT LISTND OF LIST ',I12/
+ " HAD A BAD VALUE: ',I12")

46 RETURN

C

47 80 WRITE(7,85) PAR1,PAR2
48 85 FORMAT (" INVALID DATATYPE NAME ENCOUNTERED: ',2A4/
+ " REMAINDER OF SECTION IGNORED")

49 RETURN

C

50 90 WRITE(7,95) DTENAME(IPAR1),IPAR2
51 95 FORMAT(" A RECORD INDEX USED TO REFER TO DATATYPE: ',A8/
+ " IS A DUPLICATE. THE INDEX IS: ',I3,' RECORD IGNORED")

52 RETURN

C

53 100 WRITE(7,105) PAR1,DTPNAME(IPAR2),IPAR3
54 105 FCRRMAT("WARNING: VALUE OF ',F8.2,' OUT OF RANGE FOR DATATYPE ',A8/
+ " RECORD REFERENCE IS: ',I10")

UTILITIES - ERR SUBROUTINE

ISN

55 RETURN

 C

56 110 WRITE(7,115) IPAR1,IPAR2,IPAR3

57 115 FORMAT(' BAD RECORD REFERENCE IN RECORD ',I10,' SET TO NULL'//
 + 'REFERENCE IS ',A2,I3)

58 RETURN

 C

59 120 WRITE(7,125) DTENAM(IPAR1),IPAR_ R3

60 125 FORMAT (' LIST NODES EXHAUSTED WHILE PROCESSING DATATYPE ',A8//
 + 'RECORD NUMBER ',I5,' FIELD NUMBER ',I2,'.')

61 RETURN

 C

62 130 WRITE(7,135) IPAR1,IPAR2,IPAR3

63 135 FORMAT('MULTIPLE REFERENCE TO LIST ',I3,'. REFERENCED IN RECORDS'//
 + ' 2(I10,2X),'SECCND REFERENCE IGNORED.')

64 RETURN

 C

65 140 WRITE(7,145) IPAR1,IPAR2

66 145 FORMAT(' LIST ',I3,' REFERENCED IN RECORD ',I10,' AFTER BEING ',
 + 'DEFINED.'// ' REFERENCE ALLOWED, BUT MAY BE IN ERROR')

67 RETURN

 C

68 150 WRITE(7,155) DTENAM(IPAR1),IPAR2

69 155 FORMAT('DATATYPE ',A8,' RECORD ',I3,' REFERENCED BUT NOT DEFINED')

70 RETURN

 C

71 160 WRITE(7,165) DTENAM(IPAR1),IPAR2

72 165 FORMAT(' DATATYPE ',A8,' RECORD ',I3,' SKIPPED ON INPUT')

73 RETURN

 C

74 170 WRITE (7,175) IPAR1,DTENAM(IPAR2)

75 175 FORMAT (' THE RECCRD REFERENCE ',I12,' OF DATA TYPE "',A8,'" ,'/
 + ' WAS NOT OF A PROPER TYPE AS AN ARGUMENT TO THE FJNCTION'//
 + ' DIST.')

76 RETURN

 C

UTILITIES - ERR SUBROUTINE

1SN

77 180 CALL PARSRF(IPAR2, IDTYP, IRECNO, IDUM)
78 WRITE (7,185) ARRAY(IPAR1),IPAR3,DTPABR(IDTYP),IRECNO
79 185 FORMAT (' THE AMOUNT OF STORAGE ALLOCATED FOR ARRAY ',A8/
+ ' = ',I5,', WAS OVERFLOWED WHILE PROCESSING RECORD REFERENCE',
+ A3,I4,'-')
80 RETURN
C
81 190 CALL PARSRF(IPAR1, IDCUR, IRCCUR, IDUM)
82 CALL PARSRF(IPAR2, IDSCU, IRCSOU, IDUM)
83 CALL PARSRF(IPAR3, IDSNK, IRCSNK, IDUM)
84 WRITE (7,195) DTPABB(IDSCU),IRCSOU,DTPABR(IDSNK),IRCSNK,
+ DTPABR(IDCUR),IRCCUR
85 195 FORMAT (' NO PATH COULD BE FOUND FROM ',A3,I4,' TO ',A3,I4,'-'/
+ ' THE PATH ENDED AT ',A3,I4,'-')
86 RETURN
C
87 200 IFLENM = IFDNPT(IPAR3) + IPAR1
88 WRITE (7,205) IFDBAM(IFLENM),DTPABR(IPAR3),IPAR2
89 205 FORMAT (' FIELD ',A8,' OF RECORD ',A3,I4,' IS TO BE CHANGED.'/
+ ' ITS TYPE INDICATES IN IT GENERIC RECORD SPECIFIED A "4"/'
+ ' CODE, INDICATING THAT IT COULD ASSUME VALUES OF MULTIPLE'/
+ ' DATA CLASSES. THIS IS INCONSISTENT WITH THE ASSUMPTIONS'/
+ ' MADE WHEN THE SUBCUTINE DETECTING THE "ERROR" WAS CODED.')
90 RETURN
C
91 210 WRITE (7,215) DTYP(IPAR1),IPAR2,DTABLE(IPAR1)
92 215 FORMAT (' A ',A8,' OF THE TYPE ',I12,' WAS NOT RECORDED IN TABLE'/
+ ' A8,', ' THUS NOT ALLOWING RELIABLE DETERMINATION OF THE'/
+ ' DATATYPES OF ITS PARAMETERS.')
93 RETURN
C
94 220 WRITE (7,225) IPAR1,IEAR2,IPAR3
95 225 FORMAT (' THE ELEMENT OR LIST NODE ',I12,' TO BE DELETED FROM'/
+ ' A LIST REFERENCED FROM FIELD ',I12,' OF RECORD ',I12/
+ ' COULD NOT BE FOUND.')
96 RETURN

UTILITIES - ERR SUBROUTINE

ISN

C
97 230 WRITE (7,235) IPAR1
98 235 FORMAT (" OVERFLOW OF GARBAGE COLLECTOR QUEUE AT ",I6," RECORDS.")
99 RETURN
100 240 WRITE(7,245)
101 WRITE(7,246) IPAR1,IPAR2,IPAR3
102 FORMAT(" ILLEGAL MOVE MISSION -- NULL DESTINATION")
103 FORMAT(" PAR1 = ",I4," PAR2 = ",I4," PAR3 = ",I4)
104 RETURN

C
105 250 WRITE(7,255) IPAR1
106 255 FORMAT(' ILLEGAL ACTIVITY TYPE',I6,'GIVEN IN MISSION')
107 RETURN
108 260 WRITE(7,265) IPAR1
109 265 FORMAT(' OVERFLOW IN TABLE JREFT OF POTENTIAL OBSERVATIONS.'/
+ ' THE CURRENT MAXIMUM SIZE OF THIS TABLE IS ',I3)
110 RETURN
111 270 WRITE(7,275) IPAR1
112 275 FORMAT(" PERSON",I4," HAS NO PLACE IN HIS LOCATION RECORD")
113 RETURN

C
114 280 WRITE (7,285) IEAF1,IEAF2
115 285 FORMAT (" MORE THAN ",I3," LEVELS OF EMBEDDING WERE ENCOUNTERED"/
+ " WHILE TRYING TO EVALUATE THE RELATION ",I12);
116 RETURN

C
117 290 WRITE (7,295) IPAR1,CTPNAM(IPAR2)
118 295 FORMAT ("AN ATTEMPT TO EVALUATE THE RELATION ",I12," ENCOUNTERED"/
+ " A RECORD OF DATATYPE ",A8," WHERE A RELATION RECORD WAS"
+ " EXPECTED.")
119 RETURN

C
120 300 WRITE (7,305) IPAR3,IEAR2,IPAR1
121 305 FORMAT (" IN ATTEMPTING TO EVALUATE THE RELATION EXPRESSION "/
+ " HEADED BY RECORD ",I12," ROUTINE NRTEST ENCOUNTERED IN "/
+ " RECORD ",I12," THE RELATION INDEX ",I5," WHICH IS OUT",

UTILITIES - ERR SUBROUTINE

1SN
122 + * OF BCUNDS FOR A LEGAL RELATION TYPE.")
122 RETURN
C
123 310 WRITE (7,315) IPAR1,IPAR2
124 315 FORMAT (' A CONDITIONAL ACTION TYPE OF RELATION WAS ENCOUNTERED '/
+ ' AT OTHER THAN LEVEL 1. THE HEAD RELATION WAS ',I12,'.'/
+ 'THE LEVEL AT WHICH THE CONDITIONAL ACTION OCCURRED WAS ',I2)
125 RETURN
C
126 320 WRITE(7,325) IPAR1,IPAR2,IPAR3
127 325 FORMAT('PLAYER ',I12,' CAPTURED PLAYER ',I12,
+ ' AT LOCATION ',I12)
128 RETURN
C
129 330 WRITE (7,335) IPAR1,DTEAER(IPAR2)
130 335 FORMAT (' THE RECORD REFERENCE ',I12,' WHICH WAS TO REFER TO '/
+ ' A PLANE, HAD AN IMPROPER DATA TYPE: ',A3,'.')
131 RETURN
C
132 340 WRITE (7,345) IPAR1,DTPAER(IPAR2)
133 345 FORMAT (' THE RECCRD REFERENCE ',I12,' WHICH WAS TO REFER TO '/
+ ' A REGION, HAD AN IMPROPER DATA TYPE: ',A3,'.')
134 RETURN
C
135 350 RETURN
136 360 RETURN
137 370 RETURN
138 380 WRITE (7,385) IPAR1,IPAR2
139 385 FORMAT (' THE NUMBER OF FIELD VALUES IN A RECORD OF THE '/
+ ' CURRENT TYPE IS ',I3,' WHEREAS THE NUMBER OF VALUES '/
+ ' ALLOWED FOR WAS ONLY ',I3)
140 RETURN
141 390 RETURN
142 400 RETURN
C
143 410 WRITE(7,415) IEAR1

UTILITIES - ERR SUBROUTINE

```
1SN
144      415 FORMAT(' PLAYER REFERENCE',I11,' NOT AN INSIDER')
145      RETURN
C
146      420 WRITE(7,425) IFAIL
147      425 FORMAT(' COULD NOT FIND APPROPRIATE FORCE FOR PLAYER',I11)
148      RETURN
C
149      END
```

UTILITIES - FCNLIN FUNCTION

1 ISN

1 FUNCTION FCNLIN(ARRAY,NSTEP,R,RSTEP)

C

C GIVEN A FUNCTION Y=P(X) WHICH IS TABULATED AT N EQUALLY SPACED
C POINTS, 0, RSTEP, ..., (N-1)*RSTEP, THIS ROUTINE RETURNS THE VALUE
C Y=P(R) FOR AN INPUT R. IF 0 <= R <= (N-1)*RSTEP, P(R) IS OBTAINED
C BY LINEAR INTERPOLATION. FOR R<0, P(R) = P(0). FOR R>(N-1)*RSTEP,
C P(R) = P((N-1)*RSTEP).

C

2 REAL ARRAY(NSTEP)

3 N=R/RSTEP

4 N1=N+1

5 IF(R.GT.0.0) GO TO 25

6 FCNLIN=ARRAY(1)

7 RETURN

8 25 IF(N1.LT.NSTEP) GO TO 35

9 FCNLIN=ARRAY(NSTEP)

10 RETURN

11 35 FCNLIN=ARRAY(N1)+(ARRAY(N1+1)-ARRAY(N1))*(R-FLOAT(N)*RSTEP)/RSTEP

12 RETURN

13 END

UTILITIES - FIRST FUNCTION

ISN

1

FUNCTION FIRST(IARG,ICUBNO)

C
C FUNCTION TO RETURN THE FIRST VALUE ON A LIST OF VALUES, OR IF ITS
C ARGUMENT IS NOT A LIST, TO RETURN THE ARGUMENT AS ITS VALUE INSTEAD.
C THE FUNCTION THUS RETURNS A NULL VALUE IF ITS ARGUMENT IS NULL OR
C IF IT IS A LIST CONTAINING A SINGLE NULL VALUE. IF ITS ARGUMENT
C IS A LIST, THIS FUNCTION SETS A CURRENT ITEM REFERENCE TO
C REFERENCE THE FIRST ITEM ON THE LIST; OTHERWISE IT SETS THE
C CURRENT ITEM POINTER TO NULL.

C

C INPUT PARAMETERS:

C IARG EITHER A REFERENCE TO A LIST, A REFERENCE TO SOME OTHER
C DATA ITEM, AN INTEGER, OR A NULL VALUE.

C OUTPUT PARAMETERS:

C ICUBNO CURRENT ITEM POINTER IS SET TO THE RECORD
C INDEX OF THE FIRST LISTND OF THE LIST, OR TO
C NULL IF IARG ISN'T A LIST

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7 REAL*8 DTPNAM,FIDNAM,FCFFECT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 EQUIVALENCE (IFIRST,XFIRST)

C

11 C CHECK WHETHER THE ARGUMENT IS A LIST
12 IF (IARG .LT. 1000000000) GO TO 10
13 CALL PARSHF(IARG,IITYP,IRECNO,IDUM)
14 IF (IDTYP .NE. 1LIST) GO TO 10
15 C IF DO HAVE A LIST, SET THE CURRENT ELEMENT POINTER TO DESIGNATE
16 C THE HEAD OF THE LIST.
17 ICUBNO = IRECNO

UTILITIES - FIRST FUNCTION

ISN

```
C RETRIEVE THE VALUE OF THE FIRST ITEM ON THE LIST AND RETURN
15   IFIRST = LISTND(JVAL,IRECNO)
16   FIRST = XFIRST
17   RETURN

C
C ARGUMENT IS NOT A LIST.  RETURN WITH THE ARGUMENT AS THE VALUE OF THE
C FUNCTION
18 10   IFIRST = IARG
19   FIRST = XFIRST
20   ICURNO = NULL
21   RETURN
22 END
```

UTILITIES - FNCVAL FUNCTION

ISN

1

FUNCTION FNCVAL(IFNCRF)

C

C FUNCTION TO EVALUATE A FUNCTION RECORD AND RETURN A (REAL)
C VALUE. SEE ALSO IFNCVL.

C

C INPUT PARAMETERS:

C IFNCBF A REFERENCE TO A FUNCTION RECORD TO BE EVALUATED

C

2

COMMON /PARS/

3

EQUIVALENCE (PNULL,NULL),(IFAIL,FAIL)

4

REAL*8 DTPNAM,FLDNAM,FORMAT

5

COMMON /PARS1/

6

COMMON /PARS2/

7

COMMON /PARS3/

C

8

FNCVAL = 0.0

9

RETURN

10

END

UTILITIES - FNEXT FUNCTION

ISN

1

FUNCTION FNEXT(IARG,ICURNO)

C
C FUNCTION TO RETURN THE NEXT VALUE OF THE LIST SPECIFIED BY IARG.
C THE CURRENT ITEM ECINTER, ICURNO, IDENTIFIES THE CURRENT
C POSITION ON THE LIST. IF IARG DOES NOT REFER TO A LIST OR IF THE
C LIST'S CURRENT ITEM POINTER ALREADY REFERENCES THE LAST ELEMENT OF
C THE LIST, THE FUNCTION SETS A FAILURE SIGNAL AND RETURNS A NULL
C VALUE. NOTE THAT "FNEXT" SHOULD NOT BE
C CALLED UNLESS "FIRST" WAS INITIALLY CALLED TO INITIALIZE THE
C CURRENT POINTER TO REFERENCE THE BEGINNING OF THE LIST.
C

C INPUT PARAMETERS:

C IARG A VALUE WHICH MAY BE EITHER (1) A LIST REFERENCE,
C (2) A REFERENCE TO SOME OTHER STRUCTURE, (3) AN INTEGER,
C OR (4) NULL.

C ICURNO THE RECORDED INDEX OF THE LIST NODE AT THE
C CURRENT POSITION IN THE LIST.

C

C OUTPUT PARAMETERS:

C ICURNO UPDATED TO POINT TO THE NEXT LOGICAL LISTEND OF
C THE LIST, OR TO NULL IF THERE ARE NO MORE ITEMS ON THE
C LIST. NOT CHANGED IF IARG ISN'T A LIST.

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IPAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCRMOT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 EQUIVALENCE (NEXT,XNEXT)

C

11 IRTNCD = IPAIL
12 FNEY3 = FNULL

UTILITIES - FNEXT FUNCTION

ISN

```
      C CHECK WHETHER THE ARGUMENT IS A LIST
13      IF (IARG .LT. 1000000000) RETURN
14      CALL FABSRF(IARG,IETYP,IBECNO,IDEW)
15      IF (IETYP .NE. 1LIST) RETURN
      C CHECK WHETHER THE CURRENT ECINTER HAS A LEGAL VALUE
16      IF (ICURNO .GE. 1 .AND. ICURNO .LE. NRECS(LLIST)) GO TO 5
17      CALL EBB(7,5,IARG,ICUFNC,0)
18      RETURN
      C IF DO HAVE A LIST, UPDATE THE CURRENT ELEMENT POINTER
19      5      ICURNO = LISTND(JNXT,ICUFNC)
      C IF THERE IS NO NEXT ITEM, RETURN
20      IF (ICURNO .LE. 0) RETURN
      C IF THERE IS A NEXT ITEM ON THE LIST, GET ITS VALUE
      C AND RETURN
21      NEXT = LISTND(JVAL,ICUBNC)
22      PNEXT = XNEXT
23      IETNCD = IOK
24      RETURN
25      END
```

UTILITIES - ICOPY FUNCTION

1 FUNCTION ICOPY(IRECNP)

C

C FUNCTION TO MAKE A COPY OF A PLEX RECORD AND RETURN A
C REFERENCE TO THIS COPY. THIS FUNCTION ONLY MAKES A COPY OF THE
C RECORD PASSED, NOT OF ANY OF THE RECORDS THIS RECORD MAY POINT
C TO. THUS FOR THOSE PURPOSES WHERE ONE WISHES TO COPY A WHOLE
C PLEX STRUCTURE, THE CURRENT ROUTINE MUST BE CALLED MULTIPLE
C TIMES.

C

C INPUT PARAMETERS:

C IRECRF A REFERENCE TO A PLEX RECORD

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULI,NULL),(IPAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCRMOT

C

C IS IRECRF AN APPROPRIATE REFERENCE ?
8 IF (IRECRF .GT. 1000000000) GO TO 10
9 CALL ERR(2,8,IRECRF,0,0)
10 RETURN

11 10 CALL PARSPR(IRECRF, IDTYP, IRECNO, IDGM)
C GET A NEW RECORD OF THE SAME TYPE
12 IREC = NEWREC(IDTYP)
C GET ZERO DISPLACEMENT POINTERS TO THE OLD AND NEW RECORDS
13 N = NFLES(IDTYP)
14 IOLD = IDTOFF(IDTYP) + (IRECNO-1)*N
15 INEW = IDTOFF(IDTYP) + (IREC-1)*N
C ITERATE THROUGH THE FIELDS OF THE OLD RECORD, COPYING THE CONTENTS
C TO THE NEW RECORD
16 DO 15 I=1,N
17 15 ITEM(INEW+I) = ITEM(ICLD+I)
C RETURN WITH A REFERENCE TO THE NEW RECORD

UTILITIES - ICOPY FUNCTION

ISN
18 ICOPY = NEWREF(IDTYP,IREC,0)
19 RETURN
20 END

UTILITIES - ICOUNT FUNCTION

ISN

1 FUNCTION ICOUNT(LIST,NULICT)

C FUNCTION TO COUNT THE NUMBER OF ITEMS ON A LIST AND RETURN THIS
C INTEGER. THE COUNT INCLUDES BOTH NULL AND NON-NUL ITEMS. HOWEVER,
C A SEPARATE COUNTER "NULLCT" KEEPS TRACK OF THE NULL ITEMS SO THAT
C THE CALLING ROUTINE CAN USE WHICHEVER COUNT IS APPROPRIATE. ANY
C LIST EMBEDDED IN "LIST" IS COUNTED AS A SINGLE ITEM. TO FACILITATE
C GENERAL USE OF THIS ROUTINE, THE PARAMETER "LIST" NEED NOT BE A LIST.
C IF IT ISN'T, A COUNT OF 1 IS RETURNED BY ICOUNT.

C

C INPUT PARAMETERS:

C LIST A LIST OF ITEMS (OR ANY OTHER DATA ITEM)

C

C OUTPUT PARAMETERS:

C NULLCT A NON-NEGATIVE INTEGER EQUAL TO THE NUMBER OF NULL
C ITEMS ENCOUNTERED IN "LIST".

C

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (IPFAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FC6MCT

C

5 ICOUNT = 1
6 NULLCT = 0
7 IV = IFIRST(LIST,I)
8 IF (IV .EQ. NULL) NULICT = NULLCT + 1
9 5 IV = NFXT(LIST,I)
10 IF (IERTNCD .EQ. IPFAIL) RETURN
11 ICOUNT = ICOUNT + 1
12 IF (IV .EQ. NULL) NULICT = NULLCT + 1
13 GO TO 5
14 END

UTILITIES - IFIRST FUNCTION

ISN

1

FUNCTION IPIFIRST(IARG,ICUFNO)

C

C FUNCTION TO RETURN THE FIRST VALUE ON A LIST OF VALUES, OR IF ITS
C ARGUMENT IS NOT A LIST, TO RETURN THE ARGUMENT AS ITS VALUE INSTEAD.
C THE FUNCTION THUS RETURNS A NULL VALUE IF ITS ARGUMENT IS NULL OR
C IF IT IS A LIST CONTAINING A SINGLE NULL VALUE. IF ITS ARGUMENT
C IS A LIST, THIS FUNCTION SETS A CURRENT ITEM REFERENCE TO
C REFERENCE THE FIRST ITEM ON THE LIST; OTHERWISE IT SETS THE
C CURRENT ITEM POINTER TO NULL.

C

C INPUT PARAMETERS:

C IARG EITHER A REFERENCE TO A LIST, A REFERENCE TO SOME OTHER
C DATA ITEM, AN INTEGER, OR A NULL VALUE.

C OUTPUT PARAMETERS:

C ICURNO CURRENT ITEM SCINTER IS SET TO THE RECORD
C INDEX OF THE FIRST LISTND OF THE LIST, OR TO
C NULL IF IARG ISN'T A LIST

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (PNULL,NULL),(IPFAIL,FAIL)
7 REAL*8 DTPNAM,FIDNAM,FCB#OT
8 COMMON /PARS1/
9 COMMON /PARS3/

C

10 C CHECK WHETHER THE ARGUMENT IS A LIST
11 IF (IARG .LT. 1000000000) GO TO 10
12 CALL PARSBF(IARG,IETYP,IRECNO,IDLUM)
13 C IF DO HAVE A LIST, SET THE CURRENT ELEMENT POINTER TO DESIGNATE
C THE HEAD OF THE LIST.
14 C ICURNO = IRECNO
15 C RETRIEVE THE VALUE CF THE FIRST ITEM ON THE LIST AND RETURN

UTILITIES - IFIRST FUNCTION

```
15M  
14      IFIRST = LISTND(JVAL,IRECNO)  
15      RETURN  
C  
C   ARGUMENT IS NOT A LIST.  RETURN WITH THE ARGUMENT AS THE VALUE OF THE  
C   FUNCTION  
16 10  IFIRST = IARG  
17      ICUBNO = NULL  
18      RETURN  
19      END
```

UTILITIES - IFNCVL FUNCTION

1 FUNCTION IFNCVL(IPNCBF)

C FUNCTION TO EVALUATE A FUNCTION RECORD AND RETURN AN (INTEGER)
C VALUE. SEE ALSO FNCVAL.

C INPUT PARAMETERS:

C IPNCBF A REFERENCE TO A FUNCTION RECORD TO BE EVALUATED

C

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL),(TFAIL,FAIL)
4 REAL*8 DTPNAM,FLENAM,FORMAT
5 COMMON /PARS1/
6 COMMON /PARS2/
7 COMMON /PARS3/

C

8 IFNCVL = 0
9 RETURN
10 END

UTILITIES - INITVL SUBROUTINE

ISN

1 SUBROUTINE INITVL(IVEC,N,IVAL)

C

C SUBROUTINE TO INITIALIZE ALL THE ELEMENTS OF AN INTEGER ARRAY TO

C A SPECIFIED VALUE.

C

C INPUT PARAMETERS:

C IVEC A BLOCK OF CCNTIGUOUS STORAGE TO BE INITIALIZED

C (IT NEED NOT BE A SINGLE 1-DIM ARRAY IN THE CALLING

C ROUTINE, THOUGH IT IS TREATED AS SUCH IN INITVL).

C N THE NUMBER CF COMPUTER WORDS IN IVEC TO BE INITIALIZED.

C IVAL THE VALUE TO SET EACH OF THE WORDS OF IVEC TO.

C

2 DIMENSION IVEC(N)

3 DC 10 I=1,N

4 10 IVEC(I) = IVAL

5 RETURN

6 END

UTILITIES - IPLACE FUNCTION

```
1      FUNCTION IPLACE(LOC)
C
C      FUNCTION TO DETERMINE A RECORD REFERENCE TO A PLACE (I.E., A NODE
C      VIEWED AS A REGION) WHEN GIVEN A LOCATION (I.E., A SPECIFICATION
C      IN THE FORM (SOURCE NCDE, SINK NODE, FRACTIONAL DISTANCE BETWEEN)).
C      FUNCTION RETURNS THE PLACE REFERENCE.
C
C      INPUT PARAMETERS:
C          LCC      A REFERENCE TO A LCCN RECORD.
C
2      COMMON /PARS1/
C
3      DIMENSION IPREC(15)
4      DATA IPREC/8*0,1,3*2,3,2*4/,EPS/0.5/
C
5      ISOURC = IVAL(JSOURC,LOC)
6      ISINK = IVAL(JSINK,LCC)
7      FRAC = VAL(JFRAC,LOC)
8      CALL PARSRF(ISOURC,ISINKP,IDUM,IDUM)
9      CALL PARSRF(ISINK,ISNKTP,IDUM,IDUM)
C
C      DETERMINE WHETHER TO USE THE SOURCE NODE OR THE SINK NODE TO
C      DETERMINE THE BOUNDARY BETWEEN THEM.
C
10     NODE = ISOURC
11     NCDE2 = ISINK
12     ITYPE = ISOUTP
13     IF (IPREC(ISNKTP) .LE. IPREC(ISOUTP)) GO TO 10
14     NCDE = ISINK
15     NCDE2 = ISOURC
16     ITYPE = ISNKTP
C
C      DETERMINE BOUNDARY BETWEEN NCDES BASED ON THE TYPE OF NODE.
C
17    10     IF (IPREC(ITYPE) .NE. 1) GO TO 20
```

UTILITIES - IPLACE FUNCTION

ISN

C

C IF DETERMINING NODE IS A YARD NODE, THE PLACE IS THE NODE CLOSEST
C TO THE LCC.

C

18 IPLACE = ISOURC
19 IF (FRAC .GT. 0.5) IPLACE = ISINK
20 RETURN

C

C IF DETERMINING NODE IS OF SCME OTHER TYPE THAN A YARD NODE, BASE
C DECISICN AS TO PLACE ON WHETEER THE LOCATION FALLS IN THE REGION
C OF THE DETERMINING NODE.

C

21 20 IF (IPREC(ITYPE) .NE. 2) GO TO 30 .

C

C DETERMINING NODE IS A ROOM, HALL, CR ROOF

C

22 CALL COORDS(LOC,X,Y,Z)
23 IPLACE = NODE
24 IF (ABS(X - VAL(JXCC,NODE)) .GT. VAL(JKLEN,NODE)/2. .OR.
+ ABS(Y - VAI(JYCC,NODE)) .GT. VAL(JYLEN,NODE)/2.)
+ IPLACE = NODE2
25 RETURN

C

C DETERMINING NODE IS A PORTAL OR STAIRS NODE.

C

26 30 D = DIST(ISOURC,ISINK)
27 IPLACE = NODE2
28 IF (D*FRAC .LE. EPS) IPLACE = NODE
29 RETURN
30 END

UTILITIES - IQUEUE FUNCTION

ISN

1

FUNCTION IQUEUE(IELEM,LIST)

C

C A ROUTINE TO ADD AN ITEM CNTO THE END OF A LIST. TYPICALLY
C "IELEM" WILL BE A SCALAR OR NON-LIST REFERENCE AND "LIST" WILL BE
C A LIST REFERENCE. HOWEVER TO FACILITATE EASY USE OF THIS FUNCTION
C WITHOUT CHECKING FOR SPECIAL CASES, THE FUNCTION IS CODED TO ALLOW
C EITHER PARAMETER TO BE ANY DATATYPE. IF "LIST" IS NULL, THE FUNCTION
C RETURNS "IELEM" AS A VALUE. IF "LIST" IS NON-NULL BUT "IELEM" IS
C NULL, THE FUNCTION RETURNS "LIST" AS A VALUE. IF "LIST" IS NOT
C A LIST REFERENCE, A LIST NODE IS OBTAINED AND "LIST" BECOMES THE
C CONTENTS OF THIS LIST NODE. IF NEITHER "IELEM" NOR "LIST" IS NULL,
C A LIST NODE IS ALSO OBTAINED FOR "IELEM". THIS LIST NODE IS
C CHAINED SO THAT IT FOLLOWS "LIST". THE FUNCTION RETURNS WITH IQUEUE
C REFERENCING THE FRONT OF THE RESULTANT LIST. NOTE THAT IF
C IELEM IS A LIST THEN THE LAST ELEMENT ON THE RETURNED LIST IS
C ITSELF A LIST. SEE ALSO ROUTINES ISTACK AND JOIN FOR RELATED
C OPERATIONS.

C

C INPUT PARAMETERS:

C IELEM AN ELEMENT TO BE ADDED TO A LIST (ANY DATATYPE)
C LIST A LIST REFERENCE (OR ITEM OF ANY OTHER DATATYPE)

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORMT
8 COMMON /PARS1/
9 COMMON /PARS3/

C

10 IF (LIST .NE. NULL) GO TO 5
11 IQUEUE = IELEM
12 RETURN
13 5 IF (IELEM .NE. NULL) GO TO 10

UTILITIES - IQUEUE FUNCTION

```
14      IQUEUE = LIST
15      RETURN
16      C NEITHER "IELEM" NOR "LIST" ARE NULL. DOES "LIST" REFERENCE A LIST ?
17      10     IF (LIST .LT. 1000000000) GO TO 15
18          CALL PARSHF(LIST,IDLTYPE,IRECNO,IDUM)
19          IF (IDLTYPE .EQ. ILIST) GO TO 20
20          C PARAMETER "LIST" IS NOT A LIST. MAKE IT THE CONTENTS OF A LIST NODE.
21          15     IRECNC = NEWREC(LLIST)
22              LISTND(JVAL,IRECNO) = LIST
23              LAST = IRECNC
24              GO TO 25
25          C FIND THE LAST NODE OF "LIST"
26          20     LAST = IRECNO
27          22     IR = LISTND(JNXT,LAST)
28          25     IF (IR .EQ. NULL) GO TO 25
29              LAST = IR
30              GO TO 22
31          C GET A LIST NODE TO ENCASE "IELEM" & APPEND TO END OF LIST
32          25     NEW = NEWREC(LLIST)
33              LISTND(JVAL,NEW) = IELEM
34              LISTND(JNXT,LAST) = NEW
35          C CONVERT THE RECORD INDEX "IRECNO" TO A LIST REFERENCE & RETURN
36          IQUEUE = NEWREF(LLIST,IRECNO,0)
37          RETURN
38          END
```

UTILITIES - IREF FUNCTION

ISN

1 FUNCTION IREF(IFLDNO,IRECRF)

C FUNCTION TO TRANSLATE A FIELD INDEX AND RECORD REFERENCE INTO
C A LOCATION OF STORAGE RELATIVE TO THE ARRAY ITEM (DITEM). FUNCTION
C RETURNS THIS LOCATION IF IT SUCCEEDS. IF AN ERROR IS ENCOUNTERED IN
C PROCESSING THE FUNCTION RETURNS A NULL VALUE INSTEAD.

C

C INPUT PARAMETERS:
C IFLDNO A FIELD INDEX WITHIN THE RECORD OF INTEREST
C IRECRF A RECCRF REFERENCE

C

2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
4 REAL*8 DTPNAM,FIDNAM,FCRECT

C

5 IREF = NULL
C BREAK RECCRF REFERENCE INTO ITS COMPONENTS
6 CALL FARSRF(IRECRF,IDLTYPE,IRECNO,IDLUM)
C CHECK THAT THE FIELD INDEX SPECIFIED HAS A LEGAL VALUE.
7 IF (IFLDNO .GE. 1 .AND. IFLDNO .LE. NFLDS(IDLTYPE)) GO TO 10
8 CALL ERR(4,3,IDLTYPE,IRECNO,IFLDNO)
9 RETURN
C COMPUTE THE STORAGE LOCATION OFFSET AND RETURN
10 10 IREF = IDLOFF(IDLTYPE) + (IRECNO-1)*NFLDS(IDLTYPE) + IFLDNO
11 RETURN
12 END

UTILITIES - IREF1 FUNCTION

1 FUNCTION IREF1(IFLDRF)
C
C FUNCTION TO TRANSLATE A FIELD REFERENCE INTO THE IMPLIED
C LOCATION OF STORAGE RELATIVE TO THE ARRAY ITEM (DITEM). FUNCTION
C RETURNS THIS LOCATION IF IT SUCCEEDS.
C
C INPUT PARAMETERS:
C IFLDRF A FIELD REFERENCE
C
2 COMMON /PARS/
3 EQUIVALENCE (NULL,NULL), (FAIL,FAIL)
4 REAL*8 DTPNAM,FLDNAM,FCBECT
C
C BREAK RECORD REFERENCE INTO ITS COMPONENTS
5 CALL PARSRF(IFLDRF, IDTYP,IRECNO,IFLDNO)
C COMPUTE THE STORAGE LOCATIONS OFFSET AND RETURN
6 10 IREF1 = IDTOFF(IDTYP) + (IRECNO-1)*NFLDS(TDTYP) + IFLDNO
7 RETURN
8 END

UTILITIES - ISTACK FUNCTION

ISN

1

FUNCTION ISTACK(IELEM,LIST)

C

C A ROUTINE TO PLACE AN ITEM ONTO THE FRONT OF A LIST. TYPICALLY
C "IELEM" WILL BE A SCALAR OR NON-LIST REFERENCE AND "LIST" WILL BE
C A LIST REFERENCE. HOWEVER TO FACILITATE EASY USE OF THIS FUNCTION
C WITHOUT CHECKING FOR SPECIAL CASES, THE FUNCTION IS CODED TO ALLOW
C EITHER PARAMETER TO BE ANY DATATYPE. IF "LIST" IS NULL, THE
C FUNCTION RETURNS "IELEM" AS A VALUE. IF "LIST" IS NON-NULL BUT
C "IELEM" IS NULL, THE FUNCTION RETURNS "LIST" AS A VALUE. IF
C "LIST" IS NOT A LIST REFERENCE, A LIST NODE IS OBTAINED AND "LIST"
C BECOMES THE CONTENTS OF THIS LIST NODE. IF NEITHER "IELEM" NOR
C "LIST" IS NULL A LIST NODE IS ALSO OBTAINED FOR "IELEM". THIS LIST
C NODE IS CHAINED IN PBCNT OF THE LIST CONTAINING THE SECOND
C ARGUMENT. ISTACK THEN REFERENCES THE FRONT OF THIS EXPANDED LIST
C UPON RETURN. NOTE THAT IF IELEM IS A LIST THEN THE FIRST ELEMENT
C OF THE RETURNED LIST IS A LIST. SEE ALSO ROUTINES IQUEUE AND JOIN
C FOR RELATED OPERATIONS.

C

C INPUT PARAMETERS:

C IELEM AN ELEMENT TO BE ADDED TO A LIST (ANY DATATYPE)
C LIST A LIST REFERENCE (OR ITEM OF ANY OTHER DATATYPE)

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (PNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FORECT
8 COMMON /PARS1/
9 COMMON /PARS3/
10 COMMON /RECFREQ/
11 DIMENSION RECFREQ(140)
12 EQUIVALENCE (RECFREQ(1),IGCALS)

C
13 IF (LIST .NE. NULL) GO TO 5

UTILITIES - ISTACK FUNCTION

14 ISTACK = IELEM
15 RETURN
16 5 IF (IELEM .NE. NULL) GO TO 10
17 ISTACK = LIST
18 RETURN
C NEITHER "IELEM" NOR "LIST" ARE NULL. DOES "LIST" REFERENCE A LIST?
19 10 IF (LIST .LT. 1000000000) GO TO 15
20 CALL PARSEF(LLIST, IDTYPE,IRECNO, IDUM)
21 IF (IDTYP .EQ. LLIST) GO TO 20
C THE PARAMETER "LIST" IS NOT A LIST. MAKE IT THE CONTENTS
C OF A LIST NODE.
22 15 IRECNO = NEWREC(LLIST)
23 LISTND(JVAL,IRECNO) = LIST
24 ISTACK = NEWREF(LLIST,IRECNO,0)
C GET A LIST NODE TO ENCASE "IELEM"
25 20 LOCAL(5) = ISTACK
26 NEW = NEWREC(LLIST)
27 LISTND(JVAL,NEW) = IELEM
28 LISTND(JNXT,NEW) = IRECNC
C CONVERT THE RECORD INDEX "NEW" TO A LIST REFERENCE AND RETURN
29 ISTACK = NEWREF(LLIST,NEW,0)
30 RETURN
31 END

UTILITIES - IVAL FUNCTION

ISN

```
1      FUNCTION IVAL(IFLDNO,IRECRF)
C
C      FUNCTION TO RETURN THE VALUE REFERRED TO BY A FIELD INDEX AND
C      RECORD REFERENCE.  IF ANY ERRORS ARE DETECTED IN THE ARGUMENTS
C      TO THE FUNCTION, A NULL VALUE IS RETURNED INSTEAD.
C
C      INPUT PARAMETERS:
C          IFLDNO    FIELD INDEX
C          IRECRF   RECDREFERENCE
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,BULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCEMOT
C
8      IVAL = NULL
C      BREAK RECDREFERENCE INTO ITS COMPONENTS
9      CALL FABSRP(IRECRF, IDTYP,IRECNO, IDUM)
C      CHECK THAT THE FIELD INDEX SPECIFIED HAS A LEGAL VALUE
10     IF (IFLDNO .GE. 1 .AND. IFLDNO .LE. NPLDS(IDTYP)) GO TO 10
11     CALL ERR(4,4, IDTYP,IRECNO,IFLDNO)
12     RETURN
C      COMPUTE THE STORAGE LOCATION OFFSET
13     IR = ITOFF(IDTYP) + (IRECNO-1)*NPLDS(IDTYP) + IFLDNO
C      RETURN THE VALUE STORED AT THIS LOCATION
14     IVAL = ITEM(IR)
15     RETURN
16     END
```

UTILITIES - IVAL1 FUNCTION

ISN

```
1      FUNCTION IVAL1(IFLDREF)
C
C      FUNCTION TO RETURN THE VALUE REFERRED TO BY A FIELD
C      REFERENCE.
C
C      INPUT PARAMETERS:
C          IFLDREF   FIELD REFERENCE
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTEIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IPAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FOEMOT
C
C      BREAK RECORD REFERENCE INTO ITS COMPONENTS
8      CALL FABSRF(IFIERF,IDTYP,IRECNO,IFLDNO)
C      COMPUTE THE STORAGE LOCATION OFFSET
9      10     IR = IDTOFF(IDTYP) + (IRECNO-1)*NFLDS(IDTYP) + IFLDNO
C      RETURN THE VALUE STORED AT THIS LOCATION
10    IVAL1 = ITEM(IR)
11    RETURN
12    END
```

UTILITIES - JOIN FUNCTION

ISN

```
1      FUNCTION JOIN(LIST1,LIST2)
C
C      FUNCTION TO APPEND LIST2 TO LIST1.  IF BOTH ARGUMENTS
C ARE LISTS, THEN A NEW RESULTANT LIST IS FORMED HAVING THE
C SUM OF THE NUMBER OF ELEMENTS OF EACH OF THE COMPONENT LISTS.
C EITHER OR BOTH OF THE ARGUMENTS, HOWEVER, MAY BE OF A DATATYPE
C OTHER THAN A LIST.  IF EITHER OR BOTH IS NULL IT IS IGNORED IN
C PRODUCING THE FINAL RESULT.  IF EITHER OR BOTH IS A SINGLE
C ITEM RATHER THAN A LIST, IT IS TREATED AS A SINGLE ENTITY TO
C BE JOINED TO THE OTHER TO FORM A RESULTING LIST.  THE FUNCTION
C RETURNS A POINTER TO THE RESULTANT LIST.  SEE ALSO THE FUNCTIONS
C ISTACK AND IQUEUE WHICH PERFORM RELATED ACTIONS.
C
C      INPUT PARAMETERS:
C      LIST1      A LIST (OR VARIABLE OF OTHER DATATYPE)
C      LIST2      A LIST (OR VARIABLE OF OTHER DATATYPE)
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DITEM,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (PNULL,NULL),(IPAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FOBNOT
8      COMMON /PARS1/
9      COMMON /PARS3/
C
10     IF (LIST1 .NE. NULL) GO TO 5
11     JCIN = LIST2
12     RETURN
13     5    IF (LIST2 .NE. NULL) GO TO 10
14     JOIN = LIST1
15     RETURN
C      NEITHER LIST1 NOR LIST2 ARE NULL.  DOES LIST2 REFERENCE A LIST ?
16     10   IF (LIST2 .LT. 1000000000) GO TO 15
17       CALL PARSRF(LIST2, IDTYP, L2, IDUM)
```

UTILITIES - JOIN FUNCTION

ISH

```
18      IF (IDTYP .EQ. IILIST) GO TO 20
      C THE PARAMETER "LIST2" IS NOT A LIST. MAKE IT THE CONTENTS
      C OF A LIST NODE.
19      15    L2 = NEWREC(LLIST)
20          LISTND(JVAL,L2) = IILIST2
      C IS LIST1 A LIST ?
21      20    IF (LIST1 .LT. 1000000000) GO TO 25
22          JOIN = LIST1
23          CALL PARSRF(LIST1, IDTYP, L1, IDUM)
24          IF (IDTYP .EQ. IILIST) GO TO 30
      C IF LIST1 ISN'T ALREADY A LIST, ENCASE IT IN A LIST NODE
25      25    L1 = NEWREC(LLIST)
26          LISTND(JVAL,L1) = LIST1
27          JOIN = NEWREF(LLIST,L1,0)
28          GO TO 35
      C GET LAST NODE ON FIRST LIST
29      30    IF (LISTND(JNXT,L1) .EQ. NULL) GO TO 35
30          L1 = LISTND(JNXT,L1)
31          GO TO 30
      C MAKE 2ND LIST THE SUCCESSOR OF THE LAST ELEMENT OF THE FIRST
      C LIST.
32      35    LISTND(JNXT,L1) = L2
      C RETURN
33          RRETURN
34          END
```

UTILITIES - LCOPY FUNCTION

ISN

```
1      FUNCTION LCOPY(LIST)
C
C      FUNCTION TO COPY A LIST AND RETURN A REFERENCE TO IT.
C
C      INPUT PARAMETERS:
C          LIST      REFERENCE TO LIST TO BE COPIED
C
2      COMMON /PARS/
3      EQUIVALENCE (FNULL,NULL), (IPFAIL,FAIL)
4      REAL*8 DT$NAME,FLDNAM,FCRECT
C
5      LCOPY = NULL
6      ITM = IPFIRST(LIST,KI)
7      10     IF (ITM .EQ. NULL) RETURN
8      LCOPY = IQUEUE(ITM,LIST)
9      ITM = NEXT(LIST,KI)
10     GO TO 10
11     END
```

ISN

UTILITIES - LIST FUNCTION

1

FUNCTION LIST(I1,I2,I3,I4,I5,I6)

C ROUTINE TO CREATE A LIST OF ELEMENTS. AS CURRENTLY FORMULATED
C THIS FUNCTION ALLOWS AT MOST SIX ELEMENTS TO BE PUT ON THE LIST WHEN
C IT IS CREATED. OTHERS CAN BE ADDED AFTER THE LIST HAS BEEN CREATED
C WITH THE ISTACK, IQUEUE, OR JCIN ROUTINES. ALSO, IF THIS RESTRICTION
C OF HAVING AT MOST SIX INITIAL ITEMS TURNS OUT TO BE BOthersome, THE
C NUMBER CAN EASILY BE EXTENDED TO ANY REASONABLE FIXED NUMBER. THE
C ARGUMENTS TO THE FUNCTION ARE ELEMENTS TO BE PLACED ON THE LIST,
C IN THE ORDER THEY ARE TO APPEAR ON THE LIST. IF IT IS DESIRED TO
C CREATE A LIST WITH N INITIAL ELEMENTS, $0 \leq N \leq 6$, THEN ONLY THOSE
C ELEMENTS TO GO ON THE LIST SHOULD BE SPECIFIED, USING INPUT
C ARGUMENTS I1 THROUGH IN. IN THIS CASE ARGUMENT I(N+1) SHOULD BE
C GIVEN A NULL VALUE TO MARK THE END OF THE SET OF ELEMENTS. THE
C CALLING ROUTINE MAY THEREFORE SPECIFY FROM 1 TO 6 ARGUMENTS IN
C THE CALL.

C THE ELEMENTS TO BE PUT ON THE LIST MAY BE OF ANY DATATYPE,
C EXCEPT THAT A NULL VALUE CANNOT BE SPECIFIED AS AN ELEMENT TO BE
C ADDED TO THE LIST. IF A REAL NUMBER IS PASSED TO THIS FUNCTION
C IT WILL BE PUT ON THE LIST WITH ITS ORIGINAL CONFIGURATION WITHOUT
C ANY CONVERSION.

C THE FUNCTION RETURNS A REFERENCE TO THE LIST CREATED, OR A NULL
C VALUE IF IT FAILS TO CREATE A LIST DUE TO EXHAUSTION OF LISTND
C STORAGE.

C

C INPUT PARAMETERS:

C I1-I6 ELEMENTS TO BE PLACED ON A LIST. IF FEWER THAN 6
C ELEMENTS ARE DESIRED TO BE ON THE LIST INITIALLY,
C THE N+1 PARAMETER SHOULD BE NULL.

C

2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULI),(IFAIL,FAIL)

UTILITIES - LIST FUNCTION

```
15N
 7      REAL*8 DTPNAM,FLDNAM,FOBJECT
 8      COMMON /PARS1/
 9      COMMON /PARS3/
C
10     DIMENSION II(6)
C
11     LIST = NULL
12     IF (I1 .EQ. NULL) RETURN
C
C FIRST PUT THE ELEMENTS IN A VECTOR TO MAKE FURTHER PROCESSING MORE
C UNIFORM
13     N = 1
14     II(N) = I1
15     IF (I2 .EQ. NULL) GO TO 10
16     N = N + 1
17     II(N) = I2
18     IF (I3 .EQ. NULL) GO TO 10
19     N = N + 1
20     II(N) = I3
21     IF (I4 .EQ. NULL) GO TO 10
22     N = N + 1
23     II(N) = I4
24     IF (I5 .EQ. NULL) GO TO 10
25     N = N + 1
26     II(N) = I5
27     IF (I6 .EQ. NULL) GO TO 10
28     N = N + 1
29     II(N) = I6
C
30   10   I = 0
C GET FIRST LIST NODE
31     LIST = NEWREC(ILIST)
32     IF (LIST .EQ. NULL) RETURN
C SET PCINTER TO CURRENT LIST NODE ELEMENT
33     LPT = LIST
C SET INDEX OF LIST ELEMENT WORKING ON
```

UTILITIES - LIST FUNCTION

ISN

```
31      15      I = I + 1
      C FILL IN THE CONTENTS OF THE LIST NODE.
35          LISTND(JVAL,LPT) = II(I)
      C IF HAVE REACHED THE END OF THE LIST, EXIT
36          IF (I .GE. N) GO TO 20
      C OTHERWISE GET A NEW LIST NODE
37          NEW = NEWREC(LLIST)
38          IF (NEW .EQ. NULL) GO TO 20
      C JOIN THE NEW LIST NODE TO THE PREVIOUS ONE
39          LISTND(JNXT,IPT) = NEW
      C MAKE THE NEW LIST NODE THE CURRENT ONE & MAKE ANOTHER ITERATION
40          LPT = NEW
41          GO TO 15
      C
      C CONVERT THE RECORD INDEX IDENTIFYING THE FIRST LISTND ON THE LIST
      C TO A REFERENCE TO THE LIST
42      20      LIST = NEWREF(LLIST,LIST,0)
43      RETURN
44      END
```

UTILITIES - LSERCH FUNCTION

ISN

```
1      FUNCTION LSERCH(LISREF,IEECRF,IDTYP,IFLD,IFLDV)
C ... LSERCH IS A FUNCTION TO SEARCH A LIST FOR A PARTICULAR
C ENTRY. SEARCH CRITERIA CAN BE EITHER A RECORD REFERENCE,
C A RECORD TYPE, A RECORD TYPE AND FIELD VALUE, OR JUST A
C FIELD VALUE. VALUE RETURNED IS NULL IF THE SEARCH FAILS
C AND THE ENTRY IF THE SEARCH SUCCEEDS.
C
C     INPUT PARAMETERS:
C         LISREF ... LIST TO BE SEARCHED
C         IEECRF ... RECORD REFERENCE CRITERION (OR NULL)
C         IDTYP ... RECORD TYPE CRITERION (OR NULL)
C         IFLD ... FIELD TO BE COMPARED (OR NULL)
C         IFLDV ... FIELD VALUE CRITERION (IGNORED IF IFLD IS NULL
C
2     COMMON /PARS/
3     EQUIVALENCE (FNULL,NOLL),(IPAIL,FAIL)
4     REAL*8 DTPNAM,FLDNAM,FCBPC
C
5     LSERCH=NULL
C
C ... START LIST LOOP
C
6     IREF=IFIRST(LISREF,LETB)
7     10    IF(IREF.EQ.NULL) RETURN
8     CALL FARSBP(IREF,JDTYP,JSEC,JFLD)
9     IF(IEECRF.EQ.NULL) GO TO 20
C
C ... RECORD REFERENCE CRITERION SET
C
10    IF(IREF.NE.IRECRF) GO TO 50
11    20    IF(IDTYP.EQ.NULL) GO TO 30
C
C ... RECORD TYPE CRITERION SET
C
12    IF(IDTYP.NE.JDTYP) GO TO 50
```

UTILITIES - LSERCH FUNCTION

ISN
13 30 IF(IFLD.EQ.NULL) GO TO 40
C
C ... FIELD VALUE CRITERION SET
C
14 IF(IFLDVL.NE.IVAL(IFLD,IREF)) GO TO 50
C
C ... ALL CRITERIA MET, RETURN IREF
C
15 40 LSERCH=IREF
16 RETURN
C
C ... END LIST LOCF
C
17 50 IREF=NEXT(LISREF,IPTR)
18 GO TO 10
19 END

UTILITIES - NEWRC2 FUNCTION

ISN

```

1      FUNCTION NEWRC2(IDTYP,V1,V2)
2      ENTRY NEWRC3(IDTYP,V1,V2,V3)
3      ENTRY NEWRC4(IDTYP,V1,V2,V3,V4)
4      ENTRY NEWRC5(IDTYP,V1,V2,V3,V4,V5)
5      ENTRY NEWRC6(IDTYP,V1,V2,V3,V4,V5,V6)
6      ENTRY NEWRC7(IDTYP,V1,V2,V3,V4,V5,V6,V7)
7      ENTRY NEWRC8(IDTYP,V1,V2,V3,V4,V5,V6,V7,V8)
8      ENTRY NEWRC9(IDTYP,V1,V2,V3,V4,V5,V6,V7,V8,V9)
9      ENTRY NEWR18(IDTYP,V1,V2,V3,V4,V5,V6,V7,V8,V9,
+          V10,V11,V12,V13,V14,V15,V16,V17,V18)

C
C      FUNCTION TO GET A VACANT RECORD OF A SPECIFIED TYPE, FILL ITS
C      FIELDS WITH SPECIFIED VALUES, AND TO OUTPUT A DESCRIPTION OF THE
C      NEW RECORD TO THE FILE BECCBING THE LIST OF STATE VARIABLE CHANGES.
C      THE FUNCTION RETURNS THE INDEX TO THE NEW RECORD IN THE FILE OF
C      RECORDS OF THIS TYPE.
C
C      INPUT PARAMETERS:
C      IDTYP    RECORE TYPE TO BE ALLOCATED
C      V1       | VALUES FOR UP TO 20 FIELDS OF THE RECORD TO BE INITIAL-
C      ...      | IZED. (CURRENTLY NO RECORD HAS MORE THAN 20 FIELDS).
C      V20      | NEWRC2 USES IDTYP TO DETERMINE HOW MANY VALUES TO
C                  | EXPECT. THE CALLING ROUTINE MUST PROVIDE AT LEAST
C                  | THIS MANY VALUES IN ITS CALL TO NEWRC2.
C
10     COMMON /STATEV/
11     DIMENSION ITEM(41900),DITEM(41900)
12     EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
13     COMMON /PARS/
14     EQUIVALENCE (PNULL,NULL),(IPFAIL,FAIL)
15     REAL*8 DTPNAM,FLDNAM,RECT
16     COMMON /RECREP/
17     DIMENSION RECBFQ(140)
18     EQUIVALENCE (RECBFQ(1),IECALS)
C

```

UTILITIES - NEWRC2 FUNCTION

1SN
19 DIMENSION VEC(20)
C
C ALLOCATE A NEW RECCED
C
20 IREC : NEWREC(IDTYP)
C
C DETERMINE THE NUMBER OF FIELD VALUES ASSOCIATED WITH A RECORD OF
C THIS TYPE.
21 N = NPIDS(IDTYP)
C FILL IN VALUES ACCORDING TO N.
22 IF (N .GT. 20) CALL EEE(38,34,N,20,0)
23 GO TO (20,19,18,17,16,15,14,13,12,11,10,9,8,7,6,5,4,3,2,1),N
24 1 VEC(20) = V20
25 2 VEC(19) = V19
26 3 VEC(18) = V18
27 4 VEC(17) = V17
28 5 VEC(16) = V16
29 6 VEC(15) = V15
30 7 VEC(14) = V14
31 8 VEC(13) = V13
32 9 VEC(12) = V12
33 10 VEC(11) = V11
34 11 VEC(10) = V10
35 12 VEC(9) = V9
36 13 VEC(8) = V8
37 14 VEC(7) = V7
38 15 VEC(6) = V6
39 16 VEC(5) = V5
40 17 VEC(4) = V4
41 18 VEC(3) = V3
42 19 VEC(2) = V2
43 20 VEC(1) = V1
C
C COPY VECTOR TO RECCED
C
44 IRECRE = NEWBEE(IDTYP,IREC,0)

UTILITIES - NEWRC2 FUNCTION

1SN
45 I = IBFF(1,IRECFF)
46 CALL CCPY(VEC,DITEM(I),N)
47 LCCAL(1)=IRECFF
C
C REPORT RECORD
C
48 CALL OUTPLX(IDTYP,IREC,IFEC,6,1,0)
C
49 GO TO (21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38),N
50 21 CONTINUE
51 22 NEWBC2 = IRECF
52 RETURN
53 23 NEWBC3 = IRECF
54 RETURN
55 24 NEWBC4 = IRECF
56 RETURN
57 25 NEWBC5 = IRECF
58 RETURN
59 26 NEWBC6 = IRECF
60 RETURN
61 27 NEWBC7 = IRECF
62 RETURN
63 28 NEWBC8 = IRECF
64 RETURN
65 29 NEWBC9 = IRECF
66 RETURN
67 30 CONTINUE
68 31 CCNTINUE
69 32 CONTINUE
70 33 CCNTINUE
71 34 CONTINUE
72 35 CCNTINUE
73 36 CONTINUE
74 37 CCNTINUE
75 38 NEWB18 = IRECF
76 RETURN

ISBN
77

END

UTILITIES - NEWRC2 FUNCTION

UTILITIES - NEWREC FUNCTION

ISN

```
1      FUNCTION NEWREC(IDTYP)
C
C      FUNCTION TO GET AN EMPTY RECORD OF A SPECIFIED TYPE FROM THE
C      FILE OF SUCH RECORDS AND TO RETURN ITS RECORD INDEX.  IF NO
C      EMPTY RECORDS CAN BE FOUND THE ROUTINE CALLS THE ERROR ROUTINE AND
C      RETURNS A NULL VALUE.
C
C      INPUT PARAMETERS:
C      IDTYP      POS INTEGER SPECIFYING THE RECORD'S DATA TYPE
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(FAIL,FAIL)
7      REAL*8 DTPNAM,FIDNAM,FCBECT
8      COMMON /PARS1/
9      COMMON /PARS3/
C
10     NEWREC = NULL
C      MAKE SURE IDTYP IS IN RANGE
11     IF (IDTYP .GT. 1 .AND. IDTYP .LE. NDTYPE) GO TO 5
12     CALL ERF(1,1,IDTYP,0,0)
13     RETURN
C
C      IS THERE A VACANT RECCED ?
14     5    IF (IAVAIL(IDTYP) .NE. NULL) GO TO 20
15     IF (LSTREC(IDTYP) .LT. NRECS(IDTYP) - NGENRC) GO TO 10
16     CALL COINCT
17     IF (IAVAIL(IDTYP) .NE. NULL) GO TO 20
18     CALL ERF(2,1,IDTYP,0,0)
19     RETURN
C
C      THE FILE HAS NOT YET BEEN FILLED.  GET THE NEXT VACANT RECORD.
20     10   LSTREC(IDTYP) = LSTREC(IDTYP) + 1
```

UTILITIES - NEWREC FUNCTION

ISN

21 NEWREC = LSTREC(IOTYP)
22 RETURN
C
C A RECORD HAS BEEN RECLAIMED. USE IT.
23 20 NEWREC = IAVAIL/IOTYP)
24 IRECRP=NEWREC/IOTYP,NEWREC,0)
25 IAVAIL(IOTYP) = JVAL(JNXT,IRECRP)
C NULL OUT THE NEXT ECINTEE OF THE NEW RECORD
26 I = IBEF(JNXT,IRECRP)
27 ITEM(I) = NULL
28 RETURN
29 END

UTILITIES - NEWREF FUNCTION

```
1      FUNCTION NEWREF(IDTYP,IRECNO,IFLDNO)
C
C      FUNCTION TO PACK TOGETHER A RECORD DATATYPE (YYY),
C      A RECORD INDEX (ZZZ), AND (OPTIONALLY) A FIELD INDEX (XX) TO
C      FORM A FIELD OR RECORD REFERENCE OF THE FORM: 1XXYYYZZZ.
C      THE FUNCTION RETURNS THIS RECORD REFERENCE.
C
C      INPUT PARAMETERS:
C          IDTYP    RECORD DATATYPE
C          IRECNO   RECORD INDEX
C          IFLDNO   FIELD INDEX
C
2      NEWREF = 1000000000 + IFLDNO*10000000 + IDTYP*10000 + IRECNO
3      RETURN
4      END
```

ISN

UTILITIES - NEXT FUNCTION

1 FUNCTION NEXT(IARG,ICURNC)

C

C FUNCTION TO RETURN THE NEXT VALUE OF THE LIST SPECIFIED BY IARG.

C THE CURRENT ITEM POINTER, ICURNO, IDENTIFIES THE CURRENT

C POSITION ON THE LIST. IF IARG DOES NOT REFER TO A LIST OR IF THE

C LIST'S CURRENT ITEM POINTER ALREADY REFERENCES THE LAST ELEMENT OF

C THE LIST, THE FUNCTION SETS A FAILURE SIGNAL AND RETURNS A NULL

C VALUE. NOTE THAT "NEXT" SHOULD NOT BE

C CALLED UNLESS IFIRST WAS INITIALLY CALLED TO INITIALIZE THE

C CURRENT POINTER TO REFERENCE THE BEGINNING OF THE LIST.

C

C INPUT PARAMETERS:

C IARG A VALUE WHICH MAY BE EITHER (1) A LIST REFERENCE,

C (2) A REFERENCE TO SOME OTHER STRUCTURE, (3) AN INTEGER,

C OR (4) NULL.

C ICURNO THE RECORD INDEX OF THE LIST NODE AT THE

C CURRENT POSITION IN THE LIST.

C

C OUTPUT PARAMETERS:

C ICURNO UPDATED TO POINT TO THE NEXT LOGICAL LISTEND OF

C THE LIST, OR TO NULL IF THERE ARE NO MORE ITEMS ON THE

C LIST. NOT CHANGED IF IARG ISN'T A LIST.

C

2 COMMON /STATEV/

3 DIMENSION ITEM(41900),DITEM(41900)

4 EQUIVALENCE (DTMIN,ITEP(1),DITEM(1))

5 COMMON /PARS/

6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)

7 REAL*8 DTPNAM,FLDNAM,FCRECT

8 COMMON /PARS1/

9 COMMON /PARS3/

C

10 IRTNCD = IFAIL

11 NEXT = NULL

C CHECK WHETHER THE ARGUMENT IS A LIST

UTILITIES - NEXT FUNCTION

1SN
12 IF (IARG .LT. 1000000000) RETURN
13 CALL PARSRP(IARG, IDTYP, IRECNO, IDUM)
14 IF (IDTYP .NE. ILIST) RETURN
C CHECK WHETHER THE CURRENT PCINTER HAS A LEGAL VALUE
15 IF (ICURNO .GE. 1 .AND. ICURNO .LE. NRECS(LLIST)) GO TO 5
16 CALL ERR(7,5,IARG,ICUENO,0)
17 RETURN
C IF DO HAVE A LIST, UPDATE THE CURRENT ELEMENT POINTER
18 5 ICUENO = LISTND(JNXT,ICUENO)
C IF THERE IS NO NEXT ITEM, RETURN
19 IF (ICURNO .LE. 0) RETURN
C IF THERE IS A NEXT ITEM CN THE LIST, GET ITS VALUE
C AND RETURN
20 NEXT = LISTND(JVAL,ICURNO)
21 IRINCD = ICR
22 RETURN
23 END

UTILITIES - NITEMS FUNCTION

1 FUNCTION NITEMS(N,LIST,IREM)

C ROUTINE TO SELECT N ITEMS (ARBITRARILY) FROM A LIST AND
C TO RETURN BOTH A LIST OF THE ITEMS SELECTED AND A LIST OF THOSE
C NOT SELECTED.

C THE CURRENT VERSION OF THE ROUTINE SELECTS THE FIRST N ITEMS
C IT ENCOUNTERS. IF THERE ARE FEWER THAN N ITEMS ON THE LIST IT
C RETURNS AS MANY ITEMS AS THERE ARE.

C INPUT PARAMETERS:

C N NUMBER OF ITEMS TO SELECT
C LIST LIST OF ITEMS

C OUTPUT PARAMETERS:

C IREM LIST OF REMAINING ITEMS NOT SELECTED.

2 COMMON /PARS/
3 EQUIVALENCE (NULL,NULL),(FAIL,FAIL)
4 REAL*8 DTPNAM,FIDNAM,FCFFECT
5 COMMON /PARS1/
6 COMMON /PARS2/
7 COMMON /PARS3/

C

8 NS = 0
9 IREM = LIST
10 NITEMS = NULL
11 IF (N .LE. 0) RETURN

C

12 IR = IFIRST(LIST,KI)
13 10 IF (IR .EQ. NULL) GO TO 20
14 NITEMS = ISTACK(IR,NITEMS)
15 IR = NEXT(LIST,KI)
16 NS = NS + 1
17 IF (NS .LT. N) GO TO 10

C

UTILITIES - NITEMS FUNCTION

ISN

```
C HAVE ENOUGH ITEMS. MAKE KI A LIST REFERENCE
18      IREM = NEWREF(ILIST,KI,0)
19      RETURN
C
C HAVE EXHAUSTED LIST
20  20  IREM = NULL
21      RETURN
22      END
```

UTILITIES - NRTEST FUNCTION

ISN

1

FUNCTION NRTEST(IRELRF)

C
C FUNCTION TO TEST WHETHER A RELATION IS SATISFIED.
C THERE ARE TWO CLASSES OF CONSTRUCTS BEING TREATED AS "RELATIONS":
C (1) A TRUE BINARY RELATION, AND (2) A CONDITIONAL ACTION. BOTH ARE
C ENCODED IN RELATION RECORDS. HOWEVER THEY REPRESENT DISTINCT,
C THOUGH RELATED, CONCEPTS. THE TRUE BINARY RELATION IS EXEMPLIFIED
C BY A RELATION SUCH AS "EQUAL(X,Y)" OR "GREATER.THAN(X,Y)". A TRUE
C BINARY RELATION MAY BE NOT ONLY A SIMPLE BINARY RELATION BUT AN
C EXPRESSION COMPOSED OF SIMPLE BINARY RELATIONS USING "AND", "OR",
C AND "NOT" CONNECTIVES. THUS IT CORRESPONDS ROUGHLY TO A LOGICAL
C EXPRESSION IN FORTRAN. A CONDITIONAL ACTION "RELATION" DIFFERS IN
C THAT IT CONSISTS OF TWO DISTINCT COMPONENTS. THE FIRST IS A TRUE
C BINARY RELATION. THE SECOND IS AN ACTION RECORD. EXAMPLES OF
C CONDITIONAL ACTIONS ARE "IF(RELATION,ACTION)" OR "WHENEVER(RELATION,
C ACTION)".
C IF THE RELATION PASSED TO THE FUNCTION IS A TRUE BINARY RELATION
C IT IS EVALUATED FOR TRUTH. IF TRUE, THE FUNCTION RETURNS 1;
C OTHERWISE IT RETURNS 0. IF THE RELATION PASSED TO NRTEST IS A
C CONDITIONAL ACTION, THE FIRST ARGUMENT (A TRUE BINARY RELATION) IS
C EVALUATED FOR TRUTH. IF TRUE, THE FUNCTION RETURNS A REFERENCE TO
C THE SECOND ARGUMENT (AN ACTION, OR LIST OF ACTIONS); OTHERWISE THE
C FUNCTION RETURNS ZERO.
C
C INPUT PARAMETERS:
C IRELRF A REFERENCE TO A RELATION RECORD
C
2 COMMON /PARS1/
3 COMMON /PARS3/
4 COMMON /PARS4/
C
5 DIMENSION IREFS(10),IBETBN(10)
6 LOGICAL RESULT(10)
7 DATA NLEVS/10/
C

UTILITIES - NRTEST FUNCTION

```

ISN
8      IRECRF = IRELRF
9      ILEV = 0
C      GET LEVEL OF CURRENT RELATION
10     2      ILEV = ILEV + 1
11      IF (ILEV .GT. NLEVS) CALL ERR(28,28,NLEVS,IRELRF,0)
12      CALL PABSRP(IRECRF, IDTYP, IDUM, IDUM)
13      IF (IDTYP .NE. LRELN) CALL ERR(29,28,IRELRF, IDTYP,0)
14      IRTYPE = IVAL(JTYPE,IRECRF)
15      IF (IRTYPE .LE. 0 .OR. IRTYPE .GT. NRELTP) CALL ERR(30,28,IRTYPE,
+          IRECRF,IRELRF)
16      GO TO (5,5,5,5,5,65,65,65,65,65,130,140,150,157,157,157),
+          IRTYPE
C
C      GET VALUE OF FIRST ARGUMENT OF RELATION (VALUE IS INTEGER)
C
17     5      IPAR1 = IVAL(JPAR1,IRECRF)
C      IS THE VALUE AN INTEGER OR REFERENCE ?
18      IF (IPAR1 .LE. 1000000000) GO TO 7
C      VALUE IS A REFERENCE. DETERMINE WHETHER IT REFERS TO A FUNCTION.
19      CALL PABSRP(IPAR1, IDTYP, IDUM, IDUM)
20      IF (IDTYP .NE. LFUNC) IPAR1 = IVAL1(IPAR1)
21      IF (IDTYP .EQ. LFUNC) IPAR1 = IFNCVL(IPAR1)
C
C      GET VALUE OF SECOND ARGUMENT OF RELATION
C
22     7      IPAR2 = IVAL(JPAR2,IRECRF)
C      IS THE VALUE AN INTEGER ?
23      IF (IPAR2 .LE. 1000000000) GO TO 8
C      VALUE IS A REFERENCE. DETERMINE WHETHER IT REFERS TO A FUNCTION.
24      CALL PABSRP(IPAR2, IDTYP, IDUM, IDUM)
25      IF (IDTYP .NE. LFUNC) IPAR2 = IVAL1(IPAR2)
26      IF (IDTYP .EQ. LFUNC) IPAR2 = IFNCVL(IPAR2)
C
C      BRANCH TO CODE TO EVALUATE RELATION
C
27     8      GO TO (10,20,30,40,50,60),IRTYPE

```

ISN

UTILITIES - NRTEST FUNCTION

```
C
C LT(INTEGER,INTEGER)
C
28 10      RESULT(ILLEV) = IPAR1 .LT. IPAR2
29      GO TO 500
C
C LE(INTEGER,INTEGER)
C
30 20      RESULT(ILLEV) = IPAR1 .LE. IPAR2
31      GO TO 500
C
C EQ(INTEGER,INTEGER)
C
32 30      RESULT(ILLEV) = IPAR1 .EQ. IPAR2
33      GO TO 500
C
C NE(INTEGER,INTEGER)
C
34 40      RESULT(ILLEV) = IPAR1 .NE. IPAR2
35      GO TO 500
C
C GT(INTEGER,INTEGER)
C
36 50      RESULT(ILLEV) = IPAR1 .GT. IPAR2
37      GO TO 500
C
C GE(INTEGER,INTEGER)
C
38 60      RESULT(ILLEV) = IPAR1 .GE. IPAR2
39      GO TO 500
C
C GET VALUE OF FIRST ARGUMENT, A REAL, A FIELD REFERENCE, OR A FUNCTION
C
40 65      IR = IRTYPE - 6
41      IPAR1 = IVAL(JPAR1,IREFRF)
42      IF (IPAR1 .GT. 1000000000 .AND. IPAR1 .LE. 1400359999) GO TO 66
```

UTILITIES - NRTEST FUNCTION

ISN

C VALUE IS REAL

43 PAR1 = VAL(JPAR1,IRECRF)

44 GO TO 68

C VALUE IS A REFERENCE

45 66 CALL FARSBF(IPAR1, IDTYP, IDUM, IDUM)

46 IF (IDTYP .NE. IFUNC) PAR1 = VAL1(IPAR1)

47 IF (IDTYP .EQ. IFUNC) PAR1 = FNCVAL(IPAR1)

C

C GET VALUE OF SECOND ARGUMENT, A REAL, A FIELD REFERENCE, OR A

C FUNCTION

C

48 68 IPAR2 = IVAL(JPAR2,IRECBE)

49 IF (IPAR2 .GT. 1000000000 .AND. IPAR2 .LE. 1400359999) GO TO 69

C VALUE IS REAL

50 PAR2 = VAL(JPAR2,IRECRR)

51 GO TO (70,80,90,100,110,120),IR

C VALUE IS A REFERENCE

52 69 CALL FARSBF(IPAR2, IDTYP, IDUM, IDUM)

53 IF (IDTYP .NE. IFUNC) PAR2 = VAL1(IPAR2)

54 IF (IDTYP .EQ. IFUNC) PAR2 = FNCVAL(IPAR2)

55 GO TO (70,80,90,100,110,120),IR

C

C LT(REAL,REAL)

C

56 70 RESULT(ILEV) = PAR1 .LT. PAR2

57 GO TO 500

C

C LE(REAL,REAL)

C

58 80 RESULT(ILEV) = PAR1 .LE. PAR2

59 GO TO 500

C

C EQ(REAL,REAL)

C

60 90 RESULT(ILEV) = PAR1 .EQ. PAR2

61 GO TO 500

1SN

UTILITIES - NRTEST FUNCTION

```
C
C   NE(REAL,REAL)
C
62  100    RESULT(ILLEV) = PAR1 .NE. PAR2
63      GO TO 500
C
C   GT(REAL,REAL)
C
64  110    RESULT(ILLEV) = PAR1 .GT. PAR2
65      GO TO 500
C
C   GE(REAL,REAL)
C
66  120    RESULT(ILLEV) = PAR1 .GE. PAR2
67      GO TO 500
C
C   AND(BELN1,BELN2)
C
C   SAVE SECOND PARAMETER OF RELATION
68  130    IREFS(ILLEV) = IVAL(JPAR2,IRECBF)
C   GET 1ST PARAMETER
69      IRECBF = IVAL(JPAR1,IRECBF)
C   SAVE RETURN POINT
70      IRETURN(ILLEV) = 1
71      GO TO 2
C
C   PROCESS SECOND PORTION OF AND RELATION
C
C   IF RESULT OF FIRST ARGUMENT WAS FALSE, THEN WHOLE RELATION IS FALSE
72  135    RESULT(ILLEV) = BESUIT(ILLEV+1)
73      IF (.NOT.RESULT(ILLEV)) GO TO 500
C   SAVE RETURN POINT
74      IRETURN(ILLEV) = 2
75      GO TO 2
C
C   THE TRUTH OF THE AND RELATION IS THE SAME AS THE TRUTH OF ITS 2ND ARG
```

UTILITIES - NRTEST FUNCTION

ISN

C

76 137 RESULT(ILLEV) = BESUIT(ILLEV+1)

77 GO TO 500

C

C OR(BELN1,BELN2)

C

C SAVE SECOND PARAMETER OF THE RELATION

78 140 IREFS(ILLEV) = IVAL(JPAE2,IBECRF)

C GET 1ST PARAMETER

79 IBECRF = IVAL(JFAE1,ISECEF)

C SAVE RETURN POINT

80 IRETBN(ILLEV) = 3

81 GO TO 2

C

C PROCESS SECOND PORTION OF OR RELATION

C

C IF RESULT OF FIRST ARGUMENT WAS TRUE, WHOLE RELATION IS TRUE.

82 145 RESULT(ILLEV) = RESULT(ILLEV+1)

83 IF (RESULT(ILLEV)) GO TO 500

C SAVE RETURN POINT

84 IRETBN(ILLEV) = 4

85 GO TO 2

C

C THE TRUTH OF THE CR RELATION IS THE SAME AS THE TRUTH OF ITS 2ND ARG

C

86 147 RESULT(ILLEV) = RESULT(ILLEV+1)

87 GO TO 500

C

C NOT(BELN1)

C

C GET 1ST PARAMETER

88 150 IFECRF = IVAL(JFAE1,ISECFF)

C SAVE RETURN POINT

89 IRETBN(ILLEV) = 5

90 GO TO 2

C

UTILITIES - NRTEST FUNCTION

ISN

```

C TRUTH OF THE NOT RELATION IS THE OPPOSITIE OT THAT OF ITS
C EMBEDDED ARGUMENT
C
91 155  RESULT(ILEV) = .NCT.BRESULT(ILEV+1)
92      GC TO 500
C
C COMMON CODE FOR CONDITIONAL ACTIONS
C
93 157  IF (ILEV .NE. 1) CALL ERR(31,28,IRELRF,ILEV,0)
C SAVE 2ND PARAMETERS
94      NRTEST = IVAL(JPAH2,IBECRF)
C SAVE RETURN POINT
95      IBETRN(ILEV) = 6
96      GC TO 2
C
C EVALUATION OF CONDITIONAL ACTIONS DEPENDS ON THE EVALUATION OF ITS
C ARGUMENT AND ON THE TYPE OF THE CONDITIONAL ACTION.
C
97 160  IF (RESULT(ILEV+1) .AND. (IRTYPE .EQ. KIF .OR. IRTYPE .EQ.
+      KWHEN)) RETURN
98      IF (.NOT.RESULT(ILEV+1) .AND. IRTYPE .EQ. KUNTIL) RETURN
99      NRTEST = 0
100     RETURN
C
C POP STACK AND RETURN TO CALIER
C
101 500  ILEV = ILEV - 1
102      IF (ILEV .GE. 1) GC TO 510
103      NRTEST = 0
104      IF (.NOT.RESULT(1)) RETURN
105      NRTEST = 1
106      RETURN
C ARE STILL PROCESSING EMBEDDED RELATIONS. DETERMINE WHERE TO
C BRANCH TO COMPLETE THE EVALUATION OF THE RELATION AND GO THERE.
107 510  IR = IBETRN(ILEV)
108      IRECRF = IREFS(ILEV)

```

UTILITIES - NRTEST FUNCTION

ISN

109

110

GO TO {135, 137, 145, 147, 155, 160}, IR
END

UTILITIES - NULIFY SUBROUTINE

ISN

1 SUBROUTINE NULIFY(IBLOCK,N)

C C EXECUTIVE TO NULL OUT THE CCNTENTS OF A BLOCK OF STORAGE.

C C INPUT PARAMETERES:

C C IBLOCK LOCATION OF THE BEGINNING OF THE BLOCK TO BE NULLED.

C C N SIZE OF THE ELCCK TO BE NULLED

C

2 DIMENSION IBLOCK(N)

3 DATA NULL/-9999/

C

4 DO 10 I=1,N

5 10 IBLOCK(I) = NULL

6 RETURN

7 END

UTILITIES - NZERO SUBROUTINE

ISN

1 SUBROUTINE NZERC(IELOCK, N)

C

C ... SUBROUTINE TO SET CONTENTS OF BLOCK OF STORAGE TO

C INTEGER ZEROES

C

C ... INPUT PARAMETERS:

C

C IELOCK ... ELOCK ADDRESS

C N ... SIZE OF BLOCK

C

2 DIMENSION IBLOCK(N)

3 DO 10 I=1,N

4 10 IELOCK(I)=0

5 RETURN

6 END

UTILITIES - OUTLIS SUBROUTINE

```
1      SUBROUTINE OUTLIS(LISREF,LUNO)
C
C ... OUTLIS IS USED TO PRINT A LIST OF SYMBOLIC
C REFERENCES.
C INPUT PARAMETERS
C     LISREF ... A LIST REFERENCE FOR LIST TO BE PRINTED
C     LUNO   ... LU# FCB CPUTPUT
2     COMMON /PARS/
3     EQUIVALENCE (NULL,NULL), (FAIL,FAIL)
4     REAL*8 DTPNAM,FIDNAM,FCBPC
5     COMMON /OUT/
6     DIMENSION XMIN(50),XMAX(50),DATA(50)
7     EQUIVALENCE (XMAX(1),IMAX(1)),(XMIN(1),IMIN(1))
8     EQUIVALENCE (DATA(1),ICATE(1))
C
C ... IF NOT A LIST RETURN
C
9     I=IFIRST(LISREF,LPTB)
10    IF(LPTR.EQ.NULL) RETURN
11    CALL FARSRF(LISREF,IFF,IIIS,IDUM)
12    MPTR=0
13    MFIDS=0
C
C ... BUILD OUTPUT LIST IN IDATA
C
14   10    IF(I.LT.1000000000) GC TC 20
15    MFDS=MFIDS+1
16    CALL FARSRF(I,ICTYP,IEECIO,IPLD)
17    MPTR=MPTR+1
18    DATA(MPTR)=DTPAER(IDTYP)
19    METR=MPTR+1
20    IDATA(METR)=IRECNC
21    IF(MFIDS.LT.14) GO TO 20
C
C ... LINE COMPLETE, CPUTPUT IT
```

UTILITIES - CUTLIS SUBROUTINE

ISM

C

22 WRITE(LUNO,1000) ILIS,(IDATA(I),I=1,28)

23 MFlds=0

24 MPTR=0

25 20 I=NEXT(LISREF,IPTR)

26 IF(I.NE.NULL) GO TO 10

C

C ... END OF LIST PRINT LINE

C

27 IF(MFLDS.EQ.0) RETURN

28 WRITE(LUNO,1000) ILIS,(IDATA(I),I=1,MPTR)

29 1000 FORMAT(' ',I4,2X,I4,A2,I4,2X)

30 RETURN

31 END

ISN

UTILITIES - OUTPLX SUBROUTINE

```
1      SUBROUTINE OUTPLX(IDTYP,MREC1,MREC2,LUNO,MODE1,MODE2)
C ... A RUTINE TO OUTPUT EFLX RECORDS ON LOGICAL UNIT SPECIFIED
C BY CALLING PROGRAM.
C
C ... PARAMETERS:
C           IDTYP ... RECCED DATATYPE
C           IREC1 ... 1ST RECORD NUMBER TO BE PRINTED
C           IREC2 ... LAST RECCED NUMBER TO BE PRINTED.
C           MODE1 ... 0 PRINT RECORD WITHOUT HEADINGS
C                      1 PRINT RECORD WITH HEADINGS
C           MODE2 ... 0 DO NOT PRINT REFERENCED LISTS
C                      1 PRINT REFERENCED LISTS
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      CCHECN /PARS/
6      EQUIVALENCE (FNULL,NULL),(IPFAIL,FAIL)
7      REAL*8 DTPNAM,FIDNAM,FCBFGT
8      CCMMON /PARS3/
9      CCMMON /OUT/
10     DIMENSION XMIN(50),XMAX(50),DATA(50)
11     EQUIVALENCE (XMAX(1),IMAX(1)),(XMIN(1),IMIN(1))
12     EQUIVALENCE (DATA(1),IEATA(1))
13     REAL*8 BUP(50),ELANK,FET(12)
14     DATA ELANK /8H          /
C
C ... INSURE ARGUMENTS IN RANGE
C
15     IXREF=NULL
16     IF (IDTYP.LE.LLIST .OR. IDTYP.GT.NDTYPE) RETURN
17     IF (ISTBEC(IDTYE).LE.0) RETURN
18     IREC1=MREC1
19     IREC2=MREC2
20     CALL FTNCMD("SET MCFCHECK=OFF",17)
21     IBCMAX=NRECS(IDTYP)-NGENRC
```

ISN UTILITIES - OUTPLX SUBROUTINE

22 IF (IREC1.LE.0) IREC1=1
23 IF (IREC1.GT.IRCMAX) IREC1=IRCMAX
24 IF (IREC2.GT.IRCMAX) IREC2=IRCMAX
25 IF (IREC2.LT.IREC1) IREC2=IREC1
26 IF (IRECIO(IDTYP).NE.0) GO TO 300

C
C ... INITIALIZATION FOR IRECIC=0
C

27 DO 5 I=1,12
28 5 FMT(I)=FORMAT(I, IDTYP)
29 IF (FMT(1).EQ.BLANK) RETURN
30 NBASE=IREF1(NEWBEEF(IDTYP, NRECS(IDTYP), 0))
31 NFLD=NFLDS(IDTYP)

C
C ... COUNT NUMBER OF FIELDS TO BE OUTPUT
C AND MOVE FILENAMES TO EUF
C

32 NPPTR=IPDNPT(IDTYP)
33 NFLD1=1
34 DO 6 I=1,NFLD
35 IF (ITEM(NBASE+I).GT.3) GO TO 6
36 BUF(NFLD1)=FLDNAM(I+NEETB)
37 NFLD1=NFLD1+1
38 6 CONTINUE
39 MLIST=0
40 IF (MCDF1.EQ.0) GO TO 7

C
C ... PRINT HEADING
C

41 IF (LUNO.EQ.6) WRITE(LUNO,1003) DTPNAM(IDTYP)
42 IF (LUNO.NE.6) WRITE(LUNO,1002) DTPNAM(IDTYP)
43 1002 FORMAT(' ', //, A8)
44 1003 FORMAT (' '/ 'NEW ', A8, ' RECORD: ')
45 NN=NFLD1-1
46 WRITE(LUNO,1001) EIAKK, (EUF(I), I=1,NN)

C

UTILITIES - OUTPLX SUBROUTINE

ISN

```
C ... START RECORD PROCESSING ICOP.....  
C  
47    7 DO 100 IREC=IREC1,IREC2  
48        IDATA(1)=IREC  
49        NFR=1  
50        CALL NZERO(IUSED,50)  
51        IPTR=IREF1(NEWREF(IDTYP,IREC,0))  
52        IF(IPTR .EQ. NBASE) GO TO 100  
C  
C ... CHECK FOR NULL RECORD  
C  
53        DO 109 I=2,NFLD  
54        IF(ITEM(IPTR+I) .NE. NULL) GO TO 8  
55 109        CONTINUE  
C  
C ... NULL RECORD UNLESS WEAEON OR EQUIPMENT  
C  
56        IF(IDTYP.NE.LWEAP .AND. IDTYP.NE.LEQUIP) GO TO 100  
C  
C ... SPECIAL CHECK FOR WEAEON OR EQUIPMENT  
C  
57        MTYP=ITEM(IPTR+1)  
58        IF(MTYP .IE. 0) GO TO 100  
59        IF(IDTYP.EQ.LWEAP .AND. MTYP.GT.NWEPTP) GO TO 100  
60        IF(IDTYP.EQ.LEQUIP .AND. MTYP.GT.NEQPTP) GO TO 100  
C  
C ... IF IRECO<>0 CALL OUTSET  
C  
61 8        IF(IRECO(IETYP).EQ.0) GO TO 9  
62        CALL OUTSET(IDTYP,NBASE,NFLD,NFLD1,IPTR,FMT)  
63        IF(IRTNCD.EQ.IFAII) GO TO 100  
C  
C ... BUILD RECORD IN IDATA  
C  
64 9        DC 50 I=1,NFLD  
65        ITYP=ITEM(NEASE+I)
```

UTILITIES - OUTPLX SUBROUTINE

ISN
66 IF (ITYP.GT.3) GC TC 50
67 GO TO (10,20,30),ITYP
C
C ...
C
68 10 NPTB=NPTB+1
69 IDATA(NPTB)=ITEM(IETB+I)
70 IF (IDATA(NPTB).NE.NULL) GO TO 50
71 IDATA(NPTB)=0
72 IUSED(I)=1
73 GO TO 50
C
C ...
C
74 20 NPTB=NPTB+1
75 DATA(NPTB)=DITEM(IPTR+I)
76 IF (DATA(NPTB).NE.NULL) GO TO 50
77 DATA(NPTB)=0.0
78 IUSED(I)=1
79 GO TO 50
C
C ...
C
80 30 IRECRF=ITEM(IPTR+I)
81 IF (IRECRF.NE.NULL) GO TO 35
82 NPTB=NPTB+2
83 IDATA(NPTB)=0
84 IUSED(I)=1
85 GC TO 50
86 35 CALL PARSEF(IRECFF, IDT, IRNO, IPN)
87 NPTB=NPTB+1
88 DATA(NPTB)=DTPAFB(IDT)
89 NPTB=NPTB+1
90 IDATA(NPTB)=IRNO
91 IF (IDT.NE.ILIST) GC TO 50

C

UTILITIES - OUTPLX SUBROUTINE

ISN

```

C ...      SAVE LISREF
C
92        MLIST=ELIST+1
93        IUSED(MLIST+50)=IHECRF
94        50      CONTINUE
C
C ...      PRINT A RECCED
C
95        WRITE(99,FMT) (IDATA(I),I=1,NPTB)
96        READ(99,1000) (EUF(I),I=1,NFLD1)
97        1000 FORMAT(A6,42A8)
98        1001 FORMAT(A6,14A8,/,2X,14A8,/,4X,14A8)
C
C ... BLANK OUT NULL FIELDS
C
99        J=0
100       DO 70 I=1,NFID
101         IF(ITEM(NEASE+I) .GT. 3) GO TO 70
102         J=J+1
103         IF(IUSED(I).EQ.C) GO TO 70
104         EUF(J+1)=ELARR
105       70      CONTINUE
106       WRITE(LUNO,1001) (EUF(I),I=1,NFLD1)
107       100 CONTINUE
C ... END RECORD PROCESSING LCCP..... .
C
C ... IF REQUIRED, PRINT REFERENCED LISTS
C
108       IF(MODE2.EQ.0.OR.ELIST.EQ.0) GO TO 250
109       WRITE(LUNO,1002) DTPNAM(LLIST)
110       DO 200 I=1,MLIST
111         CALL CULTR(IUSED(I+50),LUNO)
112       200 CONTINUE
113       250 IF(IIREF .NE. NULL) CALL DELREC(IIREF)
114         RETURN
C

```

ISN

UTILITIES - OUTPLX SUBROUTINE

```
C ... INITIALIZATION FCB IBECIC=1
C
115 300  MLIST=0
116  IIREF=NEWREF(IDTYP,NRECS(IDTYP),0)
117  IIREF=ICOPY(IIREF)
118  NBASE=IBEF1(IIREF)
119  NFLD=NFILES(IDTYP)
120  IF(MODE1.EQ.0) GO TO 7
C
C ... PRINT HEADING
C
121  N1=IFDNPT(IDTYP)+1
122  N2=N1+NFLDS(IDTYP)-1
123  IF(LUNO.NE.6) WRITE(LUNO,1002) DTPNAM(IDTYP)
124  IF(LUNO.EQ.6) WRITE(LUNO,1003) DTPNAM(IDTYP)
125  WRITE(LUNO,1001) 1ANK,(FIDNAM(I),I=N1,N2)
126  GO TO 7
127  END
```

UTILITIES - OUTREE SUBROUTINE

ISN

```
1      SUBROUTINE OUTREE(IREF,LUNO)
C ... A SUBROUTINE TO PRINT CUT EITHER A PLEX RECORD, OR,
C     IF IREF IS A LIST TO PRINT OUT THE LIST AND THE RECORDS
C     ON IT
2      COMMON /PARS/
3      EQUIVALENCE (NULL,NULL), (IPAIL,PAIL)
4      REAL*8 DTPNAME,FLDNAM,FCFMCT
5      COMMON /PARS1/
6      COMMON /PARS2/
7      COMMON /PARS3/
8      CALL PARSEP(IREF, ID, IBEC, IDUM)
9      IF (ID .EQ. LLIST) GO TO 10
C
C ... SINGLE REFERENCE, PRINT IT
10     CALL CUTPLX(ID,IREC,IFEC,LUNO,0,0)
11     RETURN
C
C ... IREF IS LIST, PRINT LIST
12     10 CALL CUTLIS(IREF,LUNC)
C
C ... PRINT RECORDS ON LIST
13     IBECRF = IFIRST(IREF,LPTR)
14     20 IF (IRECRF .EQ. NULL) GO TO 30
15         CALL PARSEP(IRECRF, ID, IREC, IDUM)
16         CALL OUTFLX(ID,IBEC,IFEC,LUNO,0,0)
17         IRECRF = NEXT(IREF,LPTR)
18         GC TO 20
19     30 RETURN
20     END
```

UTILITIES - OUTSET SUBROUTINE

ISN

```
1      SUBROUTINE OUTSET(IDTYP,NBASE,NFLD,NFLD1,IPTR,FMT)
C ... OUTSET IS CALLED BY OUTEIX FOR RECORDS FOR WHICH
C IRECIC <> 0, I.E. RECCEDS WITH VARIABLE FIELD TYPES.
C THE SETUP TO OUTPUT THESE RECORDS INVOLVES TWO STEPS:
C   1. PLACE APPROPRIATE VALUES IN THE GENERIC
C      RECORD FOR THE DATA FIELD TYPES IN THE
C      RECCED.
C   2. BUILD AN SUITABLE OUTPUT FORMAT IN FMT
C
C INTNCE IS USED TO INDICATE SUCCESS OR FAILURE.
C
C INPUT PARAMETERS:
C   IDTYP ... DATA TYPE OF RECORD
C   IPTR   ... ZERO DISPLACEMENT POINTER TO RECORD
C
C OUTPUT PARAMETERS:
C   NBASE  ... ZERO DISPLACEMENT POINTER TO TYPE RECORD
C   NFLD   ... NUMBER OF FIELDS IN RECORD
C   NFLD1  ... NUMBER OF OUTPUT FIELDS
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCEMCT
8      COMMON /PARS1/
9      COMMON /PARS3/
10     REAL*8 FMTS(7),FMT(12)
11     DATA BLANK /4H      /
12     DATA FMTS /8H,I3,5X ,8H,F6.2,2X,8H,A2,I4 ,8H,2X
      + 8H,8X      ,8H(I4,2X ,EH)      /
C
C ... COMMON SETUP
C
```

UTILITIES - OUTSET SUBROUTINE

ISN

```
13      NPLD=NPLDS(IDTYP)
14      CALL INITVL(FMT,24,PIANK)
15      IF(IDTYP.EQ.LACTN) GO TO 100
16      IF(IDTYP.EQ.IRELB) GO TO 200
17      IF(IDTYP.EQ.LMESS) GO TO 300
18      IF(IDTYP.EQ.ICHNG) GO TO 400
C
C
19      IBINCE=IFAIL
20      RETURN
C
C ... PROCESS ACTION GENERIC RECCRD
C
21      100 ITYP=ITEM(IPTR+JTYPE)
22      J=JEAR1-1
23      DC 110 I=1,3
24      ITEM(NEASE+I+J)=MACYTE(I,ITYP)
25      110 CONTINUE
26      GO TO 500
C
C ... PROCESS RELATION
C
27      200 ITYP=ITEM(IPTR+JTYPE)
28      J=JEAR1-1
29      DC 210 I=1,2
30      ITEM(NEASE+I+J)=MELTE(I,ITYP)
31      210 CONTINUE
32      GO TO 500
C
C ... PROCESS MESSAGE, SET ITYP FOR CONTENT FIELD
C
33      300 IATTR=ITEM(IPTR+JATTR)
34      ISUERF=ITEM(IPTR+JSUEJ)
35      IF(ISUERF.LT.1000000000 .CR. ISUERF.GT.1009999999)
+          GO TO 500
36      CALL FARSRF(ISUERF, IDT, IFB, IFN)
```

UTILITIES - CUTSET SUBROUTINE

ISN

37 MTYPEPT=NEWREF(IDT,NRECS(IDT),0)

38 ITEM(NBASE+JCONT)=IVAL(IATTR,MTYPEPT)

39 GO TO 500

C

C ... PROCESS CHANGE, SET ITYP FOR NEW VALUE FIELD

C

40 400 IATTR=ITEM(IPTR+JATTR)

41 JSUERF = ITEM(IEFR+JSUEJ)

42 CALL PARSRF(JSUERF, IDT, IBB, IFN)

43 MTYPEPT = NEWREF(IDT,NRECS(IDT),0)

44 ITEM(NBASE+JNVAL) = IVAL(IATTR,MTYPEPT)

45 GO TO 500

C

C ... BUILD OUTPUT FORMAT

C

46 500 FMT(1)=FMTS(6)

47 J=2

48 DO 510 I=1,NFLD

49 ITYP=ITEM(NBASE+I)

50 FMT(J)=FMTS(ITYP)

51 J=J+1

52 IF(ITYP.NE.3) GO TO 510

53 FMT(J)=FMTS(4)

54 J=J+1

55 510 CCONTINUE

56 FMT(J)=FMTS(7)

57 NFLD1=NFLD+1

58 IRTNCE=IOK

59 RETURN

60 END

ISN

UTILITIES - PARSRF SUBROUTINE

1 SUBROUTINE PARSRF(IREF, IDTYP, IRECNO, IFLDNO)
C
C ROUTINE TO BREAK APART A FIELD OR RECORD REFERENCE INTO ITS
C COMPONENT PARTS, NAMELY: (1) A RECORD DATATYPE, (2) A RECORD INDEX,
C AND (3) A FIELD INDEX (IF PASSED A FIELD REFERENCE).
C
C INPUT PARAMETERS:
C IREF FIELD OR RECORD REFERENCE (OF FORM: 1XXXXYZZZ)
C WHERE XX IS FIELD INDEX (OR ZERO OTHERWISE)
C YYY IS RECORD DATATYPE
C ZZZZ IS RECORD INDEX
C
C OUTPUT VARIABLES:
C IDTYP DATA TYPE OF RECORD REFERRED TO BY IREF
C IRECNC RECORD INDEX OF RECORD REFERRED TO BY IREF
C IFLENO FIELD INDEX OF FIELD REFERRED TO BY IREF, IF IREF
C REFERS TO A FIELD. OTHERWISE ZERO.
C
2 COMMON /PARS/
3 EQUIVALENCE (FNULL,NULL), (FAIL,FAIL)
4 REAL*8 DTPNAM, FIDNAM, FCFACT
5 COMMON /PARS3/
C
6 C CHECK THAT IREF IS OF PROPER REFERENCE FORM
7 IF (IREF .GT. 1000000000) GO TO 10
8 CALL ERR(2,2,IREF,0,0)
9 RETURN
C
10 C BREAK THE REFERENCE INTO PIECES
11 IR = IREF - (IREF/1000000000) * 1000000000
12 IFLDNC = IR / 10000000
13 IREC = IR - IFLENO * 10000000
14 IDTYP = IREC / 10000
15 IRECNO = IREC - IDTYP * 10000

UTILITIES - PARSRF SUBROUTINE

```
1 ISN
2 C   CHECK THAT THE PARTS ARE WITHIN AN APPROPRIATE RANGE
3     IF (IDTYP .GT. 1 .AND. IETYP .LE. NDTYPE) GO TO 20
4     CALL ERR(1,2,IETYP,0,0)
5     RETURN
6
7 17    20    IF (IEECNO .GE. 0 .AND. IRECNO .LE. NRECS(IDTYP)) GO TO 30
8     CALL ERR(3,2,IETYP,IEECNO,0)
9     RETURN
10   20   30    IF (IFLDNO .LE. NFIDS(IDTYP)) GO TO 40
11     CALL ERR(4,2,IETYP,IRECNO,IFLDNO)
12     RETURN
13
14   C
15   C   FUNCTION SUCCEEDED.  RETURN
16 23   40    RETURN
17 24    END
```

UTILITIES - QXREFS SUBROUTINE

ISN

1

SUBROUTINE QXREFS(IBLCK,N1,N2,ITYPE)

C
C UTILITY ROUTINE USED BY GARBAGE COLLECTOR (COLECT). THE
C ROUTINE ITERATES THROUGH A BLOCK OF STORAGE IN WHICH EXTERNAL
C REFERENCES TO RECCED OB LISTS ARE KNOWN TO BE STORED, MARKS THE
C CORRESPONDING RECCEDS AS BEING ACTIVE, AND STACKS ANY NEW RECORD
C REFERENCES ENCOUNTERED CTO THE ACTIVE RECORD LIST.
C

C INPUT PARAMETERS:

C IBLCK A BLOCK OF STCBAGE CONTAINING EXTERNAL RECORD OR LIST
C REFERENCES ("EXTERNAL" MEANS KNOWN TO BE ACTIVE.)
C THE BLCCK IS VIEWED AS A FILE OF RECORDS (& AS 2-D ARRAY)
C N1 THE NUMBER OF FIELDS IN FILE IBLOCK.
C N2 NUMBER OF RECORDS IN FILE IBLOCK
C ITYPE TYPE CCDE INDICATING CONTENTS OF FIELDS OF IBLOCK
C 1 OR 2 = INTEGER OR REAL
C 3 = RECCED REFERENCE
C 4 = UNKNOWN TYPE
C

C
2 COMMON /PARS/
3 EQUIVALENCE (NULL,NULI),(FAIL,FAIL)
4 REAL*8 DTPNAM,FIDNAM,FCMPCT
5 COMMON /PARS3/
6 COMMON /GARCCI/
7 LOGICAL*1 ACTIVE

C
8 DIMENSION IBLOCK(N1,N2),ITYPE(N1)
C

9 DO 30 IFID=1,N1
10 IF (ITYPE(IFID) .IE. 2) GO TO 30
11 DO 20 IREC=1,N2
12 IF (IBLCK(IFID,IREC) .LT. 1000000000 .OR. IBLOCK(IFID,IREC)
+ .GT. 100999999) GO TO 20
13 CALL PARSE(IELCK(IFID,IREC),IDTYP,IRECNO,IDUM)

UTILITIES - QXREFS SUBROUTINE

1 ISN
2 C SKIP NCN-COLLECTABLE REFERENCES
3 14 IF (IDTYP .GE. ISITE .AND. IDTYP .LE. LWIND) GO TO 20
4 C SKIP OVER NON-COLLECTABLE RECORDS IN PARTIALLY COLLECTABLE FILES
5 15 IF (IDTYP .GE. LWEAP .AND. IDTYP .LE. LACTD .AND.
6 + IRECNO .LE. NFECS0(IDTYP)) GO TO 20
7 C COMPUTE THE INDEX OF THE RECORD MARKER
8 16 IR = IRCCFF(IDTYE) + IRECNO
9 17 NGAP = IRCOFF(LWEAP) - IRCOFF(LSITE)
10 18 IF (IDTYP .GE. LWEAP) IR = IR - NGAP
11 C CHECK TO SEE IF ALREADY ACTIVE
12 19 IF (ACTIVE(IR)) GO TO 20
13 C MARK THE RECORD ACTIVE
14 20 ACTIVE(IR) = .TRUE.
15 C APPEND A REFERENCE TO THE RECORD TO THE STACK
16 21 IQ = IQ + 1
17 22 IF (IQ .GT. NSTACK) CALL ERR(23,17,IQ,0,0)
18 23 IFEQC(IQ) = IBICCK(IPLD,IREC)
19 24 20 CONTINUE
20 25 30 CONTINUE
21 26 RETURN
22 27 END

ISN

UTILITIES - REPORT SUBROUTINE

```
1      SUBROUTINE REPORT(IPLARP,LUNO)
C ... SUBROUTINE TO PRINT A PLAYER SUMMARY FOR IPLARP ON
C LUNC. SUMMARY IS IN SENTENCE FORM INDICATING PLAYER
C NUMBER, CURRENT LOCATION, ACTIVITY, ANY SENSORS
C TRIPPED AND ANY CHANGE IN PHYSICAL STATUS.
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTEIN,ITEM(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCBMOT
8      COMMON /PARS1/
9      COMMON /PARS2/
10     COMMON /PARS3/
11     COMMON /RECREF/
12     DIMENSION RECRFQ(140)
13     EQUIVALENCE (RECRFQ(1),IGOALS)
14     REAL*8 STATUS(4),BUF(14)
15     DATA STATUS/'KILLED','CAPTURED','WOUNDED','WHOLE'/
C
16     1000 FORMAT(' PN',I3,',',A2,I3,') SUPPRESSED')
17     1001 FORMAT(' PN',I3,',',A2,I3,') CAPTURING ',A2,I3)
18     1002 FORMAT(' PN',I3,',',A2,I3,') SURRENDURING TO ',A2,I3)
19     1003 FORMAT(' PN',I3,',',A2,I3,') MOVING TO ',A2,I3)
20     1004 FORMAT(' PN',I3,',',A2,I3,') FIRING AT ',A2,I3)
21     1005 FORMAT(' PN',I3,',',A2,I3,') OBSERVING')
22     1006 FORMAT(' TRIES ALARM',I3,' AT ',A2,I3)
23     1008 FCBMAT(9A8)
24     1009 FORMAT(6X,14A8)
25     1010 FCBMAT(' SEES: ',8A8,/,8X,6A8)
26     1011 FCBMAT(' SEES: ',A2,I4)
C
C ... DETERMINE PLAYER ACTIVITIES
C
27     IEUF = 5
```

UTILITIES - REPORT SUBROUTINE

ISN

28 IACTYP = 0
29 IACTRF = NULL
30 ICAPRF = NULL
31 IACLIS = IVAL(JACTIV,IPLABF)
32 IREF = IFIRST(IACIIS,IFTE)
33 10 IF(IREF .EQ. NULL) GO TO 20
34 IF(IVAL(JTYPEE,IREF) .EQ. KCAPNG) GO TO 12
35 IACTYP = IVAL(JTYPE,IREF)
36 IACTRF = IREF
37 GO TO 14

C

C ... INVOLVED IN CAPTURE

38 12 ICAPRF = IREF
39 14 IREF = NEXT(IACLIS,IPTR)
40 GO TO 10

C

C ... PRINT ACTIVITY FOR EACH OF OUTPUT LINE

C

41 20 LCCBEP = IVAL(JLCCN,IELABF)
42 IREF = IVAL(JPLACE,LOCREF)
43 CALL PABSRP(IREF,ILCC,ILCCN,IDUM)
44 IF(IACTYP .GT. 0) GO TO (40,60,80),IACTYP
45 IF(ICAPRF .NE. NULL) GC TC 30

C

C ... PLAYERS SUPPRESSED

46 WRITE(99,1000) IPLAYE,DTEABR(ILOC),ILOCN
47 GO TC 100

C

C ... PRINT RUPPER

48 25 READ(99,1008) (EUF(I),I=1,9)
49 WRITE(LUNO,1008) (EUF(I),I=1,9)

C

C ... PLAYER MAKING CAPTURE

50 30 IF(IVAL(JDOER,ICAPRF) .NE. IPLABF) GO TO 35
51 IREF = IVAL(JPAE1,ICAEERF)
52 CALL PABSRP(IREF,IL,INUM,IDUM)

ISN UTILITIES - REPORT SUBROUTINE

53 WRITE(99,1001) IPLAYR,DTEABR(ILOC),ILOCN,DTPABR(ID),INUM
54 GO TO 100

C

C ... PLAYER BEING CAPTURED

55 35 IREF = IVAL(JDCER,ICAPRF)
56 CALL FARSBF(IREF,ID,INUM,IDUM)
57 WRITE(99,1002) IPLAYR,DTEABR(ILOC),ILOCN,DTPABR(ID),INUM
58 GO TO 100

C

C ... PLAYER MOVING

59 40 IREF = IVAL(JPAR1,IACTRF)
60 CALL FARSBF(IREF,ID,INUM,IDUM)
61 WRITE(99,1003) IPLAYR,DTEABR(ILOC),ILOCN,DTPABR(ID),INUM
62 IF(ICAPRF .NE. NULL) GO TO 25
63 GO TO 100

C

C ... PLAYER FIRING

64 60 IREF = IVAL(JPAR1,IACTRF)
65 CALL FARSBF(IREF,ID,INUM,IDUM)
66 WRITE(99,1004) IPLAYR,DTEABR(ILOC),ILOCN,DTPABR(ID),INUM
67 IF(ICAPRF .NE. NULL) GO TO 25
68 GO TO 100

C

C ... PLAYER OBSERVING

69 80 WRITE(99,1005) IPLAYR,DTEABR(ILOC),ILOCN
70 IF(ICAPRF .NE. NULL) GO TO 25

C

C ... SEE IF PLAYER TRIPPED A FINARY SENSOR

C

71 100 READ(99,1008) (EUF(I),I=1,9)
72 DC 110 I=1,2
73 ICN = LSERCH(IMNOBS(I),NULL,LMESS,JSUBJ,IPLARP)
74 IF(ION .NE. NULL) GO TO 130
75 110 CCNTINUE
76 GO TO 150

C

UTILITIES - REPORT SUBROUTINE

```
ISN
      C ... PRINT SENSOR TRIP MESSAGE
77      130 ISENBF = IVAL(JSOUBC,ICN)
78          ILOCRF = IVAL(JCONT,ICN)
79          CALL FABSRF(ISENBF,ID,I,INUM)
80          IREF = IVAL(JPLACE,ILOCRF)
81          CALL FABSRF(IREF,ID,INUM,INUM)
82          WRITE(99,1006) I,DTPAER(ID),INUM
83          READ(99,1008) (BUF(I),I=1,8)
84          IEUF = 8

      C
      C ... CHECK FOR PHYSICAL STATUS CHANGE
      C
85      150 I1 = IVAL(JPSTAT,IPLARF)
86          IF(I1 .EQ. KDEAD) GO TO 160
87          IIIS = IVAL(JPRCPS,IEFLARF)
88          ICN = LSERCH(IIIS,NULL,IEERCP,JID,IPLARF)
89          IF(ION .EQ. NULL) GO TO 170
90          IREF = IVAL(JVIEW,ION)
91          I2 = IVAL(JPSTAT,IEEF)
92          IF(I1 .GE. I2) GO TO 170

      C
      C ... PRINT NEW PHYSICAL STATUS
93      160 BUF(9) = STATUS(I1)
94          IEUF = 9 .

      C
      C ... PRINT BUFFER
95      170 WRITE(LUNO,1008) (BUF(I),I=1,IBUF)

      C
      C
      C ... PRINT LIST OF OBJECTS OBSERVED
96          LISCB=IOBSV(IPLAYR)
97          IF(LISCB .EQ. NULL) GO TO 200
98          CALL FABSRF(LISCB,ITYPE,IRECNO,INUM)
99          IF(ITYPE .NE. LLIST) GO TO 180
100         CALL CUTLIS(ICESV(IPLAYR),99)
101         NN = ICCUNT(IOESV(IPLAYR),INULL)
```

UTILITIES - REPORT SUBROUTINE

ISN
102 READ(99,1009) {EUF(I),I=1,NN}
103 WRITE(LUNO,1010) {EUF(I),I=1,NN}
104 GO TO 200
C
C ... OBSERVATION IS SINGLE SECOND, NOT LIST
C
105 180 WRITE(LUNO,1011) DTPAEB(IDTYP),IRECNO
C
C ... TERMINATION
106 200 RETURN
107 END

UTILITIES - VAL FUNCTION

1 FUNCTION VAL(IFLDNO,IRECREF)
C
C FUNCTION TO RETURN THE VALUE REFERRED TO BY A FIELD INDEX AND
C RECORD REFERENCE. IF ANY ERRORS ARE DETECTED IN THE ARGUMENTS
C TO THE FUNCTION, A NULL VALUE IS RETURNED INSTEAD.
C
C INPUT PARAMETERS:
C IFLDNO FIELD INDEX
C IRECREF RECORD REFERENCE
C
2 COMMON /STATEV/
3 DIMENSION ITEM(41900),DITEM(41900)
4 EQUIVALENCE (DTMIN,ITEM(1),DITEM(1))
5 COMMON /PARS/
6 EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7 REAL*8 DTPNAM,FLDNAM,FCRPT
C
8 VAL = FNULL
C BREAK RECCRD REFERENCE INTO ITS COMPONENTS
9 CALL PARSRF(IRECREF,IDLTYPE,IRECNO,IDUM)
C CHECK THAT THE FIELD INDEX SPECIFIED HAS A LEGAL VALUE
10 IF (IFLDNO .GE. 1 .AND. IFLDNO .LE. NFLDS(IDLTYPE)) GO TO 10
11 CALL ERR(4,4,IDLTYPE,IRECNO,IFLDNO)
12 RETURN
C COMPUTE THE STORAGE LOC'N OF OFFSET
13 10 IR = IETOFF(ID / (IRECNO-1)*NFLDS(IDLTYPE) + IFLDNO
C RETURN THE VALUE ST'N AT THIS LOCATION
14 VAL = DITEM(IR)
15 RETURN
16 END

UTILITIES - VAL1 FUNCTION

```
1      FUNCTION VAL1(IFLDRP)
C
C      FUNCTION TO RETURN THE VALUE REFERRED TO BY A FIELD
C      REFERENCE.
C
C      INPUT PARAMETERS:
C          IFLDBF   FIELD REFERENCE
C
2      COMMON /STATEV/
3      DIMENSION ITEM(41900),DITEM(41900)
4      EQUIVALENCE (DTPIB,ITEE(1),DITEM(1))
5      COMMON /PARS/
6      EQUIVALENCE (FNULL,NULL),(IFAIL,FAIL)
7      REAL*8 DTPNAM,FLDNAM,FCBNCT
C
C      BREAK RECORD REFERENCE INTO ITS COMPONENTS
8      CALL FABSRF(IFLDRP,ITDTYP,IRECNO,IFLDNO)
C      COMPUTE THE STORAGE LOCATION OFFSET
9      10     IR = IDTOFF(IDTYP) + (IRECNO-1)*NPLDS(IDTYP) + IFLDNO
C      RETURN THE VALUE STORED AT THIS LOCATION
10    VAL1 = DITEM(IR)
11    RETURN
12    END
```

8.0 EXAMPLE INPUT FILE LISTINGS

This chapter presents example input file listings for the Plex Pre-processor and the Data Preprocessor. The chapter is organized into two sections corresponding to those modules. The inputs to the Fixed Site Neutralization Model are largely binary, as prepared by these programs. Consequently, a display of such inputs would be uninformative and is omitted. Examples of all other input files are shown.

8.1 *Plex Preprocessor*

This subsection displays excerpts from sample input files to the Plex Preprocessor. In particular, exhibit 8-1 shows an example RECDESCRIP file; exhibit 8-2, a RECDATA file; and exhibit 8-3, an RFDATA file. These files are input from logical devices numbered 3, 4, and 5, respectively.

Fields showing the optional minimum, maximum, and default fields in file RECDESCRIP are not shown in exhibit 8-1. These are described in subsection 3.2.1. All other portions of the file are shown, including comment records. The lines listed in exhibits 8-2 and 8-3 are excerpts from a sample data file that does not purport to represent an existing site and/or scenario. Any omitted lines or records occur between parts of these exhibits.

The plex data structure is discussed at length in chapter 3, located in volume I of this manual. The RECDESCRIP file is the subject of subsection 3.2.1; RECDATA and RFDATA, subsection 3.2.2.

EXHIBIT 8-1: EXAMPLE RECODESCRIPT FILE (Part 1 of 10)

*	1	2	3	4	5	6
* 2345678901234567890123456789012345678901234567890123456789012345678901234567						
*						
*						
*						
SUMMARY OF RECORD TYPES						
*						
UTILITY						
* LIST NODE	LLIST	2				
* LOCATION	LLOCN	3				
* CHANGE	LCNG	4				
SITE DESCRIPTION						
ORGANIZATIONAL						
* SITE	LSITE	5				
* BUILDING	LBLDG	6				
* FLOOR	LFLOOR	7				
* BARRIER	LBARR	8				
REGION						
* YARD	LYARD	9				
* ROOM	LROOM	10				
* HALL	LHALL	11				
* ROOF	LROOF	12				
* STAIRS	LSTAIR	13				
PORTAL						
* DOOR	LDOOR	14				
* WINDOW	LWIND	15				
PHYSICAL OBJECTS						
* WEAPON	LWEAP	16				
* EQUIPMENT	LEQUIP	17				
* VEHICLE	LVEHIC	18				
* PERSON	LPERSN	19				
* SENSOR	LSENS	20				
* ACTIVATED DELAY	LACTD	21				
* COMNET	LCOMNT	22				
INTANGIBLE CONSTRUCTS						
* PERCEPT	LPERCP	23				
* MESSAGE	LMESS	24				
* ACTION	LACTN	25				
* RELATION	LRELN	26				
* GROUP	LGROUP	27				
* GOAL NODE	LGOAL	28				
* MODE-OF-IMPACT	LINODE	29				
* FORCE	LFORCE	30				
*						
*						
LIST NODE #						
NEXT	JNXT	1	I			
VALUE	JVAL	2	RC			

EXHIBIT 8-1: EXAMPLE RECDESCRIP FILE (Part 2 of 10)

LOCATION	LC			
SOURCE		JSOURC	1	YD, RM, HL, RF, ST, DR, WI
SINK		JSINK	2	YD, RM, HL, RF, ST, DR, WI
Frac		JFRAC	3	R
PLACE		JPLACE	4	YD, RM, HL, RF, ST, DR, WI, VE
CHANGE	CH			
NEXT		JNXT	1	I
SUBJECT		JSUBJ	2	RC
ATTRIBUTE		JATTR	3	I
NEW VALUE		JNVAL	4	-
*				
** SITE DESCRIPTION RECORDS				
*				
SITE	SI			
CONTAINED BUILDINGS		JBLDGS	1	BD
CONTAINED YARDS		JYARDS	2	YD
CONTAINED BARRIERS		JCBARS	3	BR
GUARDS		JGARDS	4	PN
ADVERSARIES		JADVRS	5	PN
VEHICLES		JVEHS	6	VE
ENVIRONMENT (XYZ)		JENVIR	7	I
LIGHTING (X)				
* DARK		KDARK	1	
* LIGHT		KLIGHT	2	
NOISE LEVEL (Y)				
* QUIET		KQUIET	1	
* MODERATE NOISE		KMODNS	2	
* NOISY		KNOISY	3	
CHEMICAL ENVIRON (Z)				
* CLEAN AIR		KCLEAN	1	
* TEAR GAS		KTGAS	2	
* NERVE GAS		KNAGAS	3	
XLEN		JKLFN	8	R
YLEN		JYLEN	9	R
ZLEN		JZLEN	10	R
BUILDING	BD			
XCOORD		JXCO	1	R
YCOORD		JYCO	2	R
ZCOORD		JZCO	3	R
CONTAINED FLOORS		JFLRS	4	FL
CONTAINED WALLS		JWALLS	5	BR
CONTAINED ROOFS		JROOFS	6	RF
ENVIRONMENT (XYZ)		JENVIR	7	I
XLEN		JKLEN	8	R
YLEN		JYLEN	9	R
ZLEN		JZLEN	10	R

EXHIBIT 8-1: EXAMPLE RECDESCRIP FILE (Part 3 of 10)

FLOOR	FL		
XCOORD	JYCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTAINED ROOMS	JROOMS	4	RM
CONTAINED HALLS	JHALLS	5	HL
CONTAINED STAIRS	JSTARS	6	ST
ENVIRONMENT (XYZ)	JENVIR	7	I
XLEN	JXLEN	8	R
YLEN	JYLEN	9	R
ZLEN	JZLEN	10	C
PENETRABILITY	JPEN	11	I
CONTAINING BUILDINGS	JCBDGs	12	BD
BARRIER	BR		
XCOORD1	JXC01	1	R
YCOORD1	JYC01	2	R
XCOORD2	JXC02	3	R
YCOORD2	JYC02	4	R
SENSORS	JSTNS	5	SN
ADJACENT BARRIERS	JBARRS	8	BR
CONTAINED PORTALS	JPORTS	9	DR,WI
HEIGHT	JHIGH	10	R
PENETRABILITY	JPEN	11	I
VISIBILITY	JVIS	12	I
*			
*	REGIONS		
*			
YARD	YD		
XCOORD	JXCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTENTS	JCONTS	4	PN,VE,WP,EQ
SENSORS	JSENS	5	SN
NEIGHBORS	JNBRS	6	YD,RM,HL,PF,ST,DR,WI
ENVIRONMENT (XYZ)	JENVIR	7	I
BARRIERS	JBARRS	8	BR
LINE OF SIGHT	JLOS	10	YD,RM,HL,RF,ST,DR,WI
COVER	JCOV	11	R
COVER, QUADRANT 1	JCOV1	12	R
COVER, QUADRANT 2	JCOV2	13	R
COVER, QUADRANT 3	JCOV3	14	R
COVER, QUADRANT 4	JCOV4	15	R

EXHIBIT 8-1: EXAMPLE RECDDESCRIP FILE (Part 4 of 10)

ROOM	RM		
XCOORD	JXCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTENTS	JCONTS	4	PN, VE, WP, EQ, AD
SENSORS	JSENS	5	SN
NEIGHBORS	JNBRs	6	YD, RM, HL, RF, ST, DR, WI
ENVIRONMENT (XYZ)	JENVIR	7	I
XLEN	JXLEN	8	R
YLEN	JYLEN	9	R
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI
COVER	JCOV	11	R
ROOM ACCESS	JRMACC	12	I
CONTAINING FLOOR	JFLOOR	13	FL
HALL	HL		
XCOORD	JXCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTENTS	JCONTS	4	PN, VE, WP, EQ, AD
SENSORS	JSENS	5	SN
NEIGHBORS	JNBRs	6	DR, WI, ST
ENVIRONMENT (XYZ)	JENVIR	7	I
XLEN	JXLEN	8	R
YLEN	JYLEN	9	R
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI
COVER	JCOV	11	R
ROOM ACCESS	JRMACC	12	I
CONTAINING FLOOR	JFLOOR	13	FL
ROOF	RF		
XCOORD	JXCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTENTS	JCONTS	4	PN, VE, WP, EQ
SENSORS	JSENS	5	SN
NEIGHBORS	JNBRs	6	DR, WI, ST, YD, RF
ENVIRONMENT (XYZ)	JENVIR	7	I
XLEN	JXLEN	8	R
YLEN	JYLEN	9	R
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI
PENETRABILITY	JPER	11	I

EXHIBIT 8-1: EXAMPLE RECODESRI FILE (Part 5 of 10)

*				
STAIRS	ST			
XCOORD	JXCO	1	R	
YCOORD	JYCO	2	R	
ZCOORD	JZCO	3	R	
CONTENTS	JCONTS	4	PN, VE, NP, EQ	
SENSORS	JSENS	5	SN	
NEIGHBORS	JNBRS	6	DR, WI, ST, RM, HL, YD, RF	
ENVIRONMENT (XYZ)	JENVIR	7	I	
STAIRS	JSTAIR	8	ST	
INCIDENT FLOOR	JINFLR	9	FL	
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI	
*				
*** PORTALS				
*				
DOOR	DR			
KCOORD	JKCO	1	R	
YCOORD	JYCO	2	R	
ZCOORD	JZCO	3	R	
CONTENTS	JCONTS	4	PN, VE, NP, EQ	
SENSORS	JSENS	5	SN	
NEIGHBORS	JNBRS	6	YD, RM, HL, RF, ST, DR, WI	
ENVIRONMENT (XYZ)	JENVIR	7	I	
STATUS	JSTAT	8	I	
DESTROYED	KDESTR	1		
OPEN	KOPEN	2		
CLOSED	KCLOSED	3		
LOCKED	KLOCKD	4		
SECURED	KSECUR	5		
LOCKABILITY	JLOCK	9	I	
NOT LOCKABLE	KNLOCK	1		
LOCKABLE WITH KEY	KKEYLK	2		
LOCKABLE WITHOUT KEY	KNKYLK	3		
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI	
PENETRABILITY	JPEN	11	I	
FREE AIR	KAIR	1		
1 CM GLASS	KGLASS	2		
.5 CM WIRE	KWIRE	3		
3 CM WOOD	KWOOD	4		
2 CM STEEL BARS	KBARS	5		
10 CM INSIDE WALL	KWALL	6		
16 CM MASONRY	KBRICK	7		
4 CM STEEL	KSTEFL	8		
4 CM STEEL + BRICK	KLAMTN	9		
VISIBILITY	JVIS	12	I	
PERMANENTLY OPAQUE	KPOPAQ	1		
TEMPORARILY OPAQUE	KTOPAQ	2		
TRANSPARENT	KTRAMP	3		
DOOR ACCESS	JDRACC	13	I	
HORIZONTAL	JHORIZ	14	R	
VERTICAL	JVERT	15	R	
PORTAL	JPORT	16	DR	
BARRIER CONTAINED IN	JBARIN	17	BR	

EXHIBIT 8-1: EXAMPLE RECDDESCRIP FILE (Part 6 of 10)

WINDOW	WI		
XCOORD	JXCO	1	R
YCOORD	JYCO	2	R
ZCOORD	JZCO	3	R
CONTENTS	JCONTS	4	PM, VE, UP, EQ
SENSORS	JSENS	5	SV
NEIGHBORS	JNBRS	6	YD, RM, HL, RF, ST, DR, WI
ENVIRONMENT (XYZ)	JENVIR	7	I
STATUS	JSTAT	8	I
LOCKABILITY	JLOCK	9	I
LINE OF SIGHT	JLOS	10	YD, RM, HL, RF, ST, DR, WI
PENETRABILITY	J PEN	11	I
VISIBILITY	JVIS	12	I
DOOR ACCESS	JDRACC	13	I
HORIZONTAL	JHORIZ	14	R
VERTICAL	JVERT	15	R
PORTAL	J PORT	16	DR
*			
*	PHYSICAL OBJECTS		
*			
WEAPON	WP		
TYPE	JTYPE	1	I
PISTOL	KPISTL	1	
RIFLE	KRIFLE	2	
MACHINE GUN	KMCGUN	3	
LAW	KLAW	4	
GRENADE	KGPWNAD	5	
TEAR GAS	KTRGAS	6	
NERVE GAS	KNVGAS	7	
AMMO OR EXPLOSIVE AMOUNT	JAMT	2	I
EQUIPMENT	EQ		
TYPE	JTYPE	1	I
KEY	KKEY	1	
SCALING EQUIP	KSCALR	2	
SNM	KSNM	3	
BULLET PROOF VEST	KVEST	4	
GAS MASK	KGMASK	5	
PARAMETER	JPAR	2	I

EXHIBIT 8-1: EXAMPLE RECDDESCRIP FILE (Part 7 of 10)

VEHICLE	VE		
TYPE	JTYPE	1	I
CAR	KCAR	1	
TRUCK	KTRUCK	2	
APC	KAPC	3	
HELICOPTER	KHEL	4	
PHYSICAL STATUS	JPSTAT	2	I
DESTROYED	KDESTR	1	
IMMOBILIZED	KIMMOB	2	
OPERATIONAL	KOPERL	3	
LOCATION	JLOCN	3	LC
CONTENTS	JCONTS	4	PN, WP, EQ
OWNER	JOWNER	5	I
SECURITY FORCE	KSFF	1	
ADVERSARY	KAFE	2	
VELOCITY	JVEL	6	R
DRIVER	JDRIVR	7	PN
PERSON	PN		
TYPE	JTYPE	1	I
WORKER	KWORKR	1	
SENSOR MONITOR	KSNMON	2	
COMBATANT	KCOMBT	3	
PHYSICAL STATUS	JPSTAT	2	I
DEAD	KDEAD	1	
CAPTURED	KCAPTR	2	
WOUNDED	KWCUND	3	
WHOLE	KWHOLE	4	
LOCATION	JLOCN	3	LC
ALLEGIANCE	JALLEG	4	I
SECURITY FORCE ELEMENT	KSFE	1	
ADVERSARY	KAFE	2	
CREDENTIALS	JCRED	5	I
A FE WITH FORGED CRED	KCRED1	1	
SFE WORKER OR GUARD	KCRED2	2	
ACCESS TO RESTRICTED AREAS	KCRED3	3	
SITUATION	JSITN	6	I
* FOR AFE:			
PRE-INTRUSION	KSITN1	1	
SELF UNDISCOVERED	KSITN2	2	
NOT YET ACTIVE ENGAG	KSITN3	3	
ENGAGEMENT	KSITN4	4	
* FOR SFE:			
NO KNOWN PROBLEM	KSITN1	1	
ALERT, UNRESOLVED	KSITN2	2	
NOT INVOLV IN ENGAG	KSITN3	3	
ENGAGED	KSITN4	4	
SUPPRESSION	JSUPPN	7	R
POSTURE	JPOSTR	8	R
WEAPONS & EQUIPMENT	JWEAPS	9	WP, EQ
LEADER	JLEADR	10	PN

EXHIBIT 8-1: EXAMPLE RECDESCRIP FILE (Part 8 of 10)

SUBORDINATES	JSUBS	11	PN
COMMO NETS	JCMNTS	12	CN
PERCEPTIONS	JPRCPS	13	PC
ACTIVITY TIME REQUIREMENT	JTREQ	14	R
ACTIVITY	JACTIV	15	AC,RN
PLANS	JPLANS	16	AC,RN
SOPS	JSOPS	17	RN,AC
FORCE BELONG TO	JFORCE	18	FR

* (PREVIOUSLY FIELD WAS "JACSON")

*

SENSOR	SN		
TYPE OF SENSOR	JTYPE	1	I
EXPLOSIVE SENSOR	KKPLSN	1	
METAL SENSOR	KMETSН	2	
RADIOACTIVITY SENSOR	KSNNMSN	3	
PORTAL STATUS SENSOR	KPORSN	4	
PERSON, VEHICLE SENS1	KPV1SN	5	
PERSON, VEHICLE SENS2	KPV2SN	6	
CREDENTIALS SENSOR	KCRDSN	7	
PHYSICAL STATUS	JPSTAT	2	I
DESTROYED	KDESTR	1	
TRIPPED	KTRIPD	2	
OPERATIONAL	KOPERL	3	
LOCATION	JLOCN	3	LC
COVERAGE	JCOVEG	4	YD,RM,HL,RF,ST,DR,HI
IN COMMUNICATION WITH	JINCOM	5	PI
TYPE OF REPORT	JRTYPE	6	I
BINARY, FLEETING	KBINFL	1	
BINARY, RECORDING	KBINRC	2	
PICTURE, FLEETING	KPICFL	3	
PICTURE, RECORDING	KPICRC	4	
ACTIVATED DELAY AD			
TYPE	JTYPE	1	I
DOOR LOCKER	KLOCKR	1	
SENSOR ACTIVATOR	KSENAC	2	
PHYSICAL STATUS	JPSTAT	2	I
DESTROYED	KDESTR	1	
OPERATIONAL	KOPERL	3	
LOCATION	JLOCN	3	LC
COVERAGE	JCOVRG	4	DR,SN

EXHIBIT 8-1: EXAMPLE RECODESCRIPT FILE (Part 9 of 10)

```

*
COMNET          CN
TYPE
  PHONE          KPHONE   1     I
  CB RADIO       KRADIO    2
  CB RADIO2      KRADO2    3
  FREE VOICE    KVOICE    4
  VISUAL SIGNALS KSIGNL   5
  ALARM OR SIREN KALARM   6
  LOUDSPEAKER    KLSPKR    7
RECIPIENTS      JRCVRS   2     PN
MESSAGES        JMESS    3     MS
*
*
*
* INTANGIBLE CONSTRUCTS
*
PERCEP          PC
IDENTIFICATION  JID      1     PN,VE,SN,
CURRENT VIEW   JVIEWS   2     PN,VE,SN,
HISTORY         JHIST    3     MS
TIME UPDATED    JTINE    4     R
MESSAGE         MS
SOURCE          JSOURC   1     PN,SN
SUBJECT         JSUBJ    2     PN,VE,SN,
ATTRIBUTE       JATTR    3     I
TIME            JTIME    4     R
CONTENT         JCONT    5     -
ACTION          AC
TYPE            JTYPE    1     I
PAR1            JPARI    2     -
PAR2            JPARE2   3     -
PAR3            JPARE3   4     -
MODIFIERS      JMODS    5     PN
DOER            JDOER    6     PN,VE,SN,EF
RELATION        RU
TYPE            JTYPE    1     I
PAR1            JPARI    2     -
PAR2            JPARE2   3     -
*
GROUP           GP
TYPE            JTYPE    1     I
NUMBER          JNUM     2     I
ELEMENTS        JELEMS   3     RC
PROPERTIES      JPROPS   4     RC

```

EXHIBIT 8-1: EXAMPLE RECDESCRIP FILE (Part 10 of 10)

*	GOAL NODE	GL		
	IDENTIFICATION	JID	1	RM,YD,RF,HL
	SUCCESSORS	JSUCRS	2	GL
	TIME OF CONTROL REQT	JTIMRQ	3	R
	TIME OF CONTROL SO FAR	JTIM	4	R
	EQUIPMENT REQUIREMENTS	JEQPRQ	5	EQ
	ADVERSARY REQUIREMENTS	JADVRQ	6	I
*	NODE-OF-IMPACT	NI		
	NODE ID	JID	1	RM,HL,YD,RF,ST,DR,WI
	PARAMETER 1	JPAR1	2	R
	PARAMETER 2	JPAR2	3	R
	PLAYER ID AT NODE	JCONTS	4	PN,VE,EQ,WP
	NOISE GENERATED	JNSGEN	5	I
	QUIET	KQUIET	1	
	MODERATE	KMODNS	2	
	NOISY	KNOISY	3	
FORCE	FR			
	TYPE	JTYPE	1	I
	FIRST ECHELON	KECH1	1	
	SECOND ECHELON	KECH2	2	
	THIRD ECHELON	KECH3	3	
	ALLEGIANCCE	JFALFG	2	I
	LEADER	JFLDR	3	PN
	CONTENTS	JCONTS	4	PN,VE
	STATUS	JFSTAT	5	R
*	(FRACTION EFFECTIVE PERSONNEL)			
*	SITUATION	JSITN	6	I
*	FCR AFE:			
	PRE-INTRUSION	KSITN1	1	
	FORCE UNDETECTED	KSITN2	2	
*	FORCE DETECTED:			
*	* NOT YET ENGAGED	KSITN3	3	
*	* ENGAGED	KSITN4	4	
	MAIN GOAL REACHED	KSITN5	5	
*	FCR SFE:			
	NO KNOWN THREAT	KSITN1	1	
*	ALERTED, RESOLVING BY:			
	* ANOTHER FORCE	KSITN2	2	
*	* SELF	KSITN3	3	
*	AFE DETECTED:			
	* SELF UNENGAGED	KSITN4	4	
	* SELF ENGAGED	KSITN5	5	
PLANS		JFPLNS	7	AC,RN
SCPS		JFSOPS	8	AC,PN

EXHIBIT 8-2: SAMPLE EXCERPTS FROM THE RECDATA FILE (Part 1 of 3)

YARD	REFID	YARDNAME	TRNGPN	CONTENT	SENCRN	MFGRNB	ENVNTPN	SARRIER	L35	CONVFR	QUANT	QUANT2	QUANT3	QUANT4
1	190.5	159.0	0.0			YN	2			0.1	0.1	0.1	0.1	0.1
2	191.5	163.5	0.0				32			0.1	0.1	0.1	0.1	0.1
4	30.0	150.0	0.0				40			0.1	0.1	0.1	0.1	0.1
5	60.0	150.0	0.0				41			0.1	0.1	0.1	0.1	0.1
6	90.0	150.0	0.0				42			0.1	0.1	0.1	0.1	0.1
7	120.0	150.0	0.0				43			0.1	0.1	0.1	0.1	0.1
8	150.0	150.0	0.0				44			0.1	0.1	0.1	0.1	0.1
9	180.0	150.0	0.0				45			0.1	0.1	0.1	0.1	0.1
10	30.0	120.0	0.0				46			0.1	0.1	0.1	0.1	0.1
11	60.0	120.0	0.0				47			0.1	0.1	0.1	0.1	0.1
12	90.0	120.0	0.0				48			0.1	0.1	0.1	0.1	0.1
13	120.0	117.0	0.0				49			0.1	0.1	0.1	0.1	0.1
14	122.1	117.0	0.0				50			0.1	0.1	0.1	0.1	0.1
15	150.0	120.0	0.0				51			0.1	0.1	0.1	0.1	0.1
16	180.0	120.0	0.0				52			0.1	0.1	0.1	0.1	0.1
17	96.5	94.5	0.0				53			0.1	0.1	0.1	0.1	0.1
18	94.5	97.5	0.0				54			0.1	0.1	0.1	0.1	0.1
19	97.5	97.5	0.0				55			0.1	0.1	0.1	0.1	0.1
20	136.5	97.5	0.0				56			0.1	0.1	0.1	0.1	0.1
21	133.5	97.5	0.0				57			0.1	0.1	0.1	0.1	0.1
22	130.5	94.5	0.0				58			0.1	0.1	0.1	0.1	0.1
23	69.0	90.0	0.0				59			0.1	0.1	0.1	0.1	0.1
24	150.0	90.0	0.0				60			0.1	0.1	0.1	0.1	0.1
25	180.0	90.0	0.0				61			0.1	0.1	0.1	0.1	0.1
26	94.5	79.5	0.0				62			0.1	0.1	0.1	0.1	0.1
27	94.5	76.5	0.0				63			0.1	0.1	0.1	0.1	0.1
28	97.5	76.5	0.0				64			0.1	0.1	0.1	0.1	0.1
29	133.5	76.5	0.0				65			0.1	0.1	0.1	0.1	0.1
30	139.5	76.5	0.0				66			0.1	0.1	0.1	0.1	0.1
31	137.5	77.5	0.0				67			0.1	0.1	0.1	0.1	0.1
32	30.0	60.0	0.0				68			0.1	0.1	0.1	0.1	0.1
33	63.0	60.0	0.0				69			0.1	0.1	0.1	0.1	0.1
34	65.0	65.0	0.0				70			0.1	0.1	0.1	0.1	0.1
35	120.5	60.0	0.0				71			0.1	0.1	0.1	0.1	0.1
36	160.0	60.0	0.0				72			0.1	0.1	0.1	0.1	0.1
37	180.0	60.0	0.0				73			0.1	0.1	0.1	0.1	0.1
38	30.0	70.0	0.0				74			0.1	0.1	0.1	0.1	0.1
39	60.0	70.0	0.0				75			0.1	0.1	0.1	0.1	0.1
40	70.0	70.0	0.0				76			0.1	0.1	0.1	0.1	0.1
41	170.0	70.0	0.0				77			0.1	0.1	0.1	0.1	0.1
42	159.0	70.0	0.0				78			0.1	0.1	0.1	0.1	0.1
43	180.0	70.0	0.0				79			0.1	0.1	0.1	0.1	0.1
44	63.5	75.0	0.0				80			0.1	0.1	0.1	0.1	0.1
45	63.5	16.5	0.0				81			0.1	0.1	0.1	0.1	0.1
46	28.0	6.0	0.0				82			0.1	0.1	0.1	0.1	0.1
							83							

EXHIBIT 8-2: SAMPLE EXCERPTS FROM THE RECDATA FILE (Part 2 of 3)

PERSON	(I4,2X,2(I3,5X),A2,14,2X,3(I3,5X),2(F6,2,2X),5(A2,14,2X),F6,2,2X,4(A2,14,2X))														
ID	TYPE	STATUS	LOCN	ALLEG	CREDEN	SITN	SUPPRS	POSTURE	EQUIP	LEADER	SUBORD	NETS	PCPTS	TIME	
ACTION	PLANS	SOPS	SUBJ TO												
1	2	4	LC	1	1	3	1			# 101		# 109	CN	1	# 111
	AC	1													
2	3	4	LC	2	1	2	1			# 102	PN	1	CN	1	# 112
	AC	2													
3	3	4	LC	3	1	2	1			# 103	PN	1	CN	1	# 113
	AC	3													
4	3	4	LC	4	1	2	1			# 104	PN	1	CN	1	# 114
	AC	4													
5	3	4	LC	5	2		1			# 105		# 110	CN	2	# 115
	AC	5													
6	3	4	LC	6	2		1			# 106	PN	5	CN	2	# 116
	AC	6													
7	3	4	LC	7	2		1			# 107	PN	5	CN	2	# 117
	# 263														
8	3	4	LC	8	2		1			# 108	PN	5	CN	2	# 118
	# 264														

LIST NOD (14,2X,14(A2,14,2X))

ID	ELEM1	ELEM2	ELEM3	ELEM4	ELEM5	ELEM6	ELEM7	ELEM8	ELEM9	ELEM10	ELEM11	ELEM12	ELEM13	ELEM14
88	DR	3	DR	4										
89	DR	5	DR	6										
90	DR	7	DR	8										
91	DR	9	DR	10										
92	DR	11	DR	12										
93	WI	1	WI	2										
95	AD	2	EQ	9	EQ	10								
96	PH	1	AD	1	AD	3	PH	2						
97	WI	2	DR	9	DR	19								
98	DR	20	DR	21										
99	DR	11	DR	22										
101	WP	1	EQ	1										
102	WP	2	EQ	2										
103	WP	3	EQ	3	EQ	11								
104	WP	4	EQ	4	EQ	12								
105	WP	5	EQ	5										
106	WP	6	EQ	6										
107	WP	7	EQ	7										
108	WP	8	WP	9	EQ	7								
109	PH	2	PN	3	PN	4	PN	1						
110	PN	6	PN	7	PN	8	PN	5						
111	PC	1	PC	2	PC	3	PC	4						
112	PC	5	PC	6	PC	7	PC	8						
113	PC	9	PC	10	PC	11	PC	12						
114	PC	13	PC	14	PC	15	PC	16						

EXHIBIT 8-2: SAMPLE EXCERPTS FROM THE RECDATA FILE (Part 3 of 3)

* PLAYER DESCRIPTIONS AND ASSOCIATED RECORDS CONSTITUTING PLAYER PERCEPTIONS PERSON (14,2X,2(13,5X),A2,14,2X,3(13,5X),2(F6.2,2X),5(A2,14,2X),F6.2,2X,4(A2,14,2X))														
ID	TYPE	STATUS	LOCM	ALLEG	CREDEN	SITN	SUPPRS	POSTURE	EQUIP	LEADER	SUBORD	METS	PCPTS	TIME
ACTION	PLANS	SOPS	SUBJ TO											
41	2	4	LC	41	1	3	1		# 125			CH	1	
42	2	4	LC	42	1	3	1		# 126			CH	1	
43	2	4	LC	43	1	3	1		# 127			CH	1	
12	2	4	LC	12	1	3	1		# 128			CH	1	
13	3	4	LC	13	1	2	1		# 129			CH	1	
14	3	4	LC	14	1	2	1		# 130			CH	1	
15	3	4	LC	15	1	2	1		# 131			CH	1	
16	3	4	LC	16	1	2	1		# 132			CH	1	
17	2	4	LC	17	1	2	1		# 133			CH	1	
18	3	4	LC	18	1	2	1		# 134			CH	1	
19	3	4	LC	19	1	2	1		# 135			CH	1	
20	3	4	LC	20	1	2	1		# 136			CH	2	
21	3	4	LC	21	1	2	1		# 137			CH	1	
22	3	4	LC	22	1	2	1		# 138			CH	1	
23	3	4	LC	23	1	2	1		# 139			CH	1	
24	3	4	LC	24	1	2	1		# 140			CH	1	
25	3	4	LC	25	2		1		# 141			CH	2	
26	3	4	LC	26	2		1		# 142			CH	2	
27	3	4	LC	27	2		1		# 143			CH	2	
28	3	4	LC	28	2		1		# 144			CH	2	
29	3	4	LC	29	2		1		# 145			CH	2	

EXHIBIT 8-3: SAMPLE EXCERPTS FROM THE RFDATA FILE (Part 1 of 2)

SITE {13,3X,6(A2,14,2X),13,5X,3(F6.2,2X)}

ID	BLDGS	YARDS	BARRIER GUARDS	ADVER	VLCHLS	ENVIRON	XLEN	YLEN	ZLEN
2				# 232			700.0	560.0	0.0

LIST NOD {14,2X,14(A2,14,2X)}

ID	ELEM1	ELEM2	ELEM3	ELEM4	ELEM5	ELEM6	ELEM7	ELEM8	ELEM9	ELEM10	ELEM11	ELEM12	ELEM13	ELEM14
232	PN	9	PN	10	PN	11								

PERSON {14,2X,2{13,5X},A2,14,2X,3{13,5X},2(F6.2,2X),5(A2,14,2X),F6.2,2X,4(A2,14,2X)}

ID	TYPE	STATUS	LOCN	ALLEG	CREDEN	SITN	SUPPRS	POSTURE	EQUIP	LEADER	SUBORD	METS	PCPTS	TIME
ACTION	PLANS	SOPS	SUBJ TO											
9	2	4	LC	9	1		2	1		# 233		# 234	# 235	# 236
10	3	4	LC	10	1		2	1		# 237	PN	9	# 238	# 239
11	3	4	LC	11	1		2	1		# 240	PN	9	# 241	# 242

LIST NOD {14,2X,14(A2,14,2X)}

ID	ELEM1	ELEM2	ELEM3	ELEM4	ELEM5	ELEM6	ELEM7	ELEM8	ELEM9	ELEM10	ELEM11	ELEM12	ELEM13	ELEM14
233	WP	10												
234	PN	10	PN	11	PN	9								
235	CN	1	CN	3										
236	PC	33	PC	34	PC	35								
237	WP	11	WP	12										
238	CN	1	CN	3										
239	PC	36	PC	37	PC	38								
240	WF	13	WP	14										
241	CN	1	CN	3										
242	PC	39	PC	40	PC	41								

LOCATION {14,2X,2(A2,14,2X),F6.2,2X,A2,14}

ID	SOURCE	SINK	FRAC	PLACE
9	DR	1	DR	1
10	DR	1	DR	1
11	DR	1	DR	1

CONNET {14,2X,13,5X,2(A2,14,2X)}

ID	TYPE	RCVRS	MSG
3	2	# 243	

WEAPON {14,2X,2{13,5X}}

ID	TYPE	AMMO
10	1	32
11	1	32
12	2	32
13	1	32
14	2	40

EXHIBIT 8-3: SAMPLE EXCERPTS FROM THE RFDATA FILE (Part 2 of 2)

* PLAYER DESCRIPTIONS AND ASSOCIATED RECORDS CONSTITUTE A PLAYER PERCEPTION														
PERSON	TYPE	STATUS	LOCN	ALLEG	CREDEN	SETN	SUPPNS	POSTURE	EQUIP	LEADER	SHRDPN	NFTS	PCPTS	TIME
ACTION	PLANS	SOPNS	SHPNS TO											
46	?	4	LC	44	1	2	1			#	245		#	246
47	?	4	LC	45	1	2	1			#	247		#	248
48	?	4	LC	46	1	2	1			#	249		#	249
49	?	4	LC	47	1	2	1			#	251		#	252
50	3	4	LC	48	1	2	1			#	253		#	254
51	3	4	LC	49	1	2	1			#	255		#	256
52	3	4	LC	50	1	2	1			#	257		#	258
53	3	4	LC	51	1	2	1			#	259		#	260
54	3	4	LC	52	1	2	1			#	261		#	262

FORMAT ONE		FORMAT TWO		FORMAT THREE		FORMAT FOUR	
TP	SOURCE	SINK	FAC	TP	SOURCE	SINK	FAC
46	DP	I	DP	1	0.0	DP	I
45	DP	I	DP	1	0.0	DP	I
56	DP	I	DP	1	0.0	DP	I
47	DP	I	DP	1	0.0	DP	I
48	DP	I	DP	1	0.0	DP	I
49	DP	I	DP	1	0.0	DP	I
50	DP	I	DP	1	0.0	DP	I
51	DP	I	DP	1	0.0	DP	I
52	DP	I	DP	1	0.0	DP	I

```

1152 NOD (14,2X,14)A2,14,2X)
1D ELEM41 ELEM42 ELEM43 ELEM44 ELEM45 ELEM46 ELEM47 ELEM48 ELEM49 ELEM50 ELEM51 ELEM52 ELEM53 ELEM54
245 UP 10
246 CN 1 CN 3
247 UP 10
248 CN 1 CN 3
249 UP 10
250 CN 1 CN 3
251 UP 11 UP 12
252 CN 1 CN 3
253 UP 11 UP 12

```

8.2 Data Preprocessor

This subsection displays an example input file listing for the Data Preprocessor. The listing is displayed in its entirety in exhibit 8-4 on the following pages. These records do not purport to represent or partially represent any existing site and/or scenario. Nor are there validity claims for any of the values in the performance data.

The reader may wish to scan exhibit 8-4 while opening to chapter 5 in volume I of this manual. That chapter presents variable definitions and cross-references.

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 1 of 6)

ARTRAT	25	3.25	1.0	{10F9.0}	19.00	0.50	4.00	8.00	30.00
0.62									
3.00	2.00	5.00	5.00	{10F8.0}	3.0	3.0	30.0	63.0	
APPFA	25								
0.015	0.1	0.3	0.09	0.01	0.01	0.01	0.001	0.1	0.5
0.25	0.06	7.0	10.0	12.0	15.0	1.0	0.05	0.35	0.05
0.05	0.05	0.05	0.05	0.05	14	{5F8.0}			
AACTION	7	2	5						
24.0	60.0	100.0	2.0	4.0					
COMPLA	7				{10F9.0}				
<0	0.1	0.2	0.05	0.05		3.0	8.0		
DECFLG2	2	7	2	{5F9.0}					
0.7	0.5	0.3	0.0	0.5					
0.0	0.0	0.0	0.0	0.0					
DETANC	6				{10F9.0}				
0.247	0.211	0.125	0.039	0.003	0.039	0.125	0.211		
DETPH1	0.3217E								
DETPH2	3				{3F9.0}				
DETSEN	17	7	7						
0.0	3.0	0.0	0.0	0.0	{11F5.0}				
0.5	0.5	0.5	0.5	0.5		0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0		0.5	0.5	0.5	0.5
0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	0.0
0.1	0.1	0.1	0.1	0.1		0.1	0.1	0.1	0.1
0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0		0.0	0.0	0.0	0.0
RNGCF	7	7	7	{9F8.0}					
0.5	0.5	0.5	0.5	0.5		0.5	0.5	0.5	0.5
0.0	0.2	0.4	0.2	0.8	{5F8.0}				
NYCWF0	=	a							
0.3	0.3	0.3	0.3	0.5					
0.7	0.7	0.7	0.7	0.7					
0.8	0.9	0.9	0.9	0.7					
0.2	0.3	0.3	0.3	0.2					
0.2	0.2	0.2	0.2	0.6					
0.9				0.0					
0.0				0.0					

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 2 of 6)

```

0.1    0.2    0.2    0.1
DTPLAN      (F9.0)
1.0
DTSEC      (F9.0)
10.0
EVSEN      40      (10F8.0)
-50.0   -70.0   10.0   -50.0   -50.0   0.0   0.0   0.0   0.0   0.0
70.0    40.0    40.0   60.0    0.0    40.0   0.0   3.0   0.0   0.0
60.0    50.0    50.0   40.0    30.0   10.0   0.0   20.0   0.0   100.0
0.0     10.0    0.0    0.0    0.0    0.0    0.0   0.0   0.0   0.0
10.0    10.0    10.0   10.0    10.0   10.0   10.0   10.0   10.0   10.0
30.0    20.0   100.0   0.0    0.0    0.0    0.0   0.0   0.0   0.0
HCLINc  5 2      (5F8.0)
P.0     20.0    4.0    1.0    25.0
P.0     0.0    4.0    1.0    25.0
TACTTP  25      10      (10(13.5X1))
1       1       1       1       1       1       2       2       2       2
2       2       2       2       2       2       2       2       2       2
TARHTP  5       (10(13.5X1))
2       2       1       2
TCOMPO  ?      = 25 *      (10(13.5X1))
TFPTD  ?       5      (5(13.5X1))
1       1       1       1       2
TRNKWD  7       5 7      (5(13.5X1))
0       0       2       1       0
3       2       1       4       5
2       1       3       0       4
2       1       3       0       0
1       2       0       0       3
1       2       0       7       4
0       0       0       1       0
TSCAR   2=      10      (10(13.5X1))
2       2       2       2       2       1       2       2       2       1
TSCRN  1. 2      (4(13.5X1))
2       1       1       1
2       2       1       1
TSPC   25      (10A8)
DTDTL  PTYPE  M-AO  LAW  GRENADE NOTUSED NOTUSED KEY  ROPES  S44
VECT  MACK  CCRN  TRUCK APC  UH-1 PERSON SENSOR1 SENSOR2 SFNSD93
CCNDR  TV  M-HAWC  SENSORS LOCKER

```

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 3 of 6)

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 4 of 6)

0.02	0.05	0.10	0.20	0.02	0.75	0.15	0.45
0.02	0.05	0.10	0.30	0.02	0.75	0.15	0.45
0.02	0.05	0.10	0.30	0.02	0.75	0.15	0.45
0.02	0.05	0.10	0.30	0.02	0.75	0.15	0.45
0.02	0.05	0.10	0.30	0.02	0.75	0.15	0.45
0.02	0.05	0.10	0.30	0.02	0.75	0.15	0.45
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
0.02	0.15	0.20	1.0	0.02	0.20	0.45	1.5
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	1.2	1.5	1.0	1.5	1.8	2.0	2.6
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
1.0	2.0	4.0	7.0	1.0	2.0	5.0	9.0
PWDFIND	-	-	5	(7FA.0)			
0.01	0.15	0.20	0.05	0.05	0.95	0.10	
0.02	0.20	0.25	0.05	0.05	0.95	0.15	
0.10	0.25	0.25	0.05	0.05	0.95	0.15	

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 5 of 6)

0.-5	0.-5	0.-5	0.-5	0.-5	0.-5	0.-5
0.-10	0.-10	0.-10	0.-10	0.-10	0.-10	0.-10
PFSFTO						
0.-2	0.-2	0.-2	0.-2	0.-2	0.-2	0.-2
0.-3	0.-3	0.-3	0.-3	0.-3	0.-3	0.-3
PCSWHY						
0.-5	0.-5	0.-5	0.-5	0.-5	0.-5	0.-5
0.-5	0.-5	0.-5	0.-5	0.-5	0.-5	0.-5
RCSDRAC						
0.-2	0.-2	0.-2	0.-2	0.-2	0.-2	0.-2
PFSPLH						
-0.-2	-0.-2	-0.-2	-0.-2	-0.-2	-0.-2	-0.-2
RSSTP						
0.-1	0.-1	0.-1	0.-1	0.-1	0.-1	0.-1
RSORT4						
20.-0	20.-0	20.-0	20.-0	20.-0	20.-0	20.-0
RFQAN						
0.-2	0.-2	0.-2	0.-2	0.-2	0.-2	0.-2
SICK						
0.-5	0.-5	0.-5	0.-5	0.-5	0.-5	0.-5
0.-6	0.-6	0.-6	0.-6	0.-6	0.-6	0.-6
ENCLRY						
0.-7	0.-7	0.-7	0.-7	0.-7	0.-7	0.-7
PNCPAX						
25.-0	25.-0	25.-0	25.-0	25.-0	25.-0	25.-0
FVCPIN						
0.-0	0.-0	0.-0	0.-0	0.-0	0.-0	0.-0

EXHIBIT 8-4: SAMPLE INPUT FILE FOR THE DATA PREPROCESSOR (Part 6 of 6)

SENTRIG 7
0.5 0.5 0.5 0.1 50.0 5.0 0.1
SKILL 4 2 50 4 2 11 {8E8.0}
1.0 1.0 1.0 1.0 0.8 1.2 1.2 0.0
1.0 1.0 1.0 1.0 0.9 1.1 1.1 0.9
1.0 1.0 1.0 0.8 0.7 1.3 1.3 0.9
1.0 1.0 1.2 1.0 0.9 1.2 1.5 0.5
1.0 1.0 1.0 1.0 0.9 1.2 1.2 0.9
1.0 1.0 1.0 1.0 0.9 1.1 1.1 0.8
1.0 1.0 1.0 0.8 0.7 1.3 1.3 0.9
1.0 1.0 1.2 1.0 0.9 1.2 1.5 0.5
1.0 1.0 1.0 1.0 0.9 1.2 1.2 0.9
1.0 1.0 1.0 1.0 0.9 1.2 1.2 0.9
1.0 1.0 1.0 0.8 0.7 1.3 1.3 0.9
SUPER 2 {2E8.0}
0.2 0.25
SUPER10 2 {2E8.0}
0.4 0.4
SUPERMV 2 {2E8.0}
0.4 0.4
SUPERF 2 {2E8.0}
1.0 1.0 {E8.0}
TETN 5.0 {E8.0}
TSTATC {E8.0}
10.0
WINTH 5 {5E8.0}
2.0 2.1 2.5 20.0 0.25
ENDRJ K
ENDFTI

9.0 EXAMPLE OUTPUT FILE LISTINGS

This chapter presents example output file listings for the Plex Pre-processor, the Data Preprocessor, and the Fixed Site Neutralization Model. The chapter is organized into three sections corresponding to these respective programs. Some output files from the preprocessors are binary -- i.e., consisting entirely of zeros and ones. The displays of such outputs are uninformative and are, therefore, omitted. All other output files are shown.

9.1 *Plex Preprocessor*

This section displays example output file listings generated by the Plex Preprocessor. There are four output files, located on logical devices numbered 6 through 9. The binary file on logical device numbered 6 is the principal output, containing the plex data structure input to the FSNM: it is not displayed due to its binary form. Error messages are on logical device number 7; excerpts from such a file are displayed in exhibit 9-1. The alphanumeric form of the plex data structure is on logical device number 8; excerpts from such a file are displayed in exhibit 9-2. Maps of the site and its buildings are on logical device number 9; excerpts from such a file are displayed in exhibit 9-3. The excerpts do not purport to represent an existing site and/or scenario, and are for illustrative purposes only.

EXHIBIT 9-1: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 7 (Part 1 of 2)

***** ERROR 9 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
A RECORD INDEX USED TO REFER TO DATATYPE: ROOF
IS A DUPLICATE. THE INDEX IS: 6 RECORD IGNORED

***** ERROR 14 IN ROUTINE IVALRF
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
LIST 125 REFERENCED IN RECORD 1060130001 AFTER BEING DEFINED
REFERENCE ALLOWED, BUT MAY BE IN ERROR

***** ERROR 14 IN ROUTINE IVALRF
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
LIST 126 REFERENCED IN RECORD 1060110001 AFTER BEING DEFINED
REFERENCE ALLOWED, BUT MAY BE IN ERROR

***** ERROR 14 IN ROUTINE IVALRF
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
LIST 127 REFERENCED IN RECORD 1060110002 AFTER BEING DEFINED
REFERENCE ALLOWED, BUT MAY BE IN ERROR

***** ERROR 15 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE LIST NOD RECORD 2 REFERENCED BUT NOT DEFINED

***** ERROR 15 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE LIST NOD RECORD 4 REFERENCED BUT NOT DEFINED

***** ERROR 15 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE LIST NOD RECORD 5 REFERENCED BUT NOT DEFINED

***** ERROR 15 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE LIST NOD RECORD 103 REFERENCED BUT NOT DEFINED

EXHIBIT 9-1: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL
DEVICE NUMBER 7 (Part 2 of 2)

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 61 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 62 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 63 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 64 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 65 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 66 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 67 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 68 SKIPPED ON INPUT

***** ERROR 16 IN ROUTINE INPLEX
TIME= 0.0 SIDE=SFE LAST REFERENCED PLAYER= 1
DATATYPE DOOR RECORD 69 SKIPPED ON INPUT

EXHIBIT 9-2: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 8
 (Part 1 of 3)

```

>SITE
>    BLDGS    YARDS    BARRIER GUARDS    ADVER    VEHICLE    ENVIRON XLEN    YLEN    ZLEN
>    1    8    1            8    3                                210.00    168.00    0.0

>LIST NOD
>    1    BB    1    BD    2    BD    3    BD    4    BD    5    BD    6    BR    7    BR    8    BR    9    BR    10    BR    11    BR    12    BR    13    BR    14
>    3    BE    1    BR    2    BR    3    BR    4    BR    5    BR    6    BR    7    BR    8    BR    9    BR    10    BR    11    BR    12    BR    13    BR    14
>    3    BR    15    BR    16    BR    17    BR    18    BR    19    BR    20    BR    21    BR    22    BR    23    BR    24    BR    25    BR    26    BR    27    BR    28
>    3    BR    29    BR    30    BR    31    BR    32    BR    33

>BUILDING
>    XCOORD    YCOORD    ZCOORD    FLOORS    WALLS    ROOFS    ENVIRON XLEN    YLEN    ZLEN
>    1    110.00    88.00    0.0    6    4    7    RF    1                                24.00    8.00    8.00
>    2    114.00    76.00    0.0    FL    3    8    8    RF    2                                44.00    8.00    4.00
>    3    138.00    74.00    0.0    FL    4    8    9    RF    3                                4.00    -4.00    4.00
>    4    113.00    82.00    0.0    FL    5    8    10    RF    4                                2.00    4.00    4.00
>    5    193.00    92.00    0.0    FL    6    8    11    RF    5                                6.00    8.00    4.00
>    6    105.00    83.00    0.0    FL    7    8    12    RF    6                                4.00    2.00    4.00

>LIST NOD
>    6    FL    1    FL    2
>    7    BR    5    BR    6    BR    7    BR    8
>    8    BR    9    BR    10    BR    11    BR    12
>    9    BR    13    BR    14    BR    15
>    10    BR    33    BR    34
>    11    BR    17    BR    18    BR    19    BR    20
>    12    BR    30    BR    31    BR    32

>FLOOR
>    XCOORD    YCOORD    ZCOORD    ROOMS    HALLS    STAIRS    ENVIRON XLEN    YLEN    ZLEN    PENET    CONT    IN
>    1    112.00    88.00    0.0    1    474    ST    1                                24.00    8.00    4.00    800    BB    1
>    2    112.00    88.00    4.00    RH    3                                ST    2                                12.00    8.00    4.00    800    BB    1
>    3    114.00    76.00    0.0                                475                                44.00    8.00    4.00    800    BB    2
>    4    138.00    74.00    0.0    RH    8                                4.00    4.00    4.00    800    BB    3
>    5    113.00    82.00    0.0    RH    6                                2.00    4.00    4.00    800    BB    4
>    6    105.00    83.00    0.0    RH    5                                4.00    2.00    4.00    800    BB    6
>    7    193.00    92.00    0.0    1    476                                6.00    8.00    4.00    800    BB    5

>LS4 NOD
>    474    RH    1    RH    2    RH    4
>    475    RH    1    RH    2
>    476    RH    9    RH    10

```

EXHIBIT 9-2: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 8
 (Part 2 of 3)

>YARD	XCOORD	YCOORD	ZCOORD	CONTENT	SENSOR	WEIGHT	ENFORCER LOS	COVER	OBAB1	OBAB2	OBAB3	OBAB4
5	60.00	150.00	0.0		223			0.10	0.10	0.10	0.10	0.10
6	90.00	150.00	0.0		224			0.10	0.10	0.10	0.10	0.10
7	120.00	150.00	0.0		225			0.10	0.10	0.10	0.10	0.10
8	150.00	150.00	0.0		226			0.10	0.10	0.10	0.10	0.10
9	180.00	150.00	0.0		227			0.10	0.10	0.10	0.10	0.10
10	30.00	120.00	0.0		228			0.10	0.10	0.10	0.10	0.10
11	60.00	120.00	0.0		229			0.10	0.10	0.10	0.10	0.10
12	90.00	120.00	0.0		230			0.10	0.10	0.10	0.10	0.10
13	120.00	120.00	0.0		231			0.10	0.10	0.10	0.10	0.10
14	150.00	120.00	0.0		232			0.10	0.10	0.10	0.10	0.10
15	180.00	120.00	0.0		233			0.10	0.10	0.10	0.10	0.10
16	210.00	120.00	0.0		234			0.10	0.10	0.10	0.10	0.10
17	95.50	96.50	0.0		250		254	0.10	0.10	0.10	0.10	0.10
21	126.50	96.50	0.0		252		253	0.10	0.10	0.10	0.10	0.10
23	6.00	90.00	0.0		235			0.10	0.10	0.10	0.10	0.10
24	150.00	90.00	0.0		236			0.10	0.10	0.10	0.10	0.10
25	186.00	90.00	0.0		237			0.10	0.10	0.10	0.10	0.10
26	91.50	80.50	0.0		238			0.10	0.10	0.10	0.10	0.10
27	91.50	71.50	0.0		255		257	0.10	1.00	0.10	0.10	0.10
29	135.50	80.50	0.0		262		263	0.10	0.10	0.10	0.10	0.10
30	140.50	76.50	0.0		260	-	261	0.10	0.10	0.10	1.00	0.10
31	140.50	71.50	0.0		260		259	0.10	0.10	1.00	0.10	0.10
32	30.00	60.00	0.0		248			0.10	0.10	0.10	0.10	0.10

EXHIBIT 9-2: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 8
(Part 3 of 3)

HALL	XCOORD	YCOORD	ZCOORD	CONTENT	SENSOR	NEIGHB	ENVIRON	XLEN	YLEN	LOS	COVER	ACCESS	CONT	IN
1	113.00	76.00	0.0			485		42.00	2.00		0.0		FL	3
2	135.00	76.00	0.0			490		2.00	8.00		0.0		FL	3

> 485 BR 24 BR 21 BK 18 BK 26 HL 2
 > 490 HL 1 BR 15 BK 12

ROOF	XCOORD	YCOORD	ZCOORD	CONTENT	SENSOR	NEIGHB	ENVIRON	XLEN	YLEN	LOS	PENT
1	110.00	88.00	8.00					24.00	8.00		8
2	114.00	76.00	4.00					44.00	8.00		8
3	138.00	74.00	4.00					4.00	4.00		8
4	113.00	82.00	4.00					2.00	4.00		8
5	193.00	92.00	4.00					6.00	8.00		8

```

> STAIRS
>      XCOORD  YCOORD  ZCOORD  CNTNTS  SENSRS  NBRS   ENVIR  STAIRS  IN FLR  LOS
>    1 111.00   84.50     0.0          6  481    ST  ?  FL  1
>    2 107.00   84.50     4.00          RM  3    ST  ?  FL  2

```

>LIST NOD
> 481 BR 27 RH 1 BK 34 BK 41

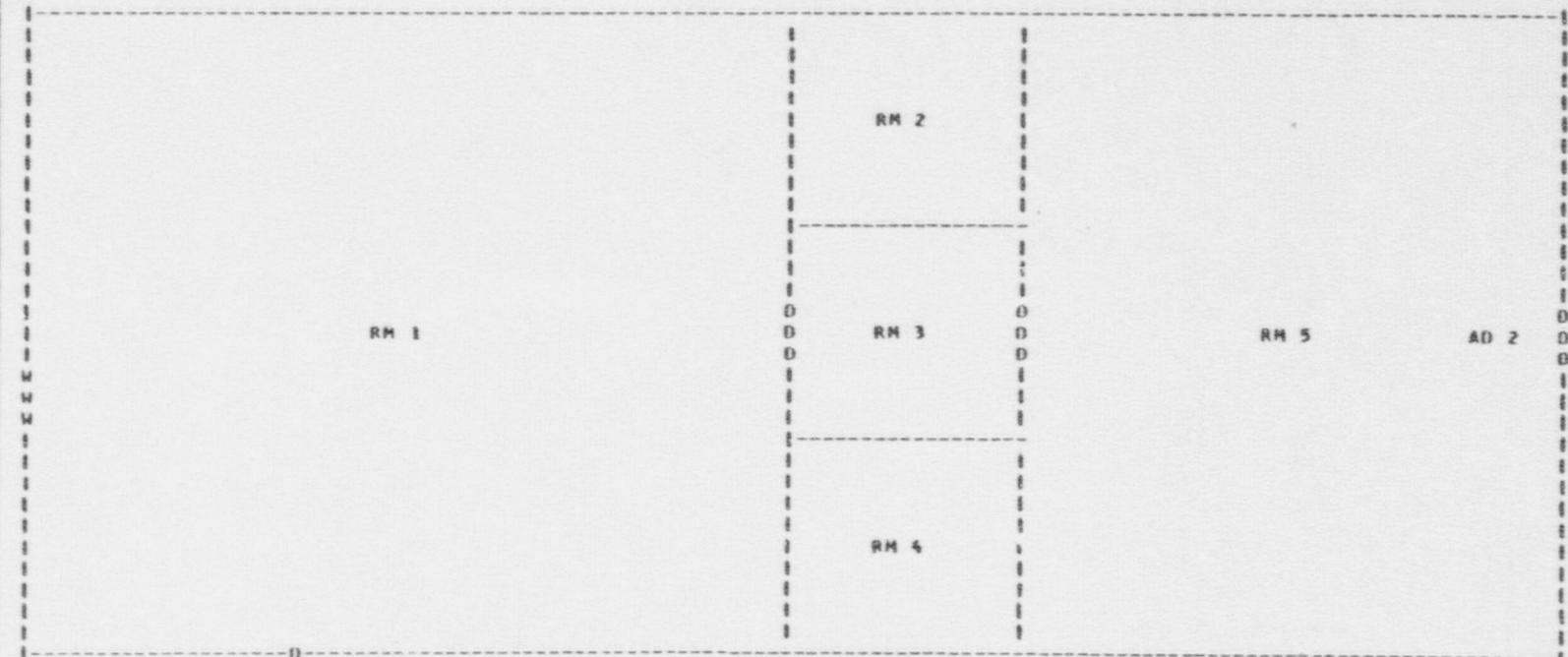
EXHIBIT 9-3: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 9 (Part 1 of 2)

PN 8
*****DD*PN PNPN 7*****
YD 6 YD 7 YD 8 YD 9 YD 10 YD 11 YD 12 YD 13 YD 14 YD 15 YD 16 YD 17 YD 18 YD 19 YD 20 YD 21 YD 22 YD 23 YD 24 YD 25 YD 26 YD 27 YD 28 YD 29 YD 30 YD 31 PN 4

EXHIBIT 9-3: EXCERPTS FROM PLEX PREPROCESSOR OUTPUTS ON LOGICAL DEVICE NUMBER 9 (Part 2 of 2)

MAP OF BUILDING 1

FLOOR 1 OF BUILDING 1



9.2 Data Preprocessor

This section displays example output file listings generated by the Data Preprocessor. There are two output files, located on logical devices numbered 6 and 7. The binary file on logical device number 6 is the principal output, containing essentially the performance data that is input to the FSNM; it is not displayed due to its binary form. An echo of all header records is produced on logical device number 7; a complete example of such a file is shown in exhibit 9-4.

EXHIBIT 9-4: EXAMPLE DATA PREPROCESSOR OUTPUT ON LOGICAL DEVICE NUMBER 7 (Part 1 of 2)

0	ACTRAT	25	0	0	19	0	0	(10F8.0)
25	AREA	25	0	0	0	0	0	{10F8.0}
50	BASLOD	7	2	0	5	1#	0	{5F8.0}
64	COMDLA	7	0	0	0	0	0	(10F8.0)
71	DEGVUL	7	2	0	5	2	0	{5F8.0}
85	DETANG	8	0	0	0	0	0	(8F8.0)
93	DETPHI	0	0	0	0	0	0	(F8.0)
94	DETRAT	3	0	0	0	0	0	(3F8.0)
97	DETSEN	17	7	0	0	0	0	(17F5.0)
216	DNGDET	9	2	0	0	0	0	(9F8.0)
234	DNGWEP	5	8	0	0	0	0	(5F8.0)
274	DTPLAN	0	0	0	0	0	0	(F8.0)
275	DTSEC	0	0	0	0	0	0	(F8.0)
276	EVALN	60	0	0	0	0	0	(10F8.0)
336	HFLIMS	5	2	0	0	0	0	(5F8.0)
346	IACTTP	25	0	0	19	0	0	(10(I3,5X))
371	IARMTP	4	0	0	0	0	0	(10(I3,5X))
375	ICOMRQ	2	5	25	*	0	0	(10(I3,5X))
625	IFIRTP	7	0	0	5	0	0	(5(I3,5X))
632	IRNKWP	7	7	0	5	7	0	(5(I3,5X))
681	ISECAC	25	0	0	19	0	0	(10(I3,5X))
706	ISECRQ	4	2	0	0	0	0	(4(I3,5X))
714	LABEL	25	0	0	0	0	0	(10A8)
739	MINPLR	2	0	0	0	0	0	(2(I3,5X))
741	MPEN	5	0	0	0	0	0	(5(I3,5X))
746	NDTSEC	25	0	0	19	0	0	(10(I3,5X))
771	PK	7	7	0	7	5	0	(7F8.0)
820	POBATT	31	0	0	0	0	0	(7F8.0,/,2(9F8.0,/),6F8.0)

EXHIBIT 9-4: EXAMPLE DATA PREPROCESSOR OUTPUT ON LOGICAL DEVICE NUMBER 7 (Part 2 of 2)

851	POBSPN	.2	2	7	2	2	5	(4F8.0)
879	POBSEN	2	0	0	0	0	0	(2F8.0)
881	POBVEH	2	2	0	0	0	0	(4F8.0)
885	POSTUR	25	0	0	19	0	0	(10F8.0)
910	PSIG	8	7	7	8	7	5	(8F8.0)
1302	PWOUND	7	7	0	7	5	0	(7F8.0)
1351	RESFIR	50	0	0	11	0	0	(10F8.0)
1401	RESMOV	50	0	0	11	0	0	(10F8.0)
1451	RESOBS	50	0	0	11	0	0	(10F8.0)
1501	RESPLN	50	0	0	1#	0	0	(F8.0)
1551	RESSUR	50	0	0	11	0	0	(10F8.0)
1601	RESPTM	0	0	0	0	0	0	(F8.0)
1602	RESWND	50	0	0	1#	0	0	(F8.0)
1652	RISK	4	2	0	0	0	0	(4F8.0)
1660	RNGLET	7	0	0	0	0	0	(10F8.0)
1667	RNGMAX	7	0	0	0	0	0	(7F8.0)
1674	RNGMIN	7	0	0	0	0	0	(7F8.0)
1681	SENRNG	7	0	0	0	0	0	(7F8.0)
1688	SKILL	4	2	50	4	2	11	(8F8.0)
2088	SUPDEC	2	0	0	0	0	0	(2F8.0)
2090	SUPFIR	2	0	0	0	0	0	(2F8.0)
2092	SUPMOV	2	0	0	0	0	0	(2F8.0)
2094	SUPOBS	2	0	0	0	0	0	(2F8.0)
2096	TFIN	0	0	0	0	0	0	(F8.0)
2097	TSTALE	0	0	0	0	0	0	(F8.0)
2098	WIDTH	5	0	0	0	0	0	(5F8.0)
2103	ENDBLK	0	0	0	0	0	0	
0	ENDFIL	0	0	0	0	0	0	

9.3 Fixed Site Neutralization Model

This section displays example output file listings generated by the Fixed Site Neutralization Model. There are three output files, located on logical devices numbered 6 through 8.

The "change file" on logical device number 6 is typically quite voluminous. It is very useful for debugging purposes, after program changes or to investigate possible errors; on production runs it may be written on a temporary file (name beginning with the character '-') for efficiency. The CHGFLD, CHGLST, and CHGVAR subroutines write in this file when values in the plex structure change, when the compositions of lists change, or when variables stored in the data arrays change. Excerpts from the change file are shown in exhibit 9-5.

The "report file" is on logical device numbered 7. At the end of every time interval--whose duration is under user control,--it records the locations, activities, and observations of all people on the site. Any error messages are also written to this file. Also, a summary is printed there every time the garbage-collecting subroutine is invoked; for efficiency on subsequent runs, the programmer may wish to reallocate in the BLOCKDATA subroutine so that such calls are minimized. Excerpts from the report file are shown in exhibit 9-6.

The "final file" is on logical device number 8. It records the final contents of the plex data structure. Excerpts from the final file are shown in exhibit 9-7.

All excerpts shown are illustrative only; they do not purport to be consistent with any existing site and/or scenario.

EXHIBIT 9-5: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER
6 (Part 1 of 3)

*** INITIALIZATIONS

TMIN	=	0.0
NSTAT	=	1
FORCES(1, 1)	=	0.0
FORCES(2, 1)	=	0.0
FORCES(3, 1)	=	0.0
FORCES(4, 1)	=	4.00
FORCES(1, 2)	=	0.0
FORCES(2, 2)	=	0.0
FORCES(3, 2)	=	0.0
FORCES(4, 2)	=	4.00
IGOALS	=	-9999

RECORD GL 1 QUEUED ON LIST IGOALS

*** LEADER PLANNING

FIELD FRAC	OF FR	1	CHANGED FROM	1.0	TO	1.0
FIELD IDENT	OF PC	42	CHANGED FROM		TO FR	1
FIELD VIEW	OF PC	42	CHANGED FROM		TO FR	4
FIELD CONTENT	OF FR	4	CHANGED FROM		TO PN	1
FIELD CONTENT	OF FR	4	CHANGED FROM PN	1	TO *	645
FIELD CONTENT	OF FR	4	CHANGED FROM *	645	TO *	646
FIELD CONTENT	OF FR	4	CHANGED FROM *	646	TO *	647
FIELD FRAC	OF FR	4	CHANGED FROM	-9999.0	TO	1.0
FIELD SITN	OF FR	4	CHANGED FROM	-9999	TO	1
FIELD FRAC	OF FR	2	CHANGED FROM		1.0 TO	1.0
FIELD SITN	OF FR	2	CHANGED FROM		2 TO	1
FIELD IDENT	OF PC	43	CHANGED FROM		TO FR	2
FIELD VIEW	OF PC	43	CHANGED FROM		TO FR	5

EXHIBIT 9-5: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER
6 (Part 2 of 3)

NEW MESSAGE RECORD:

SOURCE	SUBJECT	ATTRIBUTETIME	CONTENT
87 PN 2 PN	6 2	1.33	3
RECORD MS	87 STACKED ON LIST LASMNT		
FORCES(3, 2)	=	0.0	
FORCES(1, 2)	=	1.00	
IPLAYR	=	1	
IPLAYR	=	2	
IPLAYR	=	5	
IPLAYR	=	7	
IPLAYR	=	8	

*** DETECT OBJECTS

ISIDE	=	1
IPLAYR	=	1
NWSNLT	=	-9999
IPLAYR	=	2
ISIDE	=	2
IPLAYR	=	5
IPLAYR	=	6
IPLAYR	=	7
IPLAYR	=	8

*** PROCESS OBSVNS

ISIDE	=	1			
IPLAYR	=	1			
FIELD UPDATED OF PC	45	CHANGED FROM	1.2 TO	1.3	
FIELD UPDATED OF PC	46	CHANGED FROM	1.2 TO	1.3	
FIELD UPDATED OF PC	47	CHANGED FROM	1.2 TO	1.3	
FIELD UPDATED OF PC	48	CHANGED FROM	1.2 TO	1.3	
FIELD UPDATED OF PC	49	CHANGED FROM	1.2 TO	1.3	
FIELD UPDATED OF PC	51	CHANGED FROM	1.2 TO	1.3	

NEW MESSAGE RECORD:

SOURCE	SUBJECT	ATTRIBUTETIME	CONTENT	
86 PN 1 PN	1 9	1.33	* 1093	
FIELD UPDATED OF PC	1	CHANGED FROM	1.2 TO	1.3
FIELD MSG OF CN	1	CHANGED FROM *	1131 TO *	1092
IOBSV (1)	= -9999			
IPLAYR	=	2		
FIELD UPDATED OF PC	62	CHANGED FROM	1.2 TO	1.3

NEW MESSAGE RECORD:

SOURCE	SUBJECT	ATTRIBUTETIME	CONTENT
85 PN 2 PN	2 6	1.33	4

EXHIBIT 9-5: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER
6 (Part 3 of 3)

*** NEW TIMESLICE

TMIN = 2.50

*** CHECK FOR RESPONSE FORCE ARRIVAL
RESPONSE FORCE DOES NOT ARRIVE

*** CHECK FOR END
SIMULATION CONTINUES

*** LEADER PLANNING

FIELD FRAC	OF PR	1	CHANGED FROM	0.5	TO	0.5
FIELD FRAC	OF PR	6	CHANGED FROM	0.5	TO	0.5
FIELD FRAC	OF PR	4	CHANGED FROM	1.0	TO	1.0
FIELD FRAC	OF PR	2	CHANGED FROM	0.5	TO	0.5
FIELD FRAC	OF PR	7	CHANGED FROM	0.7	TO	0.7
FIELD FRAC	OF PR	5	CHANGED FROM	1.0	TO	1.0

*** PERSON DECISIONS

ISIDE	=	1			
IPLAYR	=	1			
FIELD ACTION	OF PN	1	CHANGED FROM	TO AC	1
IPLAYR	=	2			

NEW ACTION RECORD:

TYPE	PAR1	PAR2	PAR3	MODIFIERDOER
154 3	0.25			PN 2
FIELD ACTION	OF PN	2	CHANGED FROM	TO AC 154
ISIDE	=	2		
IPLAYR	=	5		

NEW ACTION RECORD:

TYPE	PAR1	PAR2	PAR3	MODIFIERDOER
156 3	3.14			PN 5
FIELD ACTION	OF PN	5	CHANGED FROM	TO AC 156
IPLAYR	=	8		

NEW ACTION RECORD:

TYPE	PAR1	PAR2	PAR3	MODIFIERDOER
158 3	1.57			PN 8
FIELD ACTION	OF PN	8	CHANGED FROM	TO AC 158

EXHIBIT 9-6: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER 7 (Part 1 of 3)

REPORT AT TIME: 1.17

PN 1 (RM 6) OBSERVING

PN 2 (DR 19) MOVING TO RM 5

TRIPS ALARM 4 AT DR 9

SEES: PN 4

PN 3 (DR 9) MOVING TO RM 5

TRIPS ALARM 4 AT DR 9

SEES: PN 2 PN 4

PN 4 (RM 3) MOVING TO RM 5

PN 5 (DR 12) OBSERVING

SEES: PN 6

PN 6 (DR 12) OBSERVING

SEES: PN 5

PN 7 (YD 29) MOVING TO RM 5

SEES: PN 5 PN 6 PN 8

PN 8 (DR 12) MOVING TO RM 5

SEES: PN 5 PN 6

***** ERROR 32 IN ROUTINE CAPDET

TIME= 1.17 SIDE=APE LAST REFERENCED PLAYER= 8

PLAYER 1000190008 CAPTURED PLAYER 1000190004 AT LOCATION 1000100005

EXHIBIT 9-6: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER 7 (Part 2 of 3)

REPORT AT TIME: 1.67
PN 1 (RM 6) OBSERVING
PN 2 (RM 5) OBSERVING
PN 2 (RM 5) CAPTURING PN 8
SEES: PN 4 PN 5 PN 6 PN 8
PN 3 (RM 5) MOVING TO RM 5
PN 4 (RM 5) OBSERVING
PN 5 (DR 12) FIRING AT PN 4 KILLED
SEES: PN 2 PN 3 PN 4 PN 6 PN 8
PN 6 (DR 12) FIRING AT PN 3 KILLED
SEES: PN 2 PN 3 PN 4 PN 5 PN 8
PN 7 (DR 19) MOVING TO RM 5
PN 8 (RM 5) SURRENDURING TO PE 2

***** ERROR 32 IN ROUTINE CAPDET
TIME= 1.67 SIDE=AFC LAST REFERENCED PLAYER= 7
PLAYER 1000190007 CAPTURED PLAYER 1000190002 AT LOCATION 1000140022

***** ERROR 2 IN ROUTINE PARSRF
TIME= 1.67 SIDE=AFC LAST REFERENCED PLAYER= 8
A PARAMETER WHICH IS SUPPOSED TO BE A FIELD OR RECORD
REFERENCE HAD THE VALUE -9999 AND HENCE IS INCORRECT.

***** ERROR 2 IN ROUTINE PARSRF
TIME= 1.67 SIDE=AFC LAST REFERENCED PLAYER= 8
A PARAMETER WHICH IS SUPPOSED TO BE A FIELD OR RECORD
REFERENCE HAD THE VALUE -9999 AND HENCE IS INCORRECT.

EXHIBIT 9-6: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER 7 (Part 3 of 3)

REPORT AT TIME: 1.83
 PH 1 {RM 6} OBSERVING
 PH 2 {RM 5} FIRING AT PH 6
 PH 2 {RM 5} SURROUNDING TO PH 3
 SEES: PH 5 PH 6 PM 8
 PH 5 {RM 12} FIRING AT PH 2
 SEES: PH 7 PH 2 PH 6 PM 8
 PH 6 {RM 12} FIRING AT PH 2
 PH 7 {RM 5} MOVING TO RM 5
 PH 7 {RM 5} CAPTURING PH 2
 SEES: PH 5
 PH 8 {RM 5} SUPPRESSED
 SIMULATION ENDS AT TIME= 1.83
 APE WAS THE WINNER.

KILLED

KILLED

		1.83																	
		COLLECTION RESULTS:																	
RECORD TYPE:	#	LC	CH	SI	BD	FL	BB	YG	RM	BL	RP	ST	DR	HI	WP	EQ	VE		
NUMBER COLLECTED:	0	1726	441	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
TOTAL NUMBER RECORDS:	1	4000	800	200	6	8	12	30	60	40	12	10	20	120	100	600	720		
NUMBER NEVER ALLOCATED:	-3	1592	296	196	0	2	6	11	13	30	8	4	16	94	94	582	704		
RECORD TYPE:	PH	58	1B	CR	PC	BS	AC	BN	GP	GL	ME	PH							
NUMBER COLLECTED:	232	0	0	0	31	143	106	0	0	0	04	0							
TOTAL NUMBER RECORDS:	350	320	20	20	900	300	500	300	10	10	120	24							
NUMBER NEVER ALLOCATED:	65	710	11	13	823	131	356	288	6	5	28	13							

EXHIBIT 9-7: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER 8 (Part 1 of 2)

ROOM

	XCOORD	YCOORD	ZCOORD	CONTENT	SENSOR	NEIGHB	ENVIRON	XLEN	YLEN	LOS	COVER	ACCESS	CONT	IN
1	106.50	87.00	0.0			# 390		21.00	18.00		0.20		FL	1
2	120.00	93.00	0.0					6.00	6.00		0.20		FL	1
3	120.00	87.00	0.0			# 391		6.00	6.00		0.20		FL	1
4	120.00	81.00	0.0	# 535				6.00	6.00		0.20		FL	1
5	130.50	87.00	0.0	# 2361		# 393		15.00	18.00		0.20		FL	1
6	36.00	21.00	0.0	# 394		DR 13		12.00	6.00		0.20		FL	2

LIST NOD

390	WI	2	DB	9	DR	19
391	DR	20	DR	21		
535	SN	6	SN	5	SN	4
					SM	3
2361	PN	7	PN	3	PN	2
					PR	8
393	DR	11	DR	22		
394	PN	1	AD	1	AD	3

ROOF

	XCOORD	YCOORD	ZCOORD	CONTENT	SENSOR	NEIGHB	ENVIRON	XLEN	YLEN	LOS	PENT
1	117.00	87.00	3.60					42.00	18.00		5
2	36.00	21.00	3.60					12.00	6.00		5

DOOR

HEIGHT	COORD	COORD	ZCOORD	CONTENT	SENSORS	NEIGHB	ENVIRON	STATUS	LOCKBL	LOS	PENET	VISIBL	ACCESS	WIDTH
PORTAL	CONT	IP												
1	48.00	12.15	0.0			# 108		4	2		3	3	1	6.00
2	40.00	2.00	BR	1										
2	48.00	11.85	0.0			YD 46		4	2		3	3	1	6.00
2	40.00	1.00	BR	1										
3	191.85	90.00	0.0			SN 1	# 109	5	1		3	3		0.90

EXHIBIT 9-7: EXCERPTS FROM THE FSNM OUTPUTS ON LOGICAL DEVICE NUMBER 8 (Part 2 of 2)

RELATION

	TYPE	PAR1	PAR2
1	12		
2	12		
3	3		
4	3		
5	3		
6	3		
7	3		
8	12		

GOAL NOD

	REGION	SUBORD	MIN REQ	CONTROL EQUIP	MEN REQ
1	RM	5		2.00	2

NODE-OF-

	NODE ID	PAR1	PAR2	PLAYER	NOISE
85	DR 20	1.67		PN	7
86	RM 3	1.70		PN	7
87	DR 21	1.72		PN	7
88	DR 22	1.79		PN	7

FORCE

	TYPE	ALLEG	LEADER	CONTENT	PRAC	SITN	PLANS	SOPS
1	1	1	PN	1	# 477	0.50	5	
2	1	2	PN	5	# 478	1.00	4	
3	1	1	PN	9	# 525	1.00	1	
4	1	1	PN	1	# 647	1.00	4	
5	1	2	PN	5	# 652	1.00	4	
6	1	2			# 1118	1.00	3	
7	1	1			# 1947	0.67	4	

LIST NOD

	477	478	1	PN	2	PN	3	PN	4
	478	PN	5	PN	6	PN	7	PN	8

INDEX TO VOLUME II

This index includes all programs, subroutines, functions, input files, and output files covered in volume II of the program maintenance manual. The order of presentation is alphabetical, with the decimal numbers considered to be at the end of the alphabet.

ACTTY subroutine	368 to 383
ADDVAL subroutine	384 to 385
ARRIVL subroutine	386 to 388
AUTLIS subroutine	297 to 298
AUTPLX subroutine	299 to 304
BLOCK DATA	285 to 287, 787
BRECH subroutine	389 to 390
CAPDET subroutine	391 to 400
CHGFLD subroutine	643 to 647, 787
CHGLST subroutine	648 to 649, 787
CHGVAR subroutine	650 to 653, 787
CHKOUT subroutine	401 to 402
COLECT subroutine	654 to 658
COMM0 subroutine	403 to 404
COMOBS subroutine	405 to 413
COMPER subroutine	414 to 420
CONDAC subroutine	421 to 422, 659 to 660
CONTNT subroutine	423 to 424
COORDS subroutine	661
COPY subroutine	662
COVR function	425 to 427
CREACT subroutine	428 to 430
CREREL subroutine	431 to 432
Data Preprocessor - Main Program	358 to 361, 766, 781
/DATAV/	283
DEAD subroutine	433 to 435
DECIDE subroutine	436 to 459
DELIST subroutine	663 to 664
DELREC subroutine	665
DETFIR subroutine	459 to 461
DIREC subroutine	666 to 668
DIREC2 subroutine	669 to 670
DIST function	671 to 672
DRCOM1 subroutine	462
DRCOM2 subroutine	463
DRCOM3 subroutine	464 to 465

DROBS1 subroutine	466 to 468
DROBS2 subroutine	469 to 470
DROBS3 subroutine	471 to 472
DROBS4 subroutine	473 to 474
DSTAIR subroutine	305 to 306
EFTIM function	475 to 479
ERR subroutine	673 to 680
ESCAPE subroutine	480 to 481
FASMNT subroutine	482 to 484
FCNLIN function	681
FDESIR subroutine	485 to 491
FIRST function	682 to 683
Fixed Site Neutralization Model - Main Program	363 to 367, 787
FNCVAL function	684
FNEXT function	685 to 686
/GARCOL/	284
GENOBS subroutine	492 to 496
IBLMAP function	307
ICANOB function	497
ICHOOS function	498
ICMPPN function	499 to 501
ICMPSN function	502 to 503
ICMPVE function	504
ICOPY function	687 to 688
ICOUNT function	689
IFIRST function	690 to 691
IFNCVL function	692
INBLK subroutine	308, 505
INCHK subroutine	309 to 310
INDESC subroutine	311 to 315
INIT subroutine	506 to 511
INITPP subroutine	316
INITVL subroutine	693
INPLEX subroutine	317 to 329
INTLOS function	512
INTSCT function	513 to 515
INTER function	516 to 517
INTV function	518
IPATH function	519 to 523
IPLACE function	694 to 695
IQUEUE function	696 to 697
IREF function	698
IREF1 function	699
ISEEPT function	524
ISTACK function	700 to 701
ITERM function	525 to 530
IUSPTR function	330
IVAL function	702
IVAL1 function	703
IVALRF function	331 to 334

JOIN function	704 to 705
KEYS subroutine	531 to 533
KONTNT subroutine	335 to 336
LABL subroutine	337 to 339
LCOPY function	706
LDRPER subroutine	534 to 540
LDRPLN subroutine	541 to 544
LIST function	707 to 709
LOS function	545 to 547
LOSLIS function	548 to 549
LOSXYZ function	550 to 553
LSEARCH function	710 to 711
MAPBDG subroutine	340 to 345
MAPORT subroutine	346 to 347
MAPSIT subroutine	348 to 354
MDESIR subroutine	554 to 558
MPREAD function	355
NACTSU function	559 to 561
NBRREG function	562
NDRFIR function	563 to 565
NEWLDR function	566 to 567
NEWFCP subroutine	568 to 571
NEWRC2 function	712 to 715
NEWREC function	716 to 717
NEWREF function	718
NEXT function	719 to 720
NITEMS function	721 to 722
NRTEST function	723 to 730
NULIFY subroutine	731
NWPNS subroutine	572 to 574
NZERO subroutine	732
OBS subroutine	575 to 577
OBSERV subroutine	578 to 580
ODESIR subroutine	581 to 583
OUTBLK subroutine	356
OUTLIS subroutine	733 to 734
OUTMAP subroutine	357
OUTPLX subroutine	735 to 740
OUTREE subroutine	741
OUTSET subroutine	742 to 744
/PARS/	282, 285, 289
/PARS1/	282, 285, 289
/PARS2/	282, 285, 289
/PARS3/	283, 289
/PARS4/	284
PARSRF subroutine	745 to 746
PEQUAL function	584 to 586
Plex Preprocessor - Main Program	289 to 296, 757, 781
PLANAC subroutine	587 to 595
PLNBLK subroutine	596 to 597
PLNOBS subroutine	598 to 599
POBS function	600 to 601
POCT subroutine	602 to 604

POROPN subroutine	609 to 610
PORTST subroutine	605 to 608
QXREFS subroutine	747 to 748
RECDATA file	757, 285 to 287
RECDESCRIPT file	757 to 760
/RECREF/	284 to 290
REPORT subroutine	749 to 753
RFDATA file	757, 764 to 765
SCALE subroutine	611 to 612
SDESIR subroutine	613 to 615
SECURE subroutine	616 to 617
SENSE subroutine	618 to 621
SNSACT subroutine	622 to 625
/STATEV/	282 to 289
SURFAC subroutine	626 to 629
TRAVEL function	630 to 634
TRGLST subroutine	635 to 638
UPACTS subroutine	639 to 641
VAL function	754
VAL1 function	755

DISTRIBUTION:

U.S. Nuclear Regulatory Commission (10)
MS 1130SS
Washington, DC 20555
Attn: R. Robinson

Los Alamos Scientific Laboratory
Attn: G. R. Keepin, R. A. Gore, E. P. Schlonka, D. G. Rose
Los Alamos, NM 87544

Allied-General Nuclear Services
Attn: G. Molen
P.O. Box 847
Barnwell, SC 29812

Lawrence Livermore Laboratory
University of California
P.O. Box 808
Attn: A. J. Poggio
Livermore, CA 94550

Pritsker and Associates, Inc.
P.O. Box 2413
Attn: F. H. Grant
West Lafayette, In 47906

Union Carbide Corporation
Nuclear Division
Bldg. 7601
Attn: D. Swindle
Oak Ridge, TN 37830

400	C. Winter
1000	G. A. Fowler
1213	V. E. Gibbs
1230	W. L. Stevens, Attn: R. E. Smith, 1233
1700	W. C. Myre
1710	V. E. Blake, Attn: M. R. Madsen, J. W. Kane
1716	R. L. Wilde, Attn: B. D. Link, 1716
1730	C. H. Mauney, Attn: J. D. Williams, 1739
1750	J. E. Stiegler, Attn: M. J. Eaton, 1759
1754	I. G. Waddoups, Attn: J. L. Todd, 1754
1758	C. E. Olson, Attn: D. D. Boozer, G. A. Kinemond, 1758
1760	J. Jacobs, Attn: M. N. Cravens, J. M. deMontmollin, 1760A
1761	T. A. Sellers, Attn: A. E. Winblad, J. L. Darby, 1761
1762	H. E. Hansen
1765	D. S. Miyoshi
4400	A. W. Snyder
4410	D. J. McCloskey
4413	N. F. Ortiz
4414	D. E. Bennett
4414	S. L. Daniel

DISTRIBUTION (Cont)

4414 M. S. Hill
4414 G. B. Varnado
4416 L. D. Chapman (5)
4416 K. G. Adams
4416 J. A. Allensworth
4416 H. A. Bennett
4416 D. Engi (10)
4416 L. M. Grady
4416 C. P. Harlan
4416 R. D. Jones
4416 M. T. Olascoaga
4416 C. J. Pavlakos
4416 D. W. Sasser
4416 D. R. Strip
5000 J. K. Galt
5600 D. B. Shuster, Attn: A. A. Lieber, M. M. Newsom, 5620,
R. C. Maydew 5630
5640 G. J. Simmons, Attn: R. J. Thompson, 5641,
L. F. Shampine, 5642
5641 C. A. Morgan
5642 B. L. Hulme
8266 E. A. Aas
3141 T. L. Werner (5)
3151 W. L. Garner (3)
For: DOE/TIC (Unlimited Release)
3154-3 R. P. Campbell (25)
For NRC Distribution to NTIS