October 1998

In this issue

3  Auto-install program for APPC connections
9  A CEMT log for CICS 4.1
24  DL/I database display and control facility
37  Automatic change from CSSN to CESN
39  More on macros to define statements – part 2
48  CICS news

© Xephon plc 1998
Auto-install program for APPC connections

This program is intended for those already using terminal auto-install in COBOL II, and who intend to use APPC Connection auto-install (in single or parallel sessions) for CICS/ESA 4.1.

To achieve this the communication area DFHZATDY in the CICS410.SDFHMAC, and the source code in CICS410.SDFHSAMP, must be converted.

The inclusion of the code in the terminal auto-install program is simple. As for the terminals, to simplify the conversion algorithm, the netnames of the connection should follow a previously-established rule. A model should exist for each connection type foreseen in the program. For example, an attempt to install a model that was not defined resulted in a message with code X'FA0C': ‘DFHZC6922 E date time applid Parameter list error during autoinstall for NETNAME netname. Code X'code’.

AUTOINST

IDENTIFICATION DIVISION.
PROGRAM-ID. AUTOINST.
* ****************************************************
* ****************************************************
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 WS-X    SYNC COMP PIC 9(4) VALUE ZEROS.
* Ø1 CODE-TYPE            PIC X(1) VALUE ' '.
  88 INSTALL-CODE           VALUE IS 'Ø'.
  88 DELETE-CODE            VALUE IS '1'.
  88 APPC-PS-CINIT          VALUE IS '2'.
  88 APPC-PS-BIND           VALUE IS '3'.
  88 APPC-SS-BIND           VALUE IS '4'.
  88 SHIPPED-TERM-CODE      VALUE IS '7'.
  88 SHIPPED-RSE-CODE       VALUE IS '8'.
Ø1 W-DADOS.
Ø5 WS-POSXY                PIC X(36) VALUE
© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

 Вот 23456789АВСДЕFGHIJKLMNOPQRSTUVWXYZ'.

₀₅ WS-POSIXY REDEFINES WS-POSXY
     PIC X OCCURS 36.

₀₅ WS-NAME.
  ₁₀ WS-POXY  PIC X(1).
    ₈₈ NET-X  VALUE 'X'.
    ₈₈ NET-Y  VALUE 'Y'.
    ₈₈ NET-Z  VALUE 'Z'.
  ₁₀ WS-POSX  PIC X(3) VALUE SPACES.
  ₁₀ WS-POSX-N REDEFINES
     WS-POXY  PIC 9(3).
  ₁₀ WS-POSU  PIC X(1).
    ₈₈ TIPO-LU₀  VALUE '₀'.
    ₈₈ TIPO-LU₁  VALUE '₁'.
    ₈₈ TIPO-LU₂  VALUE '₂'.
    ₈₈ TIPO-LU₃  VALUE '₃'.
    ₈₈ TIPO-LU₆₂  VALUE '₆'.
  ₁₀ WS-POSZ  PIC X(2) VALUE SPACES.
  ₁₀ WS-POSZ-N REDEFINES
     WS-POSZ  PIC 9(2).
  ₁₀ FILLER  PIC X(1) VALUE SPACES.

₀₁ VARIABLES.

₀₅ WS-TERMID.
  ₁₀ WS-CHR₁  PIC X(1) VALUE SPACE.
  ₁₀ WS-CHR₂  PIC X(1) VALUE SPACE.
  ₁₀ WS-CHR₃  PIC X(1) VALUE SPACE.
  ₁₀ WS-CHR₄  PIC X(1) VALUE SPACE.

******************************************************************************
LINKAGE SECTION.
******************************************************************************

₀₁ DFHCOMMAREA.
  ₀₂ FUNCTION-FIELD.
    ₀₃ REQUEST-TYPE  PIC X(1).
    ₀₃ REST  PIC X(3).
  ₀₂ NET-PTR  USAGE IS POINTER.
  ₀₂ MOD-PTR  USAGE IS POINTER.
  ₀₂ RET-FLD-PTR  USAGE IS POINTER.
  ₀₂ CINRU-PTR  USAGE IS POINTER.
  ₀₂ SYSID-PTR  USAGE IS POINTER.
  ₀₂ CORRID-PTR  USAGE IS POINTER.

* 

₀₁ APPC-COMMAREA REDEFINES DFHCOMMAREA.
  ₀₂ APPC-FUNCTION-FIELD.
    ₀₃ APPC-REQUEST-TYPE  PIC X(1).
    ₀₃ FILLER  PIC X(3).
  ₀₂ APPC-NAME-PTR  USAGE IS POINTER.
  ₀₂ APPC-CINIT-PTR  USAGE IS POINTER.
  ₀₂ APPC-SELECTED-PTR  USAGE IS POINTER.
  ₀₂ APPC-SYNCELEVEL-PTR  USAGE IS POINTER.
PROCEDURE DIVISION.

IF EIBCALEN EQUALS Ø THEN PERFORM RETURN-LINE.
MOVE REQUEST-TYPE TO CODE-TYPE.

IF INSTALL-CODE THEN
PERFORM INSTALL-PARAGRAPH
THRU INSTALL-PARAGRAPH-EXIT
ELSE
IF APPC-PS-CINIT OR
APPC-PS-BIND OR
APPC-SS-BIND THEN
PERFORM APPC-PARAGRAPH
ELSE
IF SHIPPED-TERM-CODE THEN
PERFORM SHIPPED-PARAGRAPH
ELSE
IF SHIPPED-RSE-CODE THEN
PERFORM SHIPPED-PARAGRAPH
ELSE
IF DELETE-CODE THEN
PERFORM DELETE-PARAGRAPH.
RETURN-LINE.
EXEC CICS RETURN END-EXEC.
GOBACK.
INSTALL-PARAGRAPH.
SET ADDRESS OF NETNAME        TO NET-PTR.
SET ADDRESS OF MODELNAME      TO MOD-PTR.
SET ADDRESS OF SELECTION-DATA TO RET-FLD-PTR.
SET ADDRESS OF CINIT-AREA     TO CINRU-PTR.
*    IF NO-MODELS EQUAL Ø THEN PERFORM RETURN-LINE.
MOVE NETNAME-NAME TO WS-NETNAME.
——— put your code for install terminals ——
APPC-PARAGRAPH.
SET ADDRESS OF APPC-TEMPLATE-NETNAME TO APPC-TEMPLATE-NET-PTR.
SET ADDRESS OF APPC-TEMPLATE-SYSID   TO APPC-TEMPLATE-SYSID-PTR.
SET ADDRESS OF SELECTION-DATA        TO APPC-SELECTED-PTR.
SET ADDRESS OF NETNAME               TO APPC-NETNAME-PTR.
MOVE NETNAME-NAME                    TO WS-NETNAME.
SET ADDRESS OF APPC-SYSID            TO APPC-SYSID-PTR.
PERFORM SET-CONNECTIONS.
MOVE WS-TERMID TO APPC-SYSID.
MOVE LOW-VALUES TO RET-STATUS.
GO TO RETURN-LINE.
SET-CONNECTIONS.
IF TIPO-LU62
   PERFORM CONVERT-NETNAME-CONNID.
   MOVE 'A' TO WS-CHR1
   PERFORM SET-CONNECTION-TYPE
   MOVE LOW-VALUES TO RET-STATUS
   ELSE
   MOVE HIGH-VALUES TO RET-STATUS
   GO TO RETURN-LINE.
CONVERT-NETNAME-CONNID
*    CONVERT WS-NETNAME TO WS-TERMID ——
* FROM      ws-posx (3 chr) to 2 chr
SET-CONNECTION-TYPE.
******************************************************************************

* MODELS for INSTALL of APPC CONNECTIONS
* CBPS - DEFAULT MODEL FOR APPC CONN W/PARALLEL SESSIONS
* CMAS - MODEL FOR APPC CONN TO APLICATION
*
******************************************************************************

IF WS-POSZ-N = ZEROS
  MOVE 'CMAS' TO APPC-TEMPLATE-SYSID
ELSE
  MOVE 'CBPS' TO APPC-TEMPLATE-SYSID.

RDO DEFINITIONS

CONNECTION DEFINITION

  Connection     : CMAS
  Group          : TCTAI62
  Description ==> APPC AUTOINSTALL ASSINATURAS PARALLEL SESSION

CONNECTION IDENTIFIERS

  Netname ==> TMPLTASS
  INDsys ==> 

REMOTE ATTRIBUTES

  REMOTESYSTEM ==> 
  REMOTENAME ==> 
  REMOTESYSNET ==> 

CONNECTION PROPERTIES

  ACCesstype ==> Vtam | IRc | INDIRECT | Xm
  PROtocol ==> Appc | Lu61 | Exci
  CONntype ==> Generic | Specific
  SINGLESess ==> No | Yes
  DAtastream ==> User | 327Ø | SCs | STrfield | Lms
  REcordformat ==> U | Vb
  QueueLimit ==> No | Ø-9999
  Maxqtime ==> No | Ø-9999

OPERATIONAL PROPERTIES

  AUtoconnect ==> Yes | No | Yes | All
  INService ==> Yes | No

SECURITY

  SEcurityname ==> 
  ATtachsec ==> Local | Identify | Verify | Mixidpe
  Persistent

  BINDPassword : PASSWORD NOT SPECIFIED
  BINDSecurity ==> No | Yes
  Usedfltuser ==> No | Yes

RECOVERY

  PSrecovery ==> Sysdefault | None

******************************************************************************
SESSION DEFINITION

Sessions       : CMAS
Group          : TCTAI62
DEscription    ==> APPC AUTOINSTALL SIGNATURES PARALLEL SESSION

SESSION IDENTIFIERS
Connection     ==> CMAS
SESSName       ==> NETNAMEQ
MOdename       ==> #INTER

SESSION PROPERTIES
Protocol       ==> Appc               Appc | Lu61 | Exci
MAXimum        ==> 008 , 004          0-999
RECEIVEPfx     ==> 1-999
SENDPfx        ==> 1-999
SENDSize       ==> 01024             1-30720
RECEIVESize    ==> 01024             1-30720
SESSPriority   ==> 000               0-255
Transaction    :

OPERATOR DEFAULTS
OPERId         :
OPERPriority   ==> 000               0-255
OPERRs1        : 0
OPERSecurity   : 1

PRESET SECURITY
USERId         ==>  

OPERATIONAL PROPERTIES
Autoconnect    ==> Yes                No | Yes | All
INservice      :
Buildchain     ==> Yes                Yes | No
USERAreaLen    ==> 000               0-255
IOAreaLEN      ==> 00000 , 00000     0-32767
RELreq         ==> No                 No | Yes
DISreq         ==> No                 No | Yes
NEPclass       ==> 000               0-255

RECOVERY
RECOVOption    ==> Sysdefault        Sysdefault | Clearconv | Uncondrel | None
Releasesess    :

RECOVNotify    : None                None | Message | Transaction

Carlos Gomes Carvalho
Systems Engineer
Grupo BPI (Portugal)
A CEMT log for CICS 4.1

THE PROBLEM
The CICS master terminal transaction (CEMT) permits authorized users to modify the gamut of resources allocated to CICS. Security measures exist to restrict access to CEMT; however, an audit trail of CEMT activity has been lacking. The absence of such a log is felt most acutely in shops like ours where applications developers have carte blanche in their use of CEMT. The code offered here attempts to fill this void.

COMPONENTS OF THE LOG
XZCOUT$ and LOGWRITE are the two program components of our CEMT log. XZCOUT$ is an XZCOUT global user exit (GLUE) program, written in Assembler. Its function is to parse terminal output from CEMT commands, write formatted records to a Global Work Area (GWA), and set TCTTETC (next trans-id) to LWRT. XZCOUT$ is activated from a PLTPI program with the following command:

```
EXEC CICS ENABLE EXIT('XZCOUT')
    PROGRAM('XZCOUT$')
    GALENGTH(10000)
    START
```

LOGWRITE is a CICS command-level program, written in COBOL II, that is initiated by transaction LWRT. Its function is to write the formatted CEMT log records from the GWA buffer to a transient data queue CSMT.

OUTPUT FROM THE LOG
Figure 1 shows a sample output from our CEMT log. For each invocation of CEMT from a terminal, a header record appears with the eyecatcher ‘<CEMTLOG>’. The header record displays the user-id and terminal-id of the operator issuing the command, as well as the date, time, and format of the CEMT command. Finally, to the far right of the header is the overall response from CICS – either ‘NORMAL’
or ‘(number of) ERRORS’. If any CICS resources are altered, either by the command itself or by operator overtyping, the log shows new resource attributes in individual records following the header. The third example in Figure 1 illustrates a case where the operator performed a pattern-matching inquiry on terminal-ids starting with ‘T’ and then over-typed ‘Ins’ with ‘Out’ for terminal T012.

**Figure 1: Sample output from CEMT log**

```assembly
XZCOUT$
TITLE 'XZCOUT EXIT TO LOG CEMT COMMANDS'
PRINT ON, NOGEN
DFHUEXIT TYPE=EP, ID=XZCOUT
DFHUEXIT TYPE=XPIENV
COPY DFHSMMCY PARMLIST FOR STORAGE CONTROL XPI
COPY DFHXMIQY PARMLIST FOR INQUIRE XACTION XPI
COPY DFHTIOA TERMINAL INPUT OUTPUT AREA
COPY DFHTCTTE TERMINAL CONTROL TABLE ENTRY
TIOABAR EQU R4
TCTTEAR EQU R5
* ZCODS DSECT
ZCOTRAN DS CL4 current tran (CEMT activates logging)
ZCOTYPE DS CL4 prog, file, tran, term, tdq etc.
ZCOGWAEA DS F GWA ending address
ZCONCNT DS H number of ptrs in NLIST (NORMAL resp)
ZCOLOG DS ØH
```

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
ZCODATE DS CL8
   DS C
ZCOTIME DS CL8
   DS C
ZCOUID DS CL8
   DS C
ZCOTERM DS CL4
   DS C
ZCOCMD DS CL5Ø          CEMT command as entered
ZOCOMDL EQU *-ZCOCMD
ZCORESP DS CL15        overall response (NORMAL or ?? ERRORS)
ZCORESPL EQU *-ZCORESP
ZCOFILL DS CL3
ZCOLOGL EQU *-ZCOLOG
ZCONLIST DS 48F       list of pointers to NORMAL response(s)
ZCOFIXL EQU *-ZCOSDS  fixed length of dynamic storage
ZCOTIOA DS ØH        tioa - variable length
*
GWAXZODS DSECT       global work area dummy sect
GWALOG DS CL1ØØ
*
XZCOUT$ CSECT
XZCOUT$ AMODE 31
XZCOUT$ RMODE ANY
SAVE (14,12)            save caller regs
LR    R12,R15          establish pgm base reg
USING XZCOUT$%$12
LR    R2,R1           address GLUE parmlist
USING DFHUEPAR,R2
L     TIOABAR,UEPTIOA address TIOA
USING DFHTIOA,TIOABAR
L     TCTTEAR,UEPTCTTE address TCTTE
USING DFHTCTTE,TCTTEAR
*
GETMAIN DS ØH
LH    R9,TIOATDL       get variable TIOA length
LTR    R9,R9           if none,
BNP   GLUEXIT          ... vamonos
LA    R9,ZCOFIXL(R9)   add fixed length
L    R6,UEPXSTOR       address XPI storage
USING DFHSMMC_ARG,R6   map XPI storage
L     R13,UEPSTACK     save exit handler stack
DFHSMCMX CALL,CLEAR,IN, X
   FUNCTION(GETMAIN), X
   GET_LENGTH((R9)), X
   INITIAL_IMAGE(X'4Ø'), X
   STORAGE_CLASS(USER), X
   SUSPEND(NO), X
   OUT, X
ADDRESS((R3)),
RESPONSE(*),
REASON(*)

CLI SMMC_RESPONSE,SMMC_OK
BE SAVESTOR
LA R9,WTOGMERR
WTO MF=(E,(R9))
B GLUEXIT

* SAVESTOR DS ØH
USING ZCODS,R3 map acquired storage
ST R3,Ø(R6) store GM addr in 1st 4 bytes of UEPXSTOR
LA R6,4(R6) use remaining 256 bytes to map XPI parms
DROP R6

* INQTRAN DS ØH
L R6,UEPXSTOR address XPI storage
LA R6,4(R6)
USING DFHXMIQ_ARG,R6 map XPI storage
L R13,UEPSTACK save exit handler stack
DFHXMIQX CALL,CLEAR,IN,
FUNCTION(INQUIRE_TRANSACTION),
OUT,
 TRANSACTION_ID(ZCOTRAN),
 USERID(ZCOURID),
 RESPONSE(*),
 REASON(*)
CLI XMIQ_RESPONSE,XMIQ_OK
BE PARSE
LA R9,WTOIXERR
WTO MF=(E,(R9))
B FREEMAIN

* PARSE DS ØH
DROP R6

CLC ZCOTRAN,CEMT is this CEMT?
BNE FREEMAIN n - then see ya
LH R9,TIOATDL get TIOA length
LA R8,ZCOTIOA
LA R10,TIOADBA
LR R11,R9
MVCL R8,R10 copy TIOA to our dyn storage
LA R8,ZCOTIOA
LR R11,R8
AH R11,TIOATDL
BCTR R11,R0
LA R10,1
LA R8,8(R8) shorten parse
MVC ZCOTYPE,Ø(R8)
LA  R6,ZCONLIST
XR  R9,R9

PARSE1  CLC  Ø(6,R8),NORMAL    successful command?
     BNE  PARSE1A                n - parse other keys
     ST  R8,Ø(R6)                y - store in NLIST
     LA  R6,4(R6)                bump NLIST ptr
     LA  R9,1(R9)                NLIST ctr
     B   PARSE1C                 resume parse for NORMAL

PARSE1A CLC  Ø(8,R8),RESPONSE  RESPONSE keyword?
     BE   PARSE2                 y - done
     CLC  Ø(7,R8),SESSION       SESSION ENDED?
     BNE  PARSE1B                n - parse error keywords
     MVC  TCTTETC,LWRT           tran writes GWA record to TD
     B   FREEMAIN                get out

PARSE1B CLC  Ø(6,R8),SYNTAX    SYNTAX CHECK?
     BE   FREEMAIN               y - get out
     CLC  Ø(5,R8),ENTER          ENTER ... ?
     BE   FREEMAIN               y - get out

PARSE1C BXLE  R8,R1Ø,PARSE1
     B   FREEMAIN                get out

PARSE2  STH  R9,ZCONCNT         save count of NLIST ptrs
     MVC  ZCORESP,11(R8)

PARSE2A CLC  Ø(4,R8),TIME
     BE   PARSE2B
     BXLE  R8,R10,PARSE2A

PARSE2B MVC  ZCOTIME,7(R8)
     MVC  ZCODATE,23(R8)
     MVC  ZCOCMD,37(R8)
     MVC  ZCOTERM,TCTTETI
     DROP  TCTTEAR
     DROP  TIOABAR
     LA  R8,ZOCMD
     LR  R11,R8
     LA  R11,ZOCMDL(R11)
     BCTR  R11,R0
     LA  R10,1

PARSE3 CLC  Ø(4,R8),CMDDLIM
     BE   PARSE4
     BXLE  R8,R10,PARSE3
     LA  R8,ZOCMD

PARSE4 MVI  Ø(R8),X'40'        insert padding blank
     SR  R11,R8
     BCTR  R11,R0
     EX  R11,CMDMVC
     B   PARSE5

CMDMVC MVC  1(1,R8),Ø(R8)

PARSE5 LA  R8,ZCORESP
     LR  R11,R8
     LA  R11,ZCORESPL(R11)
BCTR R11, R0
LA R10, 1

PARSE6 CLC @4(R8), RESPDLIM
BE PARSE7
BXLE R8, R10, PARSE6
LA R8, ZCORESP

PARSE7 MVI (@R8), X'40'
SR R11, R8
BCTR R11, R0
EX R11, RESP MVC
B PUTLOG

RESPMVC MVC 1(1, R8), Ø(R8)
*

PUTLOG DS ØH
ICM R8, B'1111', UEPGAA
BNZ PUTLOG1
LA R9, WTOGWA1
WTO MF=(E, (R9))
B FREEMAIN

PUTLOG1 L R11, UEPGAL
LH R11, Ø(R11)
AR R11, R8
ST R11, ZCOGWAEA
BCTR R11, R0
LA R10, ZCOLOGL
USING GWAXZODS, R8

PUTLOG2 CLI GWALOG, X'00'
BE PUTLOG3
BXLE R8, R10, PUTLOG2
LA R9, WTOGWA2
WTO MF=(E, (R9))
B FREEMAIN

PUTLOG3 MVC GWALOG, ZCOLOG
LH R9, ZCONCNT
LTR R9, R9
BZ FREEMAIN
LA R4, ZCONLIST
CLC ZCOTYPE, PROG
BNE PUTLOG3A
LA R5, PUTPROG
B PUTLOG4

PUTLOG3A CLC ZCOTYPE(3), TRA
BNE PUTLOG3B
LA R5, PUTTRAN
B PUTLOG4

PUTLOG3B CLC ZCOTYPE(3), FIL
BNE PUTLOG3C
LA R5, PUTFILE
B PUTLOG4

PUTLOG3C CLC ZCOTYPE(3), TER

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
BNE PUTLOG3D
LA R5,PUTTERM
B PUTLOG4

PUTLOG3D CLC ZCOTYPE(3),TDQ
BNE PUTLOG3E
LA R5,PUTTDQ
B PUTLOG4

PUTLOG3E CLC ZCOTYPE(3),TAS
BNE FREEMAIN
LA R5,PUTTAS
B PUTLOG4

PUTLOG4 LA R8,ZCOLOG(R8)
C R8,ZCOGWAEM
BL PUTLOG4A
LA R9,WTOGWA2
WTO MF=(E,(R9))
B FREEMAIN

PUTLOG4A MVI ZCOLOG,X'4Ø' clear log buffer
MVC ZCOLOG+1(99),ZCOLOG
BAL R7,Ø(R5) link to proper format routine
MVC GWALOG,ZCOLOG copy buffer to GWA
LA R4,4(R4)
BCT R9,PUTLOG4 repeat for each NLIST ptr
B FREEMAIN

* PUTFILE DS ØH
L R6,Ø(R4)
S R6,=F'75'
MVC ZCOLOG(13),Ø(R6) file
MVC ZCOLOG+14(3),15(R6) VSAM
MVC ZCOLOG+18(3),20(R6) ope/clo
MVC ZCOLOG+22(3),25(R6) ena/dis
MVC ZCOLOG+26(3),30(R6) rea
MVC ZCOLOG+30(3),35(R6) upd
MVC ZCOLOG+34(3),40(R6) add
MVC ZCOLOG+38(3),45(R6) bro
MVC ZCOLOG+42(3),50(R6) del
MVC ZCOLOG+46(3),65(R6) shr
MVC ZCOLOG+50(4),88(R6) dsn
MVC ZCOLOG+54(44),94(R6)
MVC ZCOLOG+98(1),140(R6)
BR R7

* PUTPROG DS ØH
L R6,Ø(R4)
S R6,=F'65'
MVC ZCOLOG(14),Ø(R6) program
MVC ZCOLOG+15(12),16(R6) length
MVC ZCOLOG+28(3),30(R6) language
MVC ZCOLOG+32(3),35(R6) pro/map
MVC ZCOLOG+36(3),40(R6)  ena/dis
MVC ZCOLOG+40(3),45(R6)  pri/shr
MVC ZCOLOG+44(3),55(R6)  edf
MVC ZCOLOG+48(8),78(R6)  res count
MVC ZCOLOG+57(15),88(R6)  use count
MVC ZCOLOG+73(3),105(R6)  datalocation
MVC ZCOLOG+77(3),110(R6)  execkey
MVC ZCOLOG+81(3),115(R6)  api
BR R7

* PUTTAS DS ØH
  L R6,Ø(R4)
  S R6,=F'70'
  CLC 25(3,R6),FAC
  BE PUTTAS1
MVC ZCOLOG(12),1(R6)  task
MVC ZCOLOG+13(9),15(R6)  tran
MVC ZCOLOG+33(3),36(R6)  sus/run
MVC ZCOLOG+37(3),41(R6)  ter/tas
MVC ZCOLOG+41(4),46(R6)  priority
MVC ZCOLOG+45(3),52(R6)
MVC ZCOLOG+48(1),57(R6)
B PUTTASX

PUTTAS1 MVC ZCOLOG(12),Ø(R6)  task
MVC ZCOLOG+13(9),14(R6)  tran
MVC ZCOLOG+23(9),25(R6)  facility
MVC ZCOLOG+33(3),36(R6)  sus/run
MVC ZCOLOG+37(3),41(R6)  ter/tas
MVC ZCOLOG+41(4),46(R6)  priority
MVC ZCOLOG+45(3),52(R6)
MVC ZCOLOG+48(1),57(R6)
B PUTTASX

PUTTASX BR R7

* PUTTDQ DS ØH
  L R6,Ø(R4)
  S R6,=F'67'
  CLC 27(3,R6),INT  intrapartition?
  BE PUTTDQ1
  CLC 27(3,R6),EXT  extrapartition?
  BE PUTTDQ2

PUTTDQ1 MVC ZCOLOG(9),Ø(R6)  tdq
MVC ZCOLOG+10(4),11(R6)  trigger level
MVC ZCOLOG+14(5),17(R6)
MVC ZCOLOG+19(1),24(R6)
MVC ZCOLOG+21(3),27(R6)  intra/extra/indirect
MVC ZCOLOG+25(3),42(R6)  ena/dis
CLC ØØ(3,R6),TRA  tran associated?
BNE PUTTDQX
MVC ZCOLOG+29(9),ØØ(R6)  tran
B PUTTDQX
PUTTDQ2 MVC ZCOLOG(9),3(R6) tdq
MVC ZCOLOG+25(3),42(R6) ena/dis
MVC ZCOLOG+29(3),47(R6) ope
B PUTTDQX

PUTTDQX BR R7

* PUTTERM DS ØH
L R6,Ø(R4)
S R6,=F'66'
CLC 11(3,R6),TRA
BE PUTTERM1
MVC ZCOLOG(9),1(R6) term-id
B PUTTERM2

PUTTERM1 MVC ZCOLOG(9),Ø(R6) term-id
MVC ZCOLOG+10(9),11(R6) tran-id

PUTTERM2 MVC ZCOLOG+20(4),22(R6) priority
MVC ZCOLOG+24(3),28(R6)
MVC ZCOLOG+27(1),33(R6)
MVC ZCOLOG+29(3),36(R6) pag
MVC ZCOLOG+33(3),41(R6) ins
MVC ZCOLOG+37(3),46(R6) ati
MVC ZCOLOG+41(3),51(R6) tti
MVC ZCOLOG+45(13),79(R6) netname
MVC ZCOLOG+59(3),94(R6) acq
BR R7

* PUTTRAN DS ØH
L R6,Ø(R4)
S R6,=F'74'
MVC ZCOLOG(9),Ø(R6) trans-id
MVC ZCOLOG+10(4),11(R6) priority
MVC ZCOLOG+14(3),17(R6)
MVC ZCOLOG+17(1),22(R6)
MVC ZCOLOG+20(13),25(R6) program
MVC ZCOLOG+34(4),40(R6) tclass
MVC ZCOLOG+38(8),46(R6)
MVC ZCOLOG+46(1),56(R6)
MVC ZCOLOG+49(3),59(R6) ena/dis
MVC ZCOLOG+53(3),64(R6) purge
BR R7

* FREEMAIN DS ØH
L R6,UEPXSTOR address XPI storage
LA R6,4(R6)
USING DFHSMMC_ARG,R6 map XPI storage
L R13,UEPSTACK save exit handler stack
DFHSMMCX CALL,CLEAR,IN,
FUNCTION(FREEMAIN), X
ADDRESS((R3)), X
STORAGE_CLASS(USER), X

OUT,
RESPONSE(*),
REASON(*)
CLI SMMC_RESPONSE,SMMC_OK
BE GLUEXIT
LA R9,WTOFMERR
WTO MF=(E,(R9))
B GLUEXIT
*
GLUEXIT DS ØH
    standard GLUE exit code
L R13,UEPEPSA
    RETURN (14,12),RC=UERCNORM
*
CEMT DC CL4'CEMT'
TIME DC CL4'TIME'
NORMAL DC CL6'NORMAL'
RESPONSE DC CL8'RESPONSE'
RESULTS DC CL7'RESULTS'
SESSION DC CL7'SESSION'
SYNTAX DC CL6'SYNTAX'
ENTER DC CL5'ENTER'
FIL DC CL3'Fil'
PROG DC CL4'Prog'
TAS DC CL3'Tas'
FAC DC CL3'Fac'
TDQ DC CL3'Tdq'
INT DC CL3'Int'
EXT DC CL3'Ext'
TER DC CL3'Ter'
TRA DC CL3'Tra'
CMDDLIM DC XL4'4Ø1311C1'
RESPDLIM DC XL4'115C4F1D'
LWRT DC CL4'LWRT'
*
WTOGMERR WTO 'XZCOUT - GETMAIN ERROR',ROUTCDE=(14),MF=L
WTOFMERR WTO 'XZCOUT - FREEMAIN ERROR',ROUTCDE=(14),MF=L
WTOIXERR WTO 'XZCOUT - INQUIRE XACTION ERROR',ROUTCDE=(14),MF=L
WTOGWA1 WTO 'XZCOUT - NO GWA PROVIDED',ROUTCDE=(14),MF=L
WTOGWA2 WTO 'XZCOUT - GWA EXHAUSTED',ROUTCDE=(14),MF=L
*
LTORG
END XZCOUT$

LOGWRITE
CBL XOPTS(CICS,FE,SP)
*
IDENTIFICATION DIVISION.
PROGRAM-ID. LOGWRITE.
* ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-3090.
 OBJECT-COMPUTER. IBM-3090.
 *
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01 CMD-VALUES.
    05 CMD-EXIT PIC X(08) VALUE 'XZCOUT'.
    05 CMD-PROG PIC X(08) VALUE 'GLHXZOUT'.
    05 CMD-GAL PIC S9(4) COMP.
    05 CMD-RESP PIC S9(4) COMP.
    05 CMD-TPR POINTER.
    05 CMD-TDQ PIC X(04) VALUE 'CSMT'.
    05 CMD-FROM.
       10 CMD-FROM-HDR PIC X(10) VALUE SPACE.
       10 CMD-FROM-LOG PIC X(100) VALUE SPACE.
    05 CMD-REQID PIC X(08) VALUE 'LOGWRITE'.
    05 CMD-INTO.
       10 F PIC X(3).
       10 CMD-INTO-TRAN PIC X(4).
    05 CMD-INTO-LEN PIC S9(4) COMP.
    05 CMD-INTO-MAX PIC S9(4) VALUE +7 COMP.
 01 MSG-VALUES.
    05 MSG-EXIT-ERR.
       10 F PIC X(20) VALUE 'LOGWRITE - XZCOUT IN'.
       10 F PIC X(20) VALUE 'ACTIVE OR NO GLOBAL '.
       10 F PIC X(20) VALUE 'WORK AREA ASSIGNED. '.
 01 WRK-VALUES.
    05 WRK-COLON PIC X VALUE ':'.
    05 WRK-DOT PIC X VALUE '.'.
    05 WRK-SLASH PIC X VALUE '/'.
    05 WRK-HDR PIC X(10) VALUE '<CEMTLOG>'.
    05 WRK-TALLY PIC S9(4) VALUE ZERO COMP.
COPY DFHAID.
LINKAGE SECTION.
 01 DFHCOMMAREA PIC X.
 01 GWA.
    05 GWA-CEMT-LOG OCCURS 100 INDEXED GWAX.
       10 GWA-CEMT-DATE PIC X(08).
       10 F PIC X(01).
       10 GWA-CEMT-TIME PIC X(08).
       10 F PIC X(01).
       10 GWA-CEMT-UID PIC X(08).
       10 F PIC X(01).
       10 GWA-CEMT-TERM PIC X(04).
       10 F PIC X(01).
       10 GWA-CEMT-CMD PIC X(50).
       10 GWA-CEMT-RESP PIC X(15).
PROCEDURE DIVISION.

000-WRITE-CEMT-LOG.
EXEC CICS HANDLE CONDITION
   ERROR(900-ERRORS)
END-EXEC.
EXEC CICS EXTRACT EXIT
   PROGRAM(CMD-PROG)
   GALENGTH(CMD-GAL)
   GASET(CMD-PTR)
   RESP(CMD-RESP)
END-EXEC.
IF CMD-RESP = DFHRESP(NORMAL)
   SET ADDRESS OF GWA TO CMD-PTR
   PERFORM VARYING GWAX FROM 1 BY 1
      UNTIL GWA-CEMT-LOG(GWAX) = LOW-VALUE
         OR GWAX = 100
      IF GWAX <= 100 AND GWA-CEMT-LOG(GWAX) > LOW-VALUE
         INSPECT GWA-CEMT-DATE(GWAX)
         TALLYING WRK-TALLY FOR ALL WRK-DOT
         REPLACING ALL WRK-DOT BY WRK-SLASH
         IF WRK-TALLY > ZERO
            INSPECT GWA-CEMT-TIME(GWAX)
            REPLACING ALL WRK-DOT BY WRK-COLON
            MOVE WRK-HDR TO CMD-FROM-HDR
            MOVE ZERO TO WRK-TALLY
         ELSE
            MOVE SPACE TO CMD-FROM-HDR
         END-IF
      END-IF
   END-PERFORM
ELSE
   MOVE MSG-EXIT-ERR TO CMD-FROM-LOG
   PERFORM 300-WRITE-TRANSIENT-DATA THRU 300-EXIT
END-IF
END-EXEC.

300-WRITE-TRANSIENT-DATA.
EXEC CICS WRITEQ TD
   QUEUE(CMD-TDQ)
FROM(CMD-FROM)
LENGTH(LENGTH OF CMD-FROM)
END-EXEC.
300-EXIT.
EXIT.
*
600-RESTART-THIS-TRAN.
EXEC CICS IGNORE CONDITION NOTFND END-EXEC.
EXEC CICS CANCEL
  ReqID(CMD-REQID)
  TRANSID(EIBTRNID)
END-EXEC.
EXEC CICS START
  TRANSID(EIBTRNID)
  AFTER MINUTES(1)
  ReqID(CMD-REQID)
END-EXEC.
600-EXIT.
EXIT.
*
700-EXECUTE-USER-TRAN.
EXEC CICS RECEIVE
  INTO(CMD-INTO)
  LENGTH(CMD-INTO-LEN)
  MAXLENGTH(CMD-INTO-MAX)
  NOTRUNCATE
  RESP(CMD-RESP)
END-EXEC.
IF CMD-INTO-TRAN NOT = EIBTRNID
  EXEC CICS RETURN
    TRANSID(CMD-INTO-TRAN)
    IMMEDIATE
END-EXEC
END-IF.
700-EXIT.
EXIT.
*
900-ERRORS.
EXEC CICS DUMP TASK
  DUMPCODE(EIBTRNID)
END-EXEC.
GO TO 950-RETURN.
900-EXIT.
EXIT.
*
950-RETURN.
EXEC CICS RETURN END-EXEC.
950-EXIT.
EXIT.
ADDITIONAL COMMENTS

To understand the parsing that takes place in XZCOUT$, it may help to refer to Figure 2. Here we see terminal output from a CEMT command to newcopy programs starting with ‘GLH’. The parsing logic keys on the word ‘NORMAL’ to the right of each successful newcopy, and stores the address of each instance to be used as a reference point by the format routine. In this example, programs starting with ‘GLHX’ could not be newcopied, so the parsing logic passes over them. Keyword ‘RESPONSE’ indicates that we have parsed the entire screen. If the operator had pressed PF3, we would detect ‘SESSION ENDED’, in which case there would be nothing more to log. The same is true of ‘SYNTAX ERROR’ (faulty keyword) and ‘ENTER ONE OF THE FOLLOWING’ (incomplete command).

I have included format routines for the following CEMT objects: programs, transactions, files, terminals, transient data queues, and tasks. These are the most commonly modified resources in our shop.

Figure 2: CEMT output screen
To add format routines for other CEMT objects, we would use TIOA dumps of CEMT output and code the appropriate MVC statements to move resource attributes into the log – a fairly tedious process. Without a corresponding format routine for its object, the CEMT command produces only a header record in the log. A point worth considering is that future CICS releases will probably mandate re-writes of the formatting logic.

Finally, a few words about the LOGWRITE program are perhaps also in order. LOGWRITE reads log records from the GWA buffer area and writes them to the CSMT transient data queue. It distinguishes header records from detail records by the presence of periods (full stops) in the header date.

After writing the records to transient data, LOGWRITE clears the GWA buffer area for reuse by XZCOUT$. LWRT, the transaction pointing to LOGWRITE, is set as the next trans-id (TCTTETC) in XZCOUT$, but also restarts itself every one minute to prevent the GWA buffer from becoming full before it is flushed. The code in 600-RESTART-THIS-TRAN ensures that only one instance of LWRT is queued to execute at any one time. The code in 700-EXECUTE-USER-TRAN ensures that any transaction entered by the user upon exiting CEMT will run immediately after LWRT.

Russell Hunt
Senior Systems Programmer
Great Lakes Higher Education Corporation (USA) © Xephon 1998

Why not share your expertise and earn money at the same time? CICS Update is looking for JCL, macros, program code, etc, that experienced CICS users have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2. Why not call now for a free copy of our Notes for contributors?
DL/I database display and control facility

THE PROBLEM
With Version 5.1, IMS no longer supports local DL/I in CICS. Consequently, anyone wanting to access DL/I databases from CICS must use DBCTL. With local DL/I there were CEMT commands for starting and stopping databases, but these are no longer available under DBCTL. A supplied transaction (CDBM) allows IMS commands to be issued to the DBCTL subsystem. This is fine if you’re accustomed to IMS commands and their responses, but is difficult if you are unaccustomed to their use.

THE SOLUTION
We decided to put together an easy-to-use facility to enable developers to display the status of their databases and to start and stop them in a manner similar to the way they could with CEMT.

The user is presented with a screen on which they enter the name of the database they want to display. The name can be generic – for example ‘DA*’ would result in a list of all databases beginning with DA, ‘DA*DB’ would produce a list of all those beginning with DA and ending with DB, and ‘*DB’ results in a list of those ending with DB. The user can then start or stop databases using simple line commands (S for Start and P for stop). This is illustrated in Figure 1. If there are more than 12 databases to display, the list can be scrolled using PF7 and PF8.

BEHIND THE SCENES
Sending commands to, and getting responses from, DBCTL is done by the IMS AIB (Application Interface Block) calls ICMD and RCMD. These are issued by calling AIBTDLI instead of ASMTDLI, but unfortunately there is no EXEC DLI interface available for AIB calls. In a CICS program, prior to issuing AIB calls you must schedule a PSB. In the case of DDDC we used the same PSB used by CDBM.
- some sample source is provided below. This PSB must be defined to the DBCTL subsystem.

Calls to start and stop databases are made via the program SPGDBSP, which issues the appropriate /START and /STOP IMS commands.

PROGRAM DIRECTORY
Programs used are:

- SPGDDDC – DL/I database display and control program.
- SPGDBSP – DL/I database start and stop program.
- DDDCM01 – BMS map for SPGDDDC.
- DFHDBMP – IMS PSB for inclusion in IMS GEN.
BIBLIOGRAPHY

CICS CSD DEFINITIONS
*
DEFINE MAPSET(DDDCMØ1) GROUP(DDDC)
   DESCRIPTION(MAPSET FOR DLI DATABASE DISPLAY AND CONTROL)
   RESIDENT(NO) USAGE(NORMAL) USELPACOPY(NO) STATUS(ENABLED)
*
DEFINE PROGRAM(SPGDBSP) GROUP(DDDC)
   DESCRIPTION(DLI DATABASE START AND STOP PROGRAM)
   LANGUAGE(ASSEMBLER) RELOAD(NO) RESIDENT(NO) USAGE(NORMAL)
   USELPACOPY(NO) STATUS(ENABLED) CEDF(YES) DATALOCATION(BELOW)
   EXECKEY(CICS) EXECUTIONSET(FULLAPI)
*
DEFINE PROGRAM(SPGDDDC) GROUP(DDDC)
   (DLI DATABASE DISPLAY AND CONTROL PROGRAM)
   LANGUAGE(ASSEMBLER) RELOAD(NO) RESIDENT(NO) USAGE(NORMAL)
   USELPACOPY(NO) STATUS(ENABLED) CEDF(YES) DATALOCATION(BELOW)
   EXECKEY(CICS) EXECUTIONSET(FULLAPI)
*
DEFINE TRANSACTION(DDDC) GROUP(DDDC)
   DESCRIPTION(DLI DATABASE DISPLAY AND CONTROL TRANS)
   PROGRAM(SPGDDDC) TWASIZE(Ø) PROFILE(DFHCICST) STATUS(ENABLED)
   TASKDATALOC(BELOW) TASKDATAKEY(CICS) STORAGECLEAR(NO)
   RUNAWAY(SYSTEM) SHUTDOWN(DISABLED) ISOLATE(YES) DYNAMIC(NO)
   PRIORITY(1ØØ) TRANCLASS(DFHTCLØØ) DTIMOUT(3Ø) INDOUBT(BACKOUT)
   RESTART(NO) SPURGE(YES) TPURGE(NO) DUMP(YES) TRACE(YES)
   CONFDATA(NO) RESSEC(NO) CMDSEC(NO)

IMS PSB
PSBGEN LANG=ASSEM,PSBNAME=DFHDBMP.IOASIZE=100Ø

SPGDDDC LISTING
*-------------------------------------------------------------------*
* 
* SPG D D D C 
*
* THIS PROGRAM PROVIDES A FACILITY FOR STARTING AND STOPPING DL/I DATABASES FROM WITHIN A CICS SYSTEM ATTACHED TO IMS DBCTL.

* ———————————————————*

DFHREGS
* ———————————————————*

* IMS AIB LAYOUT
* ———————————————————*

USING DFSAIB,R3
DFSAIB
* ———————————————————*

* DL/I UIB LAYOUT
* ———————————————————*

USING UIB,R4
DLIUIB
* ———————————————————*

* WORKING STORAGE
* ———————————————————*

DFHEISTG
*  
ABSTIME DS D
LENGTH DS H FOR T/S IO
ITEM DS H FOR T/S IO
PREFIX_LENGTH DS H
SUFFIX_LENGTH DS H
*
COMMAREA DS ØC
ITEMS DS H ITEMS IN TS QUEUE
START_AT DS H CURRENT STARTING POSITION
ENTERED_DBNAME DS CL8
COMMAREL EQU *-COMMAREA
*
CALLLIST CALL ,(*,*),MF=L
*

UIBPTR DS F UIB POINTER
*
RESPONSE DS F RESP
REASON DS F RESP2
*
AIB_CMD DS CL4 COMMAND FOR AIB CALL
*
IOAREA   DS    CL136                   IO AREA FOR AIB CALL
LIOAREA  EQU   *-IOAREA
           ORG  IOAREA
IOLEN    DS    CL4                     LENGTH OF TEXT
IOTEXT   DS    CL132                   TEXT
           ORG
*  
AIBAREA  DC    (AIBLL)X'00'            RESERVE SPACE FOR AIB
*  
TSQNAME  DS    CL8                     TEMP STORAGE QUEUE NAME
           ORG  TSQNAME
TSQTERM  DS    CL4                     TERMINAL-ID
TSQTRAN  DS    CL4                     TRANS-ID
*  
OUTCOMES DS    12CL11                  OUTCOME OF ANY LINE COMMANDS
*  
LAST_SEGMENT DS CL1                   LAST MESSAGE SEGMENT ENCOUNTERED
HAD_A_COMMAND DS CL1                   LINE COMMAND INDICATOR
*  
DBNAME          DS CL8                 DBNAME FROM MESSAGE
DBNAME_REVERSED DS CL8                 DBNAME REVERSED
PREFIX      DS CL8                     REQUIRED PREFIX
SUFFIX      DS CL8                     REQUIRED SUFFIX
WORK_AREA   DS CL8                     WORK AREA
*  
*  PARAMETERS FOR CALLING SPGDBSP  
*  
SPGDBSP_PARMS  DS ØCL1Ø
FUNCTION DS    CL8
DATABASE DS    CL8
RESULT   DS    CL1
*  
*  MAP  
*  
*  COPY  DDDCMØ1                
*  
*  HERE WE GO  
*  
SPGDDDC  DFHEIENT EIBREG=11,CODEREG=12,DATAREG=13
*  
MVC   TSQTERM,EIBTRMID        SET UP ..... 
MVC   TSQTRAN,EIBTRNID        ....TS QUEUE NAME
*  
CLC   EIBCALEN,ZERO           HAVE WE GOT A COMMAREA
```
BNE NOT_FIRST_TIME YES - NOT FIRST TIME THROUGH
*
BAL R10,CLEAR_MAP
EXEC CICS SEND MAP('DDDCM01') ERASE FREEKB FROM(DDDCM01O)
LA R1,1 SET TO 1
STH R1,START_AT SET STARTING POSITION
MVC ITEMS,ZERO SET NO OF ITEMS
LA R1,COMMAREL GET COMMAREA LENGTH
STH R1,LENGTH SAVE IT
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA) LENGTH(LENGTH)
*
NOT_FIRST_TIME DS ØH
L R1,DFHEICAP GET COMMAREA ADDRESS
MVC COMMAREA(COMMAREL),Ø(R1) GET COMMAREA
EXEC CICS HANDLE AID ENTER(PROCESS) CLEAR(PUNTER_WANTS_OUT) X
PF3(PUNTER_WANTS_OUT) PF15(PUNTER_WANTS_OUT) X
PF7(PAGE_BACK) PF19(PAGE_BACK) X
PF8(PAGE_FORWARD) PF20(PAGE_FORWARD) X
ANYKEY(ANYTHING_ELSE)
EXEC CICS RECEIVE MAP('DDDCM01') MAPSET('DDDCM01') INTO(DDDCM01I)
*
ANYTHING_ELSE DS ØH
LA R2,DDDCM01O POINT AT RECEIVING AREA
LA R3,DDDCM01L SET ITS LENGTH
XR R4,R4 SET DUMMY FROM ADDRESS
XR R5,R5 SET DUMMY FROM ADDRESS
MVCL R2,R4 BLANK OUT THE AREA
MVC MESSAGEO,WRONG_KEY
MVC DBNAMEO,ENTERED_DBNAME
EXEC CICS SEND MAP('DDDCM01') FROM(DDDCM01O) DATAONLY FREEKB
LA R1,COMMAREL
STH R1,LENGTH
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA) LENGTH(LENGTH)
*-----------------------------------------------------------------------*
* * * PROCESS ANY INPUT * *
* * *-----------------------------------------------------------------------*
PROCESS DS ØH
*
MVI HAD_A_COMMAND,C'N' SET COMMAND INDICATOR
LA R2,12 SET NUMBER OF LINES
LA R3,CMD1L GET ADDRESS OF 1ST COMMAND LEN
XR R4,R4 SET TO Ø
LA R5,OUTCOMES
CMD_LOOP DS ØH
MVC Ø(11,R5),SPACES
```
* NOT A LINE COMMAND SO CHECK FOR INPUT IN THE DBNAME
*

CHECK_FOR_DBNAME DS ØH
MVC PREFIX_LENGTH,ZERO ZERO OUT BEGINNING STRING LEN
MVC SUFFIX_LENGTH,ZERO ZERO OUT ENDING STRING LEN
MVC PREFIX,SPACES SPACE OUT BEGINNING STRING
MVC WORK_AREA,SPACES SPACE OUT ENDING STRING
MVC ENTERED_DBNAME,DBNAMEI SAVE ENTERED DBNAME
*

CLC DBNAMEL,ZERO HAS ANY DBNAME BEEN ENTERED
BNE GOT_SOMETHING YES - GO AND BUILD TS QUEUE
MVC PREFIX_LENGTH,ALL DATABASES SET TO ALL
B GO_TO_IT GO AND PROCESS
GOT_SOMETHING DS ØH
CLC DBNAMEI,STAR IS DBNAME JUST *
BNE NOT_STAR NO - GO AND PROCESS IT
MVC PREFIX_LENGTH,ALL DATABASES SET TO ALL
B GO_TO_IT GO AND GET INFO
NOT_STAR DS ØH
*
* HERE WE DO THE PREPARATION FOR GENERIC DISPLAYS
*
* WE START AT THE BEGINNING OF THE DBNAME AND LOOK FOR *, MOVING
* EACH BYTE TO THE PREFIX FIELD.
*
* IF AND WHEN WE HIT A * WE POINT PAST IT THEN MOVE EACH SUBSEQUENT
* BYTE TO THE WORK AREA.
*
* ONCE WE'VE EXAMINED EACH BYTE OF THE DBNAME WE TAKE WHAT'S IN THE
* WORK AREA AND REVERSE IT INTO THE SUFFIX FIELD
*
* EG IF THE DBNAME ENTERED WAS BK*AN.
*
* AT THE END OF THIS PROCESS THE FIELDS WOULD CONTAIN:
*
* PREFIX : BK
* WORK   : AN
* SUFFIX : NA
*
LA R2,DBNAMEI GET ADDRESS OF DBNAME
LH R3,DBNAMEL GET LENGTH OF DBNAME
XR R4,R4 CLEAR PREFIX LENGTH
XR R5,R5 CLEAR SUFFIX LENGTH
LA R6,PREFIX GET ADDRESS OF PREFIX AREA
LA R7,WORK_AREA GET ADDRESS OF WORK AREA
LOOP_1 DS ØH
CLI Ø(R2),C' ' END OF DBNAME?
BE END_OF_DBNAME YES - NO MORE TO DO
CLI Ø(R2),C'*' STAR ?
BE END_OF_PREFIX YES - END OF PREFIX
STH R2,ITEM       SET ITEM
LA R1,132        SET MAX LENGTH
STH R1,LENGTH AND SAVE IT
EXEC CICS READQ TS QUEUE(TSQNAME) INTO(IOTEXT) ITEM(ITEM)
          RESP(RESPONSE) LENGTH(LENGTH)
CLC RESPONSE,DFHRESP(NORMAL) READ OK?
BE GOT_ITEM YES - MOVE DATA TO MAP
B SEND_DETAIL_MAP NO - GO AND SEND THE MAP

GOT_ITEM DS ØH
MVC NAME1O-CMD1L(L'NAME1O,R4),IOTEXT+4
MVC RESULT1O-CMD1L(L'RESULT1O,R4),Ø(R5)
LH R1,LENGTH GET RECORD LENGTH
LA R6,23 GET ADDRESS OF STATUS
SR R1,R6 FIND LENGTH OF STATUS
EX R1,MOVE_STATUS GET STATUS
LA R4,CMD2L-CMD1L(R4) POINT TO NEXT DBNAME
LA R2,1(R2) ADD 1 TO ITEM
LA R5,11(R5) POINT TO NEXT OUTCOME
BCT R3,READ_TSQ_ITEM GO AND GET THE NEXT ONE

SEND_DETAIL_MAP DS ØH
MVC DBNAMEO,ENTERED_DBNAME
EXEC CICS SEND MAP('DDDCM01') FROM(DDDCM01O) ERASE FREEKB
*
LA R1,COMMAREL
STH R1,LENGTH
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA)
          LENGTH(LENGTH)
*
MOVE_STATUS MVC STATUS1O-CMD1L(Ø,R4),IOTEXT+22
*
*——————————————————————————————————*
*                                                                     *
*        PAGE BACK                                                   *
*                                                                     *
*——————————————————————————————————*
*  PAGE_BACK DS ØH
LH R2,START_AT GET START POINT
BCTR R2,Ø -1
LTR R2,R2 ZERO ?
BNZ CAN_PAGE_BACK NO - WE CAN PAGE BACK
BAL R1,CLEAR_MAP
MVC MESSAGEO,CANT_PAGE_BACK
MVC DBNAMEO,ENTERED_DBNAME
EXEC CICS SEND MAP('DDDCM01') FROM(DDDCM01O) DATAONLY FREEKB
LA R1,COMMAREL
STH R1,LENGTH
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA)
          LENGTH(LENGTH)
CAN_PAGE_BACK DS ØH
LA R3,11 SET TO 11
SR    R2,R3                        SUBTRACT
STH   R2,START_AT                  SAVE NEW START POINT
B     BUILD_THE_DISPLAY

* ———————————————————————————————————*

* *
* PAGE FORWARD *
* *
* ———————————————————————————————————*

PAGE_FORWARD DS ØH
LH    R2,START_AT                  GET START POINT
LA    R2,12(R2)                     ADD 12
CH    R2,ITEMS                      TOO MANY
BNH   CAN_PAGE_FWD                  NO - WE CAN PAGE FORWARD
BAL   R10,CLEAR_MAP
MVC   MESSAGEO,CANT_PAGE_FWD
MVC   DBNAMEO,ENTERED_DBNAME
EXEC CICS SEND MAP('DDDCMØ1') FROM(DDDCMØ1O) DATAONLY FREEKB
LA    R1,COMMAREL
STH   R1,LENGTH
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA)     X
LENGTH(LENGTH)

CAN_PAGE_FWD DS ØH
STH   R2,START_AT                  SAVE NEW START POINT
B     BUILD_THE_DISPLAY

* *
PUNTER_WANTS_OUT DS ØH
EXEC CICS DELETEQ TS QUEUE(TSQNAME) RESP(RESPONSE)
EXEC CICS SEND CONTROL ERASE FREEKB
EXEC CICS RETURN

* ———————————————————————————————————*

* DISPLAY THE REQUESTED DATABASES *
* ———————————————————————————————————*

DO_THE_DISPLAY DS ØH
* *
* INITIALIZE THE AIB *
* *
* ———————————————————————————————————*

LA    R3,AIBAREA
MVC   AIBID,'=CL8'DFSAIB'            INITIALIZE ...
MVC   AIBLEN,'=A(AIBLL)'             .. DFSAIB ...
MVC   AIBOALEN,'=A(LIOAREA)'         .. CONTROL BLOCK

* *
* DO THE PCB CALL *
* *
* ———————————————————————————————————*

CALL ASMTDLI, X
(PCB,PSBNAME,UIBPTR,SYSSERVE),  X
VL,  X
MF=(E,CALLLIST)
L R4,UIBPTR               GET UIB ADDRESS
*  EXEC CICS ENTER TRACENUM(1) FROM(UIB) FROMLENGTH(UIB_LEN)  X
   RESOURCE('SPGBDSP') RESP(RESPONSE) RESP2(REASON)
*  CLI UIBFCTR,X'ØØ'           CHECK RETURN CODE
BE PSB_SCHEDULED              ZERO - WE'RE OK
BAL R10,CLEAR_MAP
MVC MESSAGEO,PSB_SCHED_ERROR
MVC DBNAMEO,ENTERED_DBNAME
EXEC CICS SEND MAP('DDDCMØ1') FROM(DDDCMØ1O) DATAONLY FREEKB
LA R1,COMMAREL
STH R1,LENGTH
EXEC CICS RETURN TRANSID(EIBTRNID) COMMAREA(COMMAREA)  X
   LENGTH(LENGTH)
*——————————————————————————————————*
*                                                                     *
*        SET UP THE IO AREA FOR THE AIB CALL                          *
*                                                                     *
*——————————————————————————————————*
PSB_SCHEDULED DS ØH
  MVC IOLEN,=Y(L'IOTEXT,Ø)    SET COMMAND LENGTH
  MVI IOTEXT,C'
  MVC IOTEXT+1(L'IOTEXT-1),IOTEXT
*——————————————————————————————————*
*                                                                     *
*        ISSUE THE AIB CALL FOR /DIS                                  *
*                                                                     *
*——————————————————————————————————*
  MVI LAST_SEGMENT,C'N'       SET LAST SEGMENT FLAG
  MVC IOTEXT(L'CMDDIS),CMDDIS MOVE IN /DIS COMMAND
  MVC AIB_CMD,ICMD            SET AIB COMMAND TO ICMD
CALL_AIB_FOR_DISPLAY_COMMAND DS ØH
  BAL R9,ISSUE_AIB_COMMAND
*——————————————————————————————————*
*                                                                     *
*        ACT UPON THE RETURN CODE FROM THE AIB CALL                   *
*                                                                     *
*——————————————————————————————————*
  B CHECK_DISPLAY_RETURN_CODE(R15)
CHECK_DISPLAY_RETURN_CODE DS ØH
  B PARSE_MESSAGE_FROM_DISPLAY
  B LAST_SEGMENT_RETURNED
  B TERM_PSB
  B TERM_PSB
LAST_SEGMENT_RETURNED DS ØH
  MVI LAST_SEGMENT,C'Y'       SET LAST SEGMENT FLAG
Editor’s note: this article will be continued next month.

Kevin Wailes
Automatic change from CSSN to CESN

INTRODUCTION
When migrating from CICS/MVS Version 2.1.2 to any later version of CICS for MVS, certain transactions are discontinued (eg CSMT, CSSF, CSSN, CSST, and CSOT).

The most inconvenient change for our company was that for CSSF – because we had a lot of client systems that used CSSF LOGOFF and we were not able to change all our client systems.

SOLUTION
Our solution was to enable the user exit XZCATT – which is invoked before task attach for terminal tasks. This exit allows you to continue using the CSSF transaction and leave the client systems unchanged.

USER EXIT CESNZCAT
***********************************************************************
* MODULE NAME = CESNZCAT
* User exit program for before task attach (XZCATT)
*
* FUNCTION =
* This user exit program is to be invoked at the
* XZCATT global user exit point when processing task attach.
* It changes transaction CSSF to CESF.
*
***********************************************************************
SPACE
RØ EQU Ø NOT USED
R1 EQU 1 INITIAL USER EXIT PARAMETER LIST
R2 EQU 2 USER EXIT PARAMETER LIST
R3 EQU 3 UEPTRAN pointer
R4 EQU 4 NOT USED
R5 EQU 5 NOT USED
R6 EQU 6 NOT USED
R7 EQU 7 NOT USED
R8 EQU 8 NOT USED
R9 EQU 9 NOT USED
R10 EQU 10 NOT USED
R11 EQU 11 NOT USED
R12 EQU 12 PROGRAM BASE
R13 EQU 13 SAVE AREA
R14 EQU 14 RETURN ADDRESS
R15 EQU 15 INITIAL PROGRAM BASE
EJECT
DFHUEXIT TYPE=EP,ID=(XZCATT)
EJECT
CESNZCAT CSECT
CESNZCAT AMODE 31
CESNZCAT RMODE ANY
SAVE (14,12) SAVE REGS
LR R12,R15 SET-UP BASE REGISTER
USING CESNZCAT,R12 ADDRESSABILITY
LR R2,R1 GET UEP PARAMETER LIST
USING DFHUEPAR,R2 ADDRESSABILITY
SPACE
L R3,UEPTRAN GET ADDRESS OF INITIAL TRANSACTION
LTR R3,R3
BZ RETURN
CLC Ø(4,R3),=CL4'CSSF' If tran=cssf
BNE RETURN then
MVC Ø(4,R3),=CL4'CESF' tran:=cesf
SPACE
RETURN DS ØH RETURN TO THE CALLER
L R13,UEPEPSA ADDRESS OF EXIT SAVE AREA
RETURN (14,12),RC=UERCNORM RESTORE REGS AND RETURN
SPACE
LTORG
SPACE
END CESNZCAT

The following resource definition is required for the exit program CESNXZAT:

PROGram : CESNZCAT
Group : YOURGRP
DEscription ==> What ever you like
Language ==> Assembler

RELoad ==> No
RESident ==> No
USAge ==> Normal
USElpacopy ==> No
Status ==> Enabled
RSI : ØØ
Cedf ==> Yes
DATalocation ==> Any
EXECKey ==> CICS
The user exit is enabled using the following CICS API statement. Add this program to the PLTPI and make an RDO definition to enable the program:

```
EXEC CICS
   ENABLE PROGRAM('CESNZCAT') EXIT('XZCATT') START
   RESP (RESPONSE1)
END-EXEC.
```

You can change the transaction-id in any transaction you require.

---

Paul Jansen  
Systems Programmer  
Interpay (The Netherlands)  
© Xephon 1998

---

**More on macros to define statements – part 2**

This month we conclude the two-part article which is a continuation of Converting macros to define statements, published in CICS Update, Issue 147, February 1998 and Issue 148, March 1998. It provides an additional macro that eliminates all VSAM entries from an FCT. It also contains a program that merges CSD define statements.

```
***********************************************************************
***                                                                 ***
***   THIS ROUTINE CHECKS ALL POTENTIAL DEFINE STATEMENTS FOR A       ***
***   GROUP DEFINITION. IF A GROUP IS FOUND, ITS (NON-DUPLICATE)      ***
***   NAME IS FOUND IN 'GROUPS' AND USED TO BUILD ADD STATEMENTS      ***
***   TO ADD THOSE GROUPS TO LIST 'INITLIST'.                        ***
***                                                                 ***
***********************************************************************

DOGROUP  ST    RBAL,SAVDGBAL       SAVE LINKAGE REGISTER

CLI   OUTAREA,C'**'        COMMENT?
BE    DGRETURN             NO

LA    R14,OUTAREA         POINT TO START OF IMAGE TO SCAN
LA    R15,72             STATEMENT LENGTH
```
DGLOOP   EX    R15,DGTRT1          SCAN FOR FIRST NON-BLANK
  BZ    DGRETURN            EXIT IF NONE FOUND

  LR    R0,R1               ADDRESS OF NON-BLANK
  SR    R0,R14              NUMBER OF BLANKS SKIPPED
  LR    R14,R1              POINT TO NON-BLANK

  CLC   =C'DELETE ',Ø(R1)   DELETE STATEMENT?
  BE    DGRETURN            YES

  CLC   =C'ADD ',Ø(R1)      ADD STATEMENT?
  BE    DGRETURN            YES

  CLC   =C'GROUP(',Ø(R1)    GROUP DEFINITION?
  BNE   DGLOOPNG             NO

  LA    R14,6(R1)           POINT TO 1ST BYTE OF GROUP NAME
  LA    RØ,OUTAREA+71       GET ENDING ADDRESS
  SR    RØ,R14              SUBTRACT CURRENT POSITION
  SR    R15,RØ              SAVE REMAINING LENGTH
  BNP   GNRETURN            EXIT AS STATEMENT END

  LA    R1,OUTAREA+72       POINT PAST STATEMENT
  EX    R15,DGTRT2          FIND FIRST BLANK
  SR    R1,R14              LENGTH OF GROUP NAME WITH ')
  BNP   DGRETURN            EXIT IF NOT POSITIVE
  BCTR  R1,Ø                LENGTH - 1

  LA    R2,GROUPS-L'GROUPS  ADDRESS OF ENTRY(-1)
  B     DGNMNEXT            DO LOOP 'TIL END

  DGNMLOOP EX    R1,DGCLC            NAME ALREADY ENTERED?
  BE    DGRETURN            YES

  DGNMNEXT LA    R2,L'GROUPS(R2)     POINT TO CURRENT POSITION
  C     R2,GROUPLOC         END OF PREVIOUS ENTRIES?
  BNH   DGNMLOOP             NO

  LA    R0,GROUPEND         ADDRESS OF LAST AVAILABLE SPACE
  CR    R2,R0               SPACE FULL?
  BH    DGNOROOM            YES

  ST    R2,GROUPLOC         SAVE CURRENT POSITION

  MVC    Ø(L'GROUPS,R2),LINE+1 CLEAR TO BLANKS
  EX    R1,DGMOVE           MOVE NAME AND ) TO TABLE
  B     DGRETURN            EXIT

  DGNOROOM MVC    LINE(21),=C'GROUP NAME OVER FLOW' SET WARNING
  MVC    LINE+22(L'OUTAREA),OUTAREA SET STATEMENT TEXT
BAL RBAL,PRINT PRINT ERROR MESSAGE
BAL RBAL,DOUBLESP ALLOW FOR DOUBLE SPACE
B DGRETURN
*
DGLOOPNG SR R15,R0 SET REMAINING LENGTH
BNP DGRETURN EXIT IF DEPLETED
EX R15,DGTRT2 SCAN FOR NEXT BLANK
BZ DGRETURN EXIT IF NONE FOUND
*
LR R0,R1 POINT TO BLANK
SR R0,R14 NUMBER OF NON-BLANKS
LR R14,R1 POINT TO BLANK
SR R15,R0 ADJUST REMAINING SCAN LENGTH
BP DGLOOP UNLESS NULL, CONTINUE SEARCH
*
DGRETURN L RBAL,SAVDBGAL RESTORE LINKAGE REGISTER
BR RBAL RETURN
*
DGTRT1 TRT Ø(*-*,R14),TRTAB1
DGTRT2 TRT Ø(*-*,R14),TRTAB2
DGCLC CLC Ø(*-*,R2),Ø(R14)
DGMOVE MVC Ø(*-*,R2),Ø(R14)
*
EJECT
*******************************************************************************
***                                                                 ***
***   THIS ROUTINE PROCESSES EXTRACTED GROUP NAMES                      ***
***                                                                 ***
*******************************************************************************
*
PUTGROUP ST RBAL,SAVPGBAL SAVE LINKAGE REGISTER
*
STM R3,R5,PGSAVE35 SAVE REGISTERS
*
LA R3,GROUPS POINT TO FIRST GROUP
LA R4,L'GROUPS SIZE OF VECTOR
L R5,GROUPLOC GET CURRENT ADDRESS
*
CR R3,R5 VECTOR EMPTY?
BNL PGFINISH YES
*
MVC LINE(40),=C'Ø*** THE FOLLOWING ADD STATEMENTS ADDED:' BAL RBAL,PRINT PRINT ABOVE
MVC RBAL,DOUBLESP ALLOW FOR DOUBLE SPACE
*
MVC OUTAREA,line+1 CLEAR TO BLANKS
MVC OUTAREA+1(I0),=C'ADD GROUP(' BUILD ADD STATEMENT
MVC OUTAREA+12+L'GROUPS(L'LISTNAME),LISTNAME SET LIST(...) PGLOOP MVC OUTAREA+11(L'GROUPS),Ø(R3) INSERT GROUP NAME
BAL RBAL,WRITEREC WRITE ADD COMMAND
MVC   LINE+5(L'OUTAREA),OUTAREA MOVE ADD COMMAND TO PRINT LINE
BAL   RBAL,PRINT         PRINT ADD COMMAND
BXLE  R3,R4,PGLOOP       CONTINUE

*  
LA    R3,GROUPS-L'GROUPS ENTRY(-2)
ST    R3,GROUPLOC        REINITIALIZE

*  
PGFINISH  LM   R3,R5,Pgsave35  RESTORE REGISTERS

*  
L      RBAL,SAVPGBAL      RESTORE LINKAGE REGISTER
BR     RBAL               RETURN

*  
* END STUB DEFINE
*  
*  
EJECT

***********************************************************************
***  ERROR RETURNS  ***
***
***********************************************************************

ERROR    STH   RØ,COMPCODE         SET COMPLETION CODE

BAL  RBAL,HEADPAGE        EJECT PAGE

BAL  RBAL,PRINT           PRINT ERROR MESSAGE

B     ENDING              GO EXIT

EJECT

***********************************************************************
***  PRINT ROUTINE  ***
***
***********************************************************************

PRINT    PUT   PRINTER,LINE        PRINT LINE
MVI   LINE,C' '           SET SEED
MVC   LINE+1(L'LINE),LINE CLEAR LINE
DOUBLESP BCTR  R9,RBAL   RETURN IF PAGE NOT FULL

HEADPAGE MVC    PAGENO.=X'4Ø2Ø212Ø' SET EDIT PATTERN
ED      PAGENO,PAGES    FORMAT PAGE NUMBER
AP      PAGES,.=P'1'    INCREMENT PAGE COUNT
PUT     PRINTER,HEADER  PRINT PAGE HEADING
LA      R9,56           SET LINES/PAGE
MVI   LINE,C'Ø'         SET TO DOUBLE SPACE AFTER HEADER
BR     RBAL              RETURN

EJECT
WORKD DSECT
MYSAVE DS 18F MY REGISTER SAVE AREA
COMPCODE DS F PROGRAM COMPLETION CODE
RETCDE DS F INTERNAL RETURN CODE
R1SAVE DS F INITIAL VALUE IN R1
PAGES DS PL2
DOUBLE DS D
IN1DSN DS CL54
IN2DSN DS CL54
DDNAME DS CL8
COUNT1 DS PL3
COUNT2 DS PL3
IN1AREA DS CL80
IN2FLAG DS C
IN2AREA DS CL80
OUTAREA DS CL80
PASSFLAG DS C
ADEFSAVE DS A
DS F MUST FOLLOW ADEFSAVE
LDEFSAVE DS F MUST FOLLOW ADEFSAVE WITH DISP=8
DUPS DS PL3
TIME DS F
DUPFLAG DS C
TRTAB1 DS CL256
TRTAB2 DS CL256
MEMBER DS CL8
GROUP1 DS F
GROUPLOC DS A
PGSAVE35 DS 3F
*
* BEGIN STUB LINK SAVE
*
SAVJGBAL DS A BAL REGISTER SAVE AREA FOR JULGREG
SAVGNBAL DS A BAL REGISTER SAVE AREA FOR GETNAMES
SAVLGBAL DS A BAL REGISTER SAVE AREA FOR LOGOUT
SAVCRBAL DS A BAL REGISTER SAVE AREA FOR COUNTREC
SAVC1BAL DS A BAL REGISTER SAVE AREA FOR COPYIN1
SAVC2BAL DS A BAL REGISTER SAVE AREA FOR COPYIN2
SAVWRBAL DS A BAL REGISTER SAVE AREA FOR WRITEREC
SAVDTBAL DS A BAL REGISTER SAVE AREA FOR DOTOTALS
SAVCJBAL DS A BAL REGISTER SAVE AREA FOR COPYJCL
SAVDGBAL DS A BAL REGISTER SAVE AREA FOR DOGROUP
SAVPGBAL DS A BAL REGISTER SAVE AREA FOR PUTGROUP
*
* END STUB LINK SAVE
*
*
* BEGIN OPEN/CLOSE LIST
*
DS ØD
* PROPENL OPEN (,),MF=L
PROPENLN EQU *-PROPENL
PRCLOS NL CLOSE (,),MF=L
PRCLOS LN EQU *-PRCLOS L
*
I1OPENL OPEN (,),MF=L
I1OPENLN EQU *-I1OPENL
I1CLOSL CLOSE (,),MF=L
I1CLOSLN EQU *-I1CLOSL
*
I2OPENL OPEN (,),MF=L
I2OPENLN EQU *-I2OPENL
I2CLOSL CLOSE (,),MF=L
I2CLOSLN EQU *-I2CLOSL
*
OPOPENL OPEN (,),MF=L
OPOPENLN EQU *-OPOPENL
OPCLOS L CLOSE (,),MF=L
OPCLOS LN EQU *-OPCLOS L
*
* END OPEN/CLOSE LIST
*
*
* BEGIN DCB DSECTS
*
PRINTER DCB DDNAME=PRINTER,DEVD=DA,DSORG=PS,LRECL=133, -
BLKSIZE=133,MACRF=(PM),RECFM=FBA
PRINTERL EQU *-PRINTER
*
INPUT1 DCB DDNAME=INPUT1,DSORG=PS,MACRF=GM,EODAD=I1EOF
INPUT1L EQU *-INPUT1
*
INPUT2 DCB DDNAME=INPUT2,DSORG=PS,MACRF=GM,EODAD=I2EOF
INPUT2L EQU *-INPUT2
*
OUTPUT DCB DDNAME=OUTPUT,DSORG=PS,MACRF=PM
OUTPUTL EQU *-OUTPUT
*
* END DCB DSECTS
*
JGMOTBL DS PL2'Ø'
JANUARY DS P'31'
*                    M  A  M  J  J  A  S  O  N
FEBRUARY DS P'28,31,30,31,30,31,31,30,31,30'
DECEMBER DS P'31'
JGDAYS DS PL2
JGMONTHS DS PL2
JGMMDYY DC C'MM/DD/YY'
JGYYDDD DS F
* END DSECT INSERT
HEADER DS CL133
  ORG HEADER+L'HEAD+5
HEADJOBN DS CL8,C' DSN='
HEADDRS SN DS CL44,2C
HEADDRATE DS CL8
HEADTIME DS CL(LTIMEPAT)
  ORG HEADER+L'HEADER-5
PAGENO DS CL4
  ORG
LINE DS CL133,C
ERRORMSG EQU LINE
*
GROUPS DS 100CL9
GROUPEND DS CL(L'GROUPS)
*
  DS ØD
WORKDLEN EQU *-WORKD
*
* THIS AREA IS OBTAINED TO STORE A TABLE OF NEWLY DEFINED TYPE/ENTRY
* NAMES (EG TRANSACTION(TRANSID), PROGRAM(PROGRAMID), ETC).
* DEFSTACK DSECT
DEFINE DS CL50
*
* SYSTEM DSECTS
*
  IHAPSA MAP OF PSA DSECT=PSA
  IKJTCB MAP OF TCB DSECT=TCB
TIOT DSECT
  IEFtioti MAP OF TIOT
  CVT DSECT=YES MAP OF CVT DSECT=CVTMAP
JFCB DSECT MAP OF JFCB
JFCBPREF DS CL16 PREFIX
  IEFJFCBN LIST=NO JFCB PROPER
*
  DCBD DSORG=PO,DEVD=DA A.T.
*
  EJECT
*******************************************************************************
***
*** REGISTER EQUATES
***
*******************************************************************************
*
RØ EQU 0
R1 EQU 1
R2 EQU 2
R3 EQU 3
R4 EQU 4
R5 EQU 5
DFHMACS2.FG1

RDOMERGE — MERGE RDO DEFINITIONS  JOB=SYST002L
DSN=SYST002.FILE.TRANSFER

INPUT FILES ARE:  SYST002.TEST.PDS2(KMFCTRDO)
AND:  SYST002.TEST.PDS2(RDOFILE)

***********************************************************************
*                                                                     *
* RECORD  8631 DUPLICATE, SCRATCHED  DEFINE FILE(KMDRGMD) GROUP(MP3FCT)
* RECORD  8638 DUPLICATE, SCRATCHED  DEFINE FILE(KMCODMD) GROUP(MP3FCT)
* RECORD  8645 DUPLICATE, SCRATCHED  DEFINE FILE(KMMRMMD) GROUP(MP3FCT)
* RECORD  8652 DUPLICATE, SCRATCHED  DEFINE FILE(KMNMEMD) GROUP(MP3FCT)
* RECORD  8659 DUPLICATE, SCRATCHED  DEFINE FILE(KMMRNPD) GROUP(MP3FCT)
* RECORD  8665 DUPLICATE, SCRATCHED  DEFINE FILE(KMADMPD) GROUP(MP3FCT)
* RECORD  8671 DUPLICATE, SCRATCHED  DEFINE FILE(KMDISPD) GROUP(MP3FCT)

*** THE FOLLOWING ADD STATEMENTS ADDED:
   ADD GROUP(MP3FCT)  LIST(INITLIST)
   ADD GROUP(R3Ø#TRW) LIST(INITLIST)
   ADD GROUP(R3Ø#KP)  LIST(INITLIST)
   ADD GROUP(MP3PPT)  LIST(INITLIST)
   ADD GROUP(MP3PCT)  LIST(INITLIST)

   RECORDS FROM INPUT1     9Ø
   RECORDS FROM INPUT2  8681

DUPLICATE DEFINE STATEMENTS     7

Keith H Nicaise
Technical Services Manager
Touro Infirmary (USA)  © Xephon 1998
Software Diversified Services has announced Inter-Program Command Processor (IPCP-Plus), a file control utility that keeps track of CICS resources, even when CICS is down, so that current batch-side status can be maintained. Both batch and CICS sides of IPCP-Plus have built-in security exit points.

With the IPCP-Plus on-line facility, any IPCP-Plus command can be initiated from a CICS terminal and users can determine who closed a particular file, when it was closed, and its current status.

For further information contact:
Software Diversified Services, 5155 East River Road, Minneapolis, MN 55421-1025, USA.
Tel: (612) 571 9000.

* * *

Dynasty Technologies has announced support for the CICS Transaction Server for OS/390, and Java and Enterprise Java Beans support, with its development environment.

The product will be able to generate component-based native CICS applications that will support CICS and DB2 as well as MQSeries. Support for Beans means Dynasty applications will run on all IBM application servers, including Component Broker, TXSeries, and WebSphere. In addition, Dynasty will integrate its Java and Web server capabilities with OS/390 and the Domino Go Web server.

With the upgraded Dynasty, CICS users will be able to generate CICS applications without worrying about the CICS APIs. Dynasty developers will be able to move applications, components, and middleware across platforms and generate native code without changing the application specification.

For further information contact:
Dynasty Technologies, 101 Redwood Shores Parkway, Suite 200, Redwood Shores, CA 94065, USA.
Tel: (650) 631 5430.

* * *

CICS users can now benefit from Sterling Software’s VISION:Phaseshift tool, designed to ‘insulate’ MVS applications from year 2000 date problems, without the need to change applications. The tool does this by encapsulating code and data and shifting dates dynamically so that all dates processed fall in the same century. In addition to CICS, products supported include QSAM, VSAM, IMS, DB2, and TSO.

For further information contact:
Sterling Software, 1800 Alexander Bell Drive, Reston, VA 22091, USA.
Tel: (703) 264 8000.
Sterling Software, 1 Longwalk Road, Stockley Park, Uxbridge, Middlesex, UB11 1DB.
Tel: (0181) 867 8000.