178 CICS update

September 2000

In this issue

CICS Update

Published by

Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: trevore@xephon.com

North American office

Xephon
PO Box 350100
Westminster, CO 80035-0100
USA
Telephone: 303 410 9344

Subscriptions and back-issues

A year's subscription to *CICS Update*, comprising twelve monthly issues, costs £175.00 in the UK; $270.00 in the USA and Canada; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £16.00 ($23.50) each including postage.

CICS Update on-line

Code from *CICS Update* can be downloaded from our Web site at http://www.xephon.com/cicsupdate.html; you will need the user-id shown on your address label.

Editor

Trevor Eddolls

Disclaimer

Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, and other contents of this journal before making any use of it.

Contributions

Articles published in *CICS Update* are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*, or you can download a copy from www.xephon.com/contnote.html.

Printed in England.

Rununits, enclaves, and storage management under CICS/390

A CICS rununit is instantiated to execute the 'initial_link' to run the top-level program in a CICS transaction and also whenever the application executes an EXEC CICS LINK command. On each link, including the initial link, CICS calls Language Environment, provided that the application is deemed to be 'LE-enabled'.

An application is considered to be LE-enabled when it has either been compiled with an 'LE-conforming' compiler or been prepared with a compiler such as VS COBOL II, which Language Environment is able to support in 'compatibility' mode. On the first call to Language Environment to run an application program in a transaction, which is known as thread_initialization, Language Environment creates 'thread-level' or 'process-level' control blocks. These are functionally equivalent to the CICS TCA (Task Control Area) and associated control blocks such as the EIB, EIS, and EIUS.

For each link level (ie rununit instance), including the initial link level, Language Environment creates an 'enclave'. An enclave and its associated control blocks can be considered as Language Environment's instantiation of a rununit. CICS calls Language Environment with 'rununit_initialization' to create an enclave. Before this call, CICS has pre-allocated one or more rununit storage areas (the sizes of which are determined and remembered when an application is loaded into the CICS address space), and the addresses of these storage areas are passed to Language Environment on the rununit_initialization call.

When a rununit terminates, which normally means that an application program has issued an EXEC CICS RETURN or a native language return such as a COBOL GOBACK statement, both the CICS rununit and the Language Environment enclave control blocks are disposed of and the storage areas holding these blocks are FREEMAINed.

If a transaction makes minimal usage of EXEC CICS LINK, for example it normally issues LINK only two or three times, the

overhead of building, initializing, and disposing of rununit/enclave level control blocks and the associated GETMAIN/FREEMAIN activity may be small relative to the cost of executing the business logic in the transaction. However, if EXEC CICS LINK is issued a hundred times, this overhead may become significant. A new function has been introduced in CICS/390 and Language Environment to improve the performance of EXEC CICS LINK processing.

The 'reusable rununit workarea' function, controlled by the CICS system initialization parameter, RUWAPOOL, has been introduced to improve the performance of transactions that use EXEC CICS LINK commands. When running CICS with 'RUWAPOOL=YES', a history is maintained by CICS, by transaction id, of rununit work area storage requirements, and this history is used to predict and pre-allocate a storage pool to satisfy all rununit work area requirements for every rununit/enclave instance in subsequent executions of each transaction. This should, taking the example given above, replace a hundred GETMAIN and FREEMAIN commands with a single GETMAIN/FREEMAIN, and have a considerable impact on performance.

This enhancement was originally introduced in APAR PQ16844 for CICS/ESA 4.1, and APAR PQ19370 for CICS TS 1.1 and 1.2. Language Environment APAR PQ14888 for Releases 1.5 through 1.9 is also required. Later releases of CICS and Language Environment support this function in the base code.

Paul Willats and Andy Krasun
IBM UK Laboratories (UK)

As a free service to subscribers and to remove the need to rekey the scripts, code in individual articles can be accessed on our Web site – starting with the January 1997 issue.

Subscribers need the user-id printed on the envelope containing their *Update* issue. Once they have registered, any code requested from issues to which they have subscribed will be e-mailed to them.

Resource definition display and alter commands – part 2

The resource definition displays provided by CICS standard facilities do not always provide the detail or the summary-level information desired. This month we conclude a utility that will list files by various categories and provide the ability to alter individual file and transaction attributes dynamically.

```
LOCALS1   DS   ØH
          LA   R9,FILETAB           Details
          LH   RA,EIBCPOSN
          SH   RA,=H'719'
          LA   R8,81(R8)                          Up display ptr
          LA   R8,81(R8)                          First Line
          MVC  Ø(7Ø,R8),Ø(R9)                     Move first line
          BAL  LINKREG,BOTSEL
          LA   R9,7Ø(R9)                          Up Detail ptr
          CLI  MOREBWD,X'1Ø'                      BWD possible ?
          BNE  HLQDET1                            No
          MVC  62(8,R8),=C'BWD(PF7)'              Show BWD
          B    HLQDET1
DETLOOP   DS   ØH
          LA   R8,81(R8)                          Up display ptr
          MVC  Ø(7Ø,R8),Ø(R9)                     Move a line
          BAL  LINKREG,BOTSEL
          LA   R9,7Ø(R9)                          Up Detail ptr
          CP   FILECNT,=P'1'                      Last Line ?
          BNE  HLQDET1                            No
          CLI  MOREFWD,X'1Ø'                      FWD possible ?
          BNE  HLQDET1                            No
          MVC  62(8,R8),=C'FWD(PF8)'              Show FWD
HLQDET1   DS   ØH
          SP   FILECNT,=P'1'                      Count down
          CP   FILECNT,=P'Ø'                      Finished ?
          BNE  DETLOOP                            No keep going
SNDMAPE        DS          ØH
          EXEC CICS SEND MAP ('SYMSYG9') ERASE FREEKB
          B    RETURNØ
* RETURN BUT COME BACK
RETURNØ   DS   ØH
          EXEC CICS RETURN TRANSID(EIBTRNID)                          *
          COMMAREA(COMMAS) LENGTH(COMMAL)
* RETURN AND FINISH
RETURN1   DS   ØH
          CLI  TSTAT,X'2Ø'                        At detail level ?
          BNE  RETURN1A                           No
```

```
         MVI  MODEFLG,C' '                        Clear mode flag
         B    BEGIN1                              Back up a level
RETURN1A DS   ØH
         EXEC CICS SEND CONTROL ERASE FREEKB
RETURN   DS   ØH
         EXEC CICS RETURN
BOTSEL   DS   ØH
         CLI  EIBAID,X'F5'
         BNER LINKREG
         SH   RA,=H'8Ø'
         BPR  LINKREG                             Not there yet
         MVI  CF,C'3'                             Signal to SYG8
         MVC  CF+1(8),1(R9)
         EXEC CICS XCTL PROGRAM('SYG8') COMMAREA(CF) LENGTH(CL)
         B    RETURN
* CONSTANTS
COMMAL   DC   H'1Ø'                               Commarea Length
CL       DC   H'9'
EDPAT    DC   X'4Ø2Ø2Ø2Ø212Ø'
OUTCTR   DC   CL6' '
ENDMARK  DC   XL3'ØØØØØØ',PL3'Ø'
HLQDHD1  DC   CL25'Files with HLQ of - "xxx"'
REMDHD1  DC   CL29'Remote files on System - ssss'
LSRDHD1  DC   CL26'Files Residing IN LSRPOOLØ'
LSRDHD2  DC   CL26'Files Residing IN NSR     '
HLQDHD2  DS   ØCL7Ø
         DC   CL1Ø' Fileid '
         DC   CL45'Datasetname '
         DC   CL7'Status '
         DC   CL8'Key Recl'
HM1      DC   CL57'To Exit Clear/PF3 For details select HLQ/RSYS or
LSRPOOL '
HM2      DC   CL57'Return to summary with PF3/Clear, File Display/Alter
PF5'
* The RSYS table initialized with spaces and zeros
* There are five entries and an end bucket
RSYITAB  DS   ØCL35
         DC   CL4'    ',PL3'Ø',CL4'    ',PL3'Ø',CL4'    ',PL3'Ø'
         DC   CL4'    ',PL3'Ø',XL4'ØØØØØØØØ',PL3'Ø'
* The HLQ table is initialized with spaces and zeros
* There are 28 entries here and it's moved twice
HLQITAB  DS   ØCL168
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
         DC   CL3'    ',PL3'Ø',CL3'    ',PL3'Ø',CL3'    ',PL3'Ø'
```

```
          DC   CL3'   ',PL3'Ø'
          LTORG
          END
```

SYMSYG9 – FILE SUMMARY DISPLAYS

Insert your company name at #1.

```
MAPSET3  DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=FREEKB,LANG=ASM,          *
               TIOAPFX=YES
SYMSYG9  DFHMDI SIZE=(24,8Ø),LINE=1,COLUMN=1,MAPATTS=(COLOR)
         DFHMDF POS=(Ø1,1),LENGTH=4,COLOR=GREEN,                       *
                 INITIAL='SYG9'
         DFHMDF POS=(Ø1,Ø6),LENGTH=11,COLOR=NEUTRAL,                   *
                  INITIAL='Partition :'
PARTI    DFHMDF POS=(Ø1,18),LENGTH=4,COLOR=NEUTRAL,                    *
                  INITIAL='rrrr'
         DFHMDF POS=(Ø1,23),LENGTH=17,COLOR=NEUTRAL,                   *
                  INITIAL='--- CICS Region :'
REGI     DFHMDF POS=(Ø1,41),LENGTH=7,COLOR=NEUTRAL,                    *
                  INITIAL='rrrrrrr'
         DFHMDF POS=(Ø1,49),LENGTH=5,COLOR=BLUE,INITIAL='DATE:'
DATE     DFHMDF POS=(Ø1,55),LENGTH=1Ø,COLOR=BLUE,INITIAL='XX.XX.XXXX'
         DFHMDF POS=(Ø1,66),LENGTH=5,COLOR=BLUE,INITIAL='TIME:'
TIME     DFHMDF POS=(Ø1,72),LENGTH=8,COLOR=BLUE,INITIAL='XX.XX.XX'
         DFHMDF POS=(Ø2,23),LENGTH=2Ø,COLOR=NEUTRAL,                   *
                       INITIAL='--- cccccccc ---    '     <===   #1
RDET1    DFHMDF POS=(Ø4,25),LENGTH=55,COLOR=NEUTRAL,INITIAL=' '
         DFHMDF POS=(Ø5,1),LENGTH=15,COLOR=BLUE,                       *
                INITIAL='Remote datasets'
RTOTOUT  DFHMDF POS=(Ø5,17),LENGTH=5,COLOR=BLUE
RDET2    DFHMDF POS=(Ø5,25),LENGTH=55,COLOR=BLUE,INITIAL=' '
LSRHED   DFHMDF POS=(Ø6,36),LENGTH=44,COLOR=NEUTRAL,INITIAL=' '
         DFHMDF POS=(Ø7,1),LENGTH=14,COLOR=BLUE,                       *
                INITIAL='Local datasets'
LTOTOUT  DFHMDF POS=(Ø7,17),LENGTH=5,COLOR=BLUE
         DFHMDF POS=(Ø7,25),LENGTH=4,COLOR=BLUE,                       *
                INITIAL='Open'
OTOTOUT  DFHMDF POS=(Ø7,3Ø),LENGTH=5,COLOR=BLUE
LSRDET   DFHMDF POS=(Ø7,36),LENGTH=44,COLOR=BLUE,INITIAL=' '
LHED1    DFHMDF POS=(Ø8,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET1    DFHMDF POS=(Ø9,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED2    DFHMDF POS=(1Ø,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET2    DFHMDF POS=(11,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED3    DFHMDF POS=(12,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET3    DFHMDF POS=(13,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED4    DFHMDF POS=(14,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET4    DFHMDF POS=(15,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED5    DFHMDF POS=(16,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET5    DFHMDF POS=(17,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
```

```
LHED6    DFHMDF POS=(18,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET6    DFHMDF POS=(19,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED7    DFHMDF POS=(2Ø,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET7    DFHMDF POS=(21,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
LHED8    DFHMDF POS=(22,2),LENGTH=78,COLOR=NEUTRAL,INITIAL=' '
LDET8    DFHMDF POS=(23,2),LENGTH=78,COLOR=BLUE,INITIAL=' '
HOUT     DFHMDF POS=(24,12),LENGTH=57,COLOR=BLUE
         DFHMSD TYPE=FINAL
         END
```

SOURCE FOR PROGRAM SYG8 – RESOURCE DETAIL DISPLAY AND ALTER

Adjust the code in SNDMAP2A to suit your own LPAR naming convention.

```
*ASM XOPTS(SP)
SYG8 RMODE ANY
         TITLE 'SYG8 - Online Create'
*
*      This Program will create transactions and/or files.
*      If a resource exists its current attributes are displayed.
*                                                                  *
R1         EQU  1
R2         EQU  2
R3         EQU  3
R4         EQU  4                                  Work Reg
R5         EQU  5
R6         EQU  6                                  Work Reg
R7         EQU  7                                  Subroutine Link Reg
DATAREG    EQU  8                                  DATA REGISTER
EIBREG     EQU  9                                  EIB REGISTER
BASE1      EQU  1Ø                                 Program base1
BASE2      EQU  11                                 Program base2
           COPY DFHAID
DFHEISTG   DSECT
APPL       DS   ØCL8
           DS   CL6
REGID      DS   CL2
ATIME      DS   PL8
TSTAT      DS   CL1
PFILE      DS   CL8
TRNERR     DS   CL1
FILERR     DS   CL1
REMERR     DS   CL1
CURP       DS   H
ITRAN      DS   CL4
ITRANCL    DS   CL8
IFILE      DS   CL8
ATTRIB     DS   CL83
```

```
ATTRIF    DS   CL115
DWORK     DS   D
INQPROG   DS   CL8
INQTCL    DS   CL8
INQTDLK   DS   F
INQDTIM   DS   F
INQTWA    DS   F
INQSPURG  DS   F
INQDSN    DS   CL44
INQRECF   DS   F
INQSTRNO  DS   F
INQLSRP   DS   F
INQADD    DS   F
INQBRW    DS   F
INQUPD    DS   F
INQDEL    DS   F
INQSTAT   DS   F
INQRECV   DS   F
INQSYSID  DS   CL4
INQKLEN   DS   F
INQRLEN   DS   F
          COPY SYMSYG8
SYG8      DFHEIENT CODEREG=(BASE1,BASE2),                              X
                EIBREG=(EIBREG),                                       X
                DATAREG=(DATAREG)
          B    BEGIN
          DC   CL12'PROGRAM ID: '
          DC   CL8'SYG8 '
          DC   CL4';   '
          DC   CL24'ASSEMBLY TIME AND DATE:'
          DC   CL8'&SYSTIME'
          DC   CL8'&SYSDATE'
BEGIN     DS   ØH
          MVC  CURP,=H'334'                        Set cursor
          MVI  TRNERR,X'ØØ'
          MVI  FILERR,X'ØØ'
          MVI  REMERR,X'ØØ'
          CLC  EIBCALEN,=H'Ø'                      Any Commarea?
          BE   SNDMAPØ                             First Time Through
          L    R1,DFHEICAP                         Address Commarea
          MVC  TSTAT,Ø(R1)                         Overlay
          CLC  EIBCALEN,=H'9'                      Passed FILE?
          BNE  BEGINØ                              Not from SYG9
          MVC  TSTAT(9),Ø(R1)                      Overlay
          MVC  IFILE,PFILE                         Insert Fileid
          MVC  FILEIDO,PFILE                       and in Map
          B    GENFSIN                       As if it had been entered
BEGINØ    DS   ØH
          EXEC CICS HANDLE AID PF3(RETURN1) PF4(SNDMAPØ) CLEAR(RETURN1)
BEGIN1    DS   ØH
          EXEC CICS RECEIVE MAP('SYMSYG8')
```

```
          BAL  R7,VALTRAN                    Verify transaction fields
          BAL  R7,VALFILE                    Verify file fields
          BAL  R7,VALREMT                    Verify remote fields
SNDMAPØ   DS   ØH
          EXEC CICS ASSIGN APPLID(APPL)
          MVI  REGID+1,X'4Ø'                 Make sure its a space
          CLI  EIBAID,DFHPF4                 or was it a reset/cancel
          BE   SNDMAP1B
          CLC  EIBCALEN,=H'Ø'                Any Commarea?
          BNE  SNDMAP1A                      Not first time through
SNDMAP1B  DS   ØH
          MVI  TSTAT,X'ØØ'
          MVC  TRANIDO,=C'____'              Reset input
          MVC  PROGIDO,=C'_       '          Reset input
          MVC  FILEIDO,=C'___     '          Reset input
          MVC  REMTIDO,=C'___     '          Reset input
          MVC  SYSIDO,=C'____'               Reset input
          MVC  DSNAMEO,FITMPLT               Only to space fill
          MVC  DSNAMEO(5),=C'_____'          Set next 5 chars
          B    SNDMAP2
SNDMAP1A  DS   ØH
          CLI  TRNERR,X'FF'                  No Go for Transaction
          BE   GENFILE
          EXEC CICS HANDLE CONDITION TRANSIDERR(GODOEM)
          EXEC CICS INQUIRE TRANSACTION(ITRAN)                          X
                     PROGRAM(INQPROG) TASKDATALOC(INQTDLK)              X
                     PURGEABILITY(INQSPURG) DTIMEOUT(INQDTIM)           X
                     TWASIZE(INQTWA) TRANCLASS(INQTCL)
          CLI  TSTAT,X'Ø1'                   Is this Deja Vu ?
          BE   GODOEM                        Must be a confirm
          MVC  MESS1O,=C'Transid(s) exist Press ENTER to replace '

*      When a transaction exists its attributes are displayed
          MVC  PROGIDI,INQPROG
          MVI  DLOCI,C'A'
          CLC  INQTDLK,DFHVALUE(ANY)
          BE   INQT1
          MVI  TDLOCI,C'B'
INQT1     DS   ØH
          MVI  SPURGI,C'Y'
          CLC  INQSPURG,DFHVALUE(PURGEABLE)
          BE   INQT2
          MVI  SPURGI,C'N'
INQT2     DS   ØH
          MVC  DTIMEI,=C'NO '
          CLC  INQDTIM,=F'Ø'
          BE   INQT3
          L    R4,INQDTIM                    DTIMEOUT in binary seconds
          CVD  R4,DWORK                      must be converted to
          DP   DWORK,=P'6Ø'                  'MSS' is display format!
          MVZ  DWORK+6(1),DWORK+5
```

```
          UNPK DTIMEI,DWORK+6(2)
          OC   DTIMEI+2(1),=X'FØ'            Make sign X'Fn
INQT3     DS   ØH
          L    R4,INQTWA                     TWASIZE in binary
          CVD  R4,DWORK                      must be converted to
          UNPK TWASZI,DWORK+5(3)             display
          OC   TWASZI+3(1),=X'FØ'            Make sign X'Fn
          MVI  TCLASI,C'N'
          CLC  INQTCL,=C'DFHTCLØØ'           Means no TRANCLASS !
          BE   INQT5
          MVI  TCLASI,C'Y'
INQT5     DS   ØH
          MVI  TSTAT,X'Ø1'
          MVI  TRANIDA,C'/'                  Lock Transaction
          MVC  CURP,=H'35Ø'                  Position at program
          B    SNDMAP2A
GODOEM    DS   ØH
          CLI  PROGIDI,C'_'
          BNE  CKT14
          MVC  MESS1O,=C'Program is a mandatory field           '
          MVC   CURP,=H'35Ø'
          B     GENFILE
CKT14     DS    ØH
          CLI   PROGIDI,X'4Ø'
          BNE   CKT15
          MVC   MESS1O,=C'Program cannot start with a space!
          MVC   CURP,=H'35Ø'
          B     GENFILE
CKT15     DS    ØH
          CLC   PROGIDI(3),=C'DFH'
          BNE   CKT16
          MVC   MESS1O,=C'DFH is reserved for IBM programs only   '
          MVC   CURP,=H'35Ø'
          B     GENFILE
CKT16     DS    ØH
          MVC   ATTRIB+5(8),PROGIDI
          CLI   TCLASI,C'N'                  TRANCLASS specified ?
          BE    GENTRAN                      No bypass create
          MVC   ITRANCL(4),ITRAN             Tranclass id first 4 char
          MVC   ITRANCL+4(4),=C'CMXT'        Tranclass id second 4 char
          EXEC CICS CREATE TRANCLASS(ITRANCL)                          X
                    ATTRIBUTES(ATTRIC)                                 X
                    ATTRLEN(ATTRICL)
* Add TRANCLASS to Transaction definition
          MVC  ATTRIB+67(16),=C'TRANCL(xxxxCMXT)'
          MVC  ATTRIB+74(4),ITRAN
GENTRAN   DS   ØH
          EXEC CICS CREATE TRANSACTION(ITRAN)                          X
                     ATTRIBUTES(ATTRIB)                                X
                     ATTRLEN(ATTRITL)
          MVC  MESS1O,=C'     Transaction  Created             '
```

```
          MVC  MESS1O(4),ITRAN
          CLI  TCLASI,C'N'                        TRANCLASS specified ?
          BE   GENFIL2                            No
          MVC  MESS1O+24(13),=C', with TCLASS'
GENFIL2   DS   ØH
          MVC  TRANIDO,=C'____'                   Reset input
          MVC  PROGIDO,=C'_       '               Reset input
          CLI  TSTAT,X'ØØ'                        Create state ?
          BE   GENFILE
          MVC  MESS1O+18(3),=C'Upd'               Update state
GENFILE   DS   ØH
          CLI  FILERR,X'FF'                       No Go for File(s) ?
          BE          GENREMT
          MVC   ATTRIF+5(44),DSNAMEI
GENFSIN   DS          ØH
          EXEC CICS HANDLE CONDITION FILENOTFOUND(GODOIT)
          EXEC CICS INQUIRE FILE(IFILE)                                    X
                    DSNAME(INQDSN) RECORDFORMAT(INQRECF)                   X
                    ENABLESTATUS(INQSTAT) RECOVSTATUS(INQRECV)             X
                    STRINGS(INQSTRNO) LSRPOOLID(INQLSRP) ADD(INQADD)       X
                    BROWSE(INQBRW) DELETE(INQDEL) UPDATE(INQUPD)           X
                    REMOTESYSTEM(INQSYSID)
          CLC  INQSYSID,=C'    '                  Remote ?
          BE   NOTREM                             No
          MVC  REMTIDO,FILEIDO  Pass FILEID
          MVC  FILEIDO,=C'___     '               Reset input
          MVC  MESS2O,=C'                                        '
          MVC  CURP,=H'1Ø69'
          XC   REMERR,REMERR
          B    GENREMT                            Process Remote
NOTREM    DS   ØH
          CLI  TSTAT,X'Ø2'                        Is this Deja Vu ?
          BE   FCONF                              Must be a confirm
          MVC  MESS2O,=C'File(s) exist Press ENTER to replace    '
*      When a file exists its attributes are displayed
          MVC  DSNAMEI,INQDSN
          MVI  RECFMI,C'F'
          CLC  INQRECF,DFHVALUE(FIXED)
          BE   INQF1
          MVI  RECFMI,C'V'
INQF1     DS   ØH
          L    R4,INQSTRNO                        Strings in binary
          CVD  R4,DWORK                           must be converted to
          UNPK STRNOI,DWORK+6(2)                  display
          OC   STRNOI+1(1),=X'FØ'                 Make sign X'Fn
          MVI  LSRPI,C'N'                         Assume LSRPOOL None
          CLI  INQLSRP+3,X'ØØ'                    Was it zero
          BE   INQF2                              Carry on
          MVC  LSRPI,INQLSRP+3                   Move non zero LSRPOOLID
          OC   LSRPI,=X'FØ'                       Make sign X'Fn
INQF2     DS   ØH
```

```
          MVI  ADDI,C'Y'                          Assume Addable
          CLC  INQADD,DFHVALUE(ADDABLE)
          BE   INQF3
          MVI  ADDI,C'N'                          Not Addable
INQF3     DS   ØH
          MVI  BROWSEI,C'Y'                       Assume Browsable
          CLC  INQBRW,DFHVALUE(BROWSABLE)
          BE   INQF4
          MVI  BROWSEI,C'N'                       Not Browsable
INQF4     DS   ØH
          MVI  DELETEI,C'Y'                       Assume Deletable
          CLC  INQDEL,DFHVALUE(DELETABLE)
          BE   INQF5
          MVI  DELETEI,C'N'                       Not Deletable
INQF5     DS   ØH
          MVI  UPDATEI,C'Y'                       Assume Updatable
          CLC  INQUPD,DFHVALUE(UPDATABLE)
          BE   INQF6
          MVI  UPDATEI,C'N'                       Not Updatable
INQF6     DS   ØH
          MVI  STATUSI,C'E'                       Assume Enabled
          CLC  INQSTAT,DFHVALUE(ENABLED)
          BE   INQF7
          MVI  STATUSI,C'D'                       Disabled
          CLC  INQSTAT,DFHVALUE(DISABLED)
          BE   INQF7
          MVI  STATUSI,C'U'                       leaving Unenabled
INQF7     DS   ØH
          MVI  RCVRYI,C'B'                        Assume Backout
          CLC  INQRECV,DFHVALUE(RECOVERABLE)
          BE   INQF8
          MVI  RCVRYI,C'N'
INQF8     DS   ØH
          MVI  TSTAT,X'Ø2'
          MVI  FILEIDA,C'/'                       Lock Fileid
          MVC  CURP,=H'67Ø'                       Position on DSN
          B    SNDMAP2A
FCONF     DS   ØH
          EXEC CICS SET FILE(IFILE) DISABLED CLOSED
GODOIT    DS   ØH
          CLC  MESS2O,=C'DSName contains ## but Fileid does not  '
          BE   GENREMT
          CLI  DSNAMEI+4,C'_'
          BNE  CKF14
          MVC  MESS2O,=C'DSName is a mandatory field             '
          MVC  CURP,=H'67Ø'
          B    GENREMT
CKF14     DS   ØH
          CLI  DSNAMEI,X'4Ø'
          BNE  CKF15
          MVC  MESS2O,=C'DSName cannot start with a space!
```

```
          MVC  CURP,=H'67Ø'
          B    GENREMT
CKF15     DS   ØH
          EXEC CICS CREATE FILE(IFILE)                                  X
                     ATTRIBUTES(ATTRIF)                                 X
                     ATTRLEN(ATTRIFL)
          MVC  MESS2O,=C'          File  Created                 '
          MVC  MESS2O(8),IFILE
          CLI  TSTAT,X'ØØ'                        Create state ?
          BE   GENGIL2A
          MVC  MESS2O+15(3),=C'Upd'               Update state
GENGIL2A  DS   ØH
          MVC  FILEIDO,=C'___     '               Reset input
          MVC  DSNAMEO,FITMPLT                    Only to space fill
          MVC  DSNAMEO(5),=C'_____'               Set next 4 chars
GENREMT   DS   ØH
          CLI  REMERR,X'FF'                       No Go for Remote(s) ?
          BE          SNDMAP2
          EXEC CICS HANDLE CONDITION FILENOTFOUND(GOROIT)
          EXEC CICS INQUIRE FILE(IFILE) RECORDSIZE(INQRLEN)             X
                        REMOTESYSTEM(INQSYSID) KEYLENGTH(INQKLEN)
          CLC  INQSYSID,=C'    '                  Local ?
          BNE  REMOK       No
          MVC  FILEIDO,REMTIDO  Pass FILEID
          MVC  REMTIDO,=C'___     '               Reset input
          XC   FILERR,FILERR                      Clear flag
          B    GENFILE                            Process Remote
REMOK     DS   ØH
          CLI  TSTAT,X'Ø3'                        Is this Deja Vu ?
          BE   GOROIT                             Must be a confirm
          MVC  MESS2O,=C'Remote(s) exist Press ENTER to replace  '
*      When a remote exists its attributes are displayed
          MVC  SYSIDI,INQSYSID
          L    R4,INQKLEN                         Keyleng in binary
          CVD  R4,DWORK                           must be converted to
          UNPK KEYLENI,DWORK+5(3)                 display
          OC   KEYLENI+2(1),=X'FØ'                Make sign X'Fn
          L    R4,INQRLEN                         Recleng in binary
          CVD  R4,DWORK                           must be converted to
          UNPK RECLENI,DWORK+4(4)                 display
          OC   RECLENI+3(1),=X'FØ'                Make sign X'Fn
          MVI  TSTAT,X'Ø3'
          MVI  REMTIDA,C'/'                       Lock Fileid
          MVC  CURP,=H'1Ø69'                      Position on SYSID
          B    SNDMAP2A
GOROIT    DS   ØH
          EXEC CICS HANDLE CONDITION SYSIDERR(CKRØ2)
          EXEC CICS INQUIRE CONNECTION(SYSIDI)
          EXEC CICS CREATE FILE(IFILE)                                  X
                     ATTRIBUTES(ATTRIF)                                 X
                     ATTRLEN(ATTRIFL)
```

```
         BNE  GENRILØ                                No so finished
CKRØ2    DS   ØH
         MVC  MESS2O,=C'Remote system not recognized.          '
         MVC  CURP,=H'1Ø69'
         B    SNDMAP2
GENRILØ  DS   ØH
         MVC  MESS2O,=C'         Remote  Created               '
         MVC  MESS2O(8),IFILE
         CLI  TSTAT,X'ØØ'                            Create state ?
         BE   GENRIL2A
         MVC  MESS2O+17(3),=C'Upd'                   Update state
GENRIL2A DS   ØH
         MVC  REMTIDO,=C'___     '                   Reset input
SNDMAP2  DS   ØH
         MVI  TSTAT,X'ØØ'
SNDMAP2A DS   ØH
         EXEC CICS ASSIGN APPLID(APPL)
         MVC  REGIO,APPL                             Move Applid to Map
         MVC  PARTIO,=C'DEVL'
         CLC  APPL+3(1),=C'C'                        PROD is C
         BNE  GTIME
         MVC  PARTIO,=C'PROD'
GTIME    DS   ØH
         EXEC CICS ASKTIME ABSTIME(ATIME)
         EXEC CICS FORMATTIME ABSTIME(ATIME) DDMMYYYY(DATEO)          *
                   TIME(TIMEO) DATESEP TIMESEP
SNDMAPE  DS   ØH
         EXEC CICS SEND MAP ('SYMSYG8') CURSOR(CURP) ERASE FREEKB
* Return but Come Back
RETURNØ  DS   ØH
         EXEC CICS RETURN TRANSID('SYG8')                             *
               COMMAREA(TSTAT) LENGTH(COMMAL)
* Return and Finish
RETURN1  DS   ØH
         EXEC CICS SEND CONTROL ERASE FREEKB
         EXEC CICS RETURN
* Subroutine to Verify Transaction input fields
VALTRAN  DS   ØH
         MVC  ATTRIB,TRTMPLT                         Set the template
         CLC  TRANIDI,=C'____'
         BE   TRNERRØ
         CLI  TRANIDI,X'4Ø'
         BNE  CKT12
         MVC  MESS1O,=C'Transaction cannot start with a space!  '
         B    TRNERRØ
CKT12    DS   ØH
         CLI  TRANIDI,C'C'
         BNE  CKT13
         MVC  MESS1O,=C'Transaction beginning C reserved for IBM'
         B    TRNERRØ
CKT13    DS   ØH
```

```
         MVC  TDLOCI,C'A'
         BE   CKT1
         MVC  ATTRIB+25(3),=C'BEL'
         CLI  TDLOCI,C'B'
         BE   CKT1
         MVC  MESS1O,=C'Taskdataloc must be A (Any) or B (Below)'
         MVC  CURP,=H'374'
         B    TRNERRØ
CKT1     DS   ØH
         CLC  DTIMEO,=C'NO '
         BE   CKT2                                OK
         MVC  ATTRIB+35(3),DTIMEI
         CLC  DTIMEI,=C'ØØØ'                      Zero becomes No
         BNE  CKT1A
         MVC  DTIMEO,=C'NO '
         MVC  ATTRIB+35(3),=C'NO '
         B    CKT2                                OK
CKT1A    DS   ØH
         TRT  DTIMEI,TRANTAB1                     Numeric ?
         BZ   CKT2                                OK
         MVC  MESS1O,=C'DTimeout must be numeric (zero for none)'
         MVC  CURP,=H'388'
         B    TRNERRØ
CKT2     DS   ØH
         CLI  SPURGI,C'Y'
         BE   CKT3
         MVC  ATTRIB+43(3),=C'NO '
         CLI  SPURGI,C'N'
         BE   CKT3
         MVC  MESS1O,=C'Spurge must be Y (yes) or N (no)        '
         MVC  CURP,=H'454'
         B    TRNERRØ
CKT3     DS   ØH
         CLI  TPURGI,C'Y'
         BE   CKT4
         MVC  ATTRIB+52(3),=C'NO '
         CLI  TPURGI,C'N'
         BE   CKT4
         MVC  MESS1O,=C'Tpurge must be Y (yes) or N (no)        '
         MVC  CURP,=H'468'
         B    TRNERRØ
CKT4     DS   ØH
         CLI  TCLASI,C'Y'
         BE   CKT5
         CLI  TCLASI,C'N'
         BE   CKT5
         MVC  MESS1O,=C'TClass must be Y (yes) or N (no)        '
         MVC  CURP,=H'534'
         B    TRNERRØ
CKT5     DS   ØH
         MVC  ATTRIB+61(4),TWASZI
```

```
         TRT  TWASZI,TRANTAB1                      Numeric ?
         BZR  R7                                   All OK
         MVC  MESS1O,=C'TWASize must be numeric (zero for none) '
         MVC  CURP,=H'548'
TRNERRØ  DS   ØH
         MVI  TRNERR,X'FF'
         BR   R7
VALFILE  DS   ØH
         CLC  FILEIDI(3),=C'___'
         BE   FILERRØ
         MVC  ATTRIF,FITMPLT                       Set the template
         CLI  FILEIDI,X'4Ø'
         BNE  CKF12
         MVC  MESS2O,=C'Fileid cannot start with a space!       '
         MVC  CURP,=H'65Ø'
         B    FILERRØ
CKF12    DS   ØH
         MVC  IFILE,FILEIDI                        Fileid accepted
         CLI  RECFMI,C'F'
         BE   CKF1
         MVI  ATTRIF+59,C'V'
         CLI  RECFMI,C'V'
         BE   CKF1
         MVC  MESS2O,=C'Recfm must be F (Fixed) or V (Variable) '
         MVC  CURP,=H'755'
         B    FILERRØ
CKF1     DS   ØH
         MVC  ATTRIF+66(2),STRNOI
         TRT  STRNOI,TRANTAB1                      Numeric ?
         BZ   CKF2                                 OK
CKF1A    DS   ØH
         MVC  MESS2O,=C'STRNO must be numeric and not zero
         MVC  CURP,=H'766'
         B    FILERRØ
CKF2     DS   ØH
         CLC  STRNOI,=C'ØØØØ'                      No strings ?
         BE   CKF1A                                Error
         MVC  ATTRIF+113(1),LSRPI
         CLI  LSRPI,C'Ø'                           LSR Ø not valid
         BE   CKF2A                                Error
         CLI  LSRPI,C'9'                           LSR 9 not valid
         BE   CKF2A                                Error
         CLI  LSRPI,C'N'                           LSRpool None
         BE   CKF3                                 OK
         TRT  LSRPI,TRANTAB1                       Numeric ?
         BZ   CKF3                                 OK
CKF2A    DS   ØH
         MVC  MESS2O,=C'LSRPool must 1 through 8 or N (None)
         MVC  CURP,=H'78Ø'
         B    FILERRØ
CKF3     DS   ØH
```

```
         MVC  ATTRIF+74(1),STATUSI
         CLI  STATUSI,C'E'
         BE   CKF4
         CLI  STATUSI,C'U'
         BE   CKF4
         CLI  STATUSI,C'D'
         BE   CKF4
         MVC  MESS2O,=C'Status must be E, D or U               '
         MVC  CURP,=H'829'
         B    FILERRØ
CKF4     DS   ØH
         MVC  ATTRIF+83(1),RCVRYI
         CLI  RCVRYI,C'B'
         BE   CKF5
         CLI  RCVRYI,C'N'
         BE   CKF5
         MVC  MESS2O,=C'Recovery must be N (none) or B (backout)'
         MVC  CURP,=H'843'
         B    FILERRØ
CKF5     DS   ØH
         MVC  ATTRIF+94(1),BROWSEI
         CLI  BROWSEI,C'Y'
         BE   CKF6
         CLI  BROWSEI,C'N'
         BE   CKF6
         MVC  MESS2O,=C'Browse must be Y (yes) or N (no)       '
         MVC  CURP,=H'9Ø9'
         B    FILERRØ
CKF6     DS   ØH
         MVC  ATTRIF+88(1),ADDI
         CLI  ADDI,C'Y'
         BE   CKF7
         CLI  ADDI,C'N'
         BE   CKF7
         MVC  MESS2O,=C'Add must be Y (yes) or N (no)          '
         MVC  CURP,=H'918'
         B    FILERRØ
CKF7     DS   ØH
         MVC  ATTRIF+1Ø1(1),DELETEI
         CLI  DELETEI,C'Y'
         BE   CKF8
         CLI  DELETEI,C'N'
         BE   CKF8
         MVC  MESS2O,=C'Delete must be Y (yes) or N (no)       '
         MVC  CURP,=H'93Ø'
         B    FILERRØ
CKF8     DS   ØH
         MVC  ATTRIF+1Ø6(1),UPDATEI
         CLI  UPDATEI,C'Y'
         BER  R7
         CLI  UPDATEI,C'N'
```

```
         BER  R7
         MVC  MESS2O,=C'Update must be Y (yes) or N (no)        '
         MVC  CURP,=H'942'
FILERRØ  DS   ØH
         MVI  FILERR,X'FF'
         BR   R7
VALREMT  DS   ØH
         CLC  REMTIDI(3),=C'___'
         BE   REMERRØ
         MVC  ATTRIF,RETMPLT                        Set the template
         CLI  REMTIDI,X'4Ø'
         BNE  CKRØ1
         MVC  MESS2O,=C'Remote cannot start with a space!
         MVC  CURP,=H'1Ø5Ø'
         B    REMERRØ
CKRØ1    DS   ØH
         MVC  IFILE,REMTIDI                         Fileid accepted
         MVC  ATTRIF+13(4),SYSIDI                   Store SYSID
         MVC  ATTRIF+26(3),KEYLENI                  Store Keylen
         TRT  KEYLENI,TRANTAB1                      Numeric ?
         BZ   CKRØ3                                 All OK
CKRØ1A   DS   ØH
         MVC  MESS2O,=C'KEYLen must be in range ØØØ to 255.     '
         MVC  CURP,=H'1Ø85'
         B    REMERRØ
CKRØ3    DS   ØH
         CLC  KEYLENI,=C'ØØØ'
         BE   CKRØ1A
         CLC  KEYLENI,=C'255'
         BH   CKRØ1A
         CLC  RECLENI,=C'ØØØØ'
         BE   CKRØ3A
         MVC  ATTRIF+39(4),RECLENI                  Store REClen
         TRT  RECLENI,TRANTAB1                      Numeric ?
         BZR  R7                                    All OK
CKRØ3A   DS   ØH
         MVC  MESS2O,=C'RECLen must be numeric and not zero.    '
         MVC  CURP,=H'11ØØ'
REMERRØ  DS   ØH
         MVI  REMERR,X'FF'
         BR   R7
* Constants
SPACES    DC   CL8Ø' '
COMMAL    DC   H'1'
TDLEN     DC   H'8Ø'
ATTRITL   DC   H'83'
ATTRICL   DC   H'6'
ATTRIFL   DC   H'115'
ATTRIC    DC   CL6'MAX(1)'
TRTMPLT   DC   CL14'PROG(        )'
          DC   CL42' TASKDATAL(ANY) DTIM(3ØØ) SP(YES) TPU(YES)'
```

```
         DC    CL27' TWA(ØØØØ)                 '
FITMPLT  DC    CL51'DSNA(                                            ) '
         DC    CL35'RECORDF(F) STR(2Ø) STA(E) RECOV(B) '
         DC    CL29'A(Y) BR(Y) DEL(Y) U(Y) LSR(1)'
RETMPLT  DC    CL51'REMOTESYSTEM(    ) KEYLEN(   ) RECORDS(    )       '
         DC    CL35'                                 '
         DC    CL29'                           '
TRANTAB1 DS    ØF
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
         DC    X'ØØØØØØØØØØØØØØØØØØØØFFFFFFFFFFFF'
         LTORG
         END   SYG8
```

SYMSYG8 – RESOURCE DETAIL DISPLAY AND ALTER

Insert your company name at #1.

```
MAPSET3  DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=FREEKB,LANG=ASM,         *
                        TIOAPFX=YES
SYMSYG8  DFHMDI SIZE=(24,8Ø),LINE=1,COLUMN=1,MAPATTS=(COLOR)
         DFHMDF POS=(Ø1,1),LENGTH=4,COLOR=GREEN,                       *
                        INITIAL='SYG8'
         DFHMDF POS=(Ø1,Ø6),LENGTH=11,COLOR=NEUTRAL,                   *
                     INITIAL='Partition :'
PARTI    DFHMDF POS=(Ø1,18),LENGTH=4,COLOR=NEUTRAL,                    *
                     INITIAL='rrrr'
         DFHMDF POS=(Ø1,23),LENGTH=17,COLOR=NEUTRAL,                   *
                               INITIAL='--- CICS Region :'
REGI     DFHMDF POS=(Ø1,41),LENGTH=7,COLOR=NEUTRAL,                    *
                               INITIAL='rrrrrrr'
         DFHMDF POS=(Ø1,49),LENGTH=5,COLOR=BLUE,INITIAL='DATE:'
DATE     DFHMDF POS=(Ø1,55),LENGTH=1Ø,COLOR=BLUE,INITIAL='XX.XX.XXXX'
         DFHMDF POS=(Ø1,66),LENGTH=5,COLOR=BLUE,INITIAL='TIME:'
TIME     DFHMDF POS=(Ø1,72),LENGTH=8,COLOR=BLUE,INITIAL='XX.XX.XX'
         DFHMDF POS=(Ø2,23),LENGTH=2Ø,COLOR=NEUTRAL,                   *
                INITIAL='----- cccccccc -----'               <==== #1
```

```
         DFHMDF POS=(Ø3,21),LENGTH=23,COLOR=NEUTRAL,                   *
                            INITIAL='Dynamic Create Facility'
         DFHMDF POS=(Ø5,1),LENGTH=12,COLOR=BLUE,                       *
                            INITIAL='Transaction('
TRANID   DFHMDF POS=(Ø5,14),LENGTH=4,ATTRB=(IC,FSET),COLOR=RED,        *
                            INITIAL='____'
         DFHMDF POS=(Ø5,19),LENGTH=1Ø,COLOR=BLUE,                      *
                            INITIAL=') Program('
PROGID   DFHMDF POS=(Ø5,3Ø),LENGTH=8,ATTRB=(FSET),COLOR=RED,          *
                            INITIAL='_       '
         DFHMDF POS=(Ø5,39),LENGTH=14,COLOR=BLUE,                      *
                            INITIAL=') Taskdataloc('
TDLOC    DFHMDF POS=(Ø5,54),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='A'
         DFHMDF POS=(Ø5,56),LENGTH=11,COLOR=BLUE,                      *
                            INITIAL=') DTimeout('
DTIME    DFHMDF POS=(Ø5,68),LENGTH=3,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='3ØØ'
         DFHMDF POS=(Ø5,72),LENGTH=1,COLOR=BLUE,                       *
                            INITIAL=')'
         DFHMDF POS=(Ø6,41),LENGTH=12,COLOR=BLUE,                      *
                            INITIAL='SPurge     ('
SPURG    DFHMDF POS=(Ø6,54),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='Y'
         DFHMDF POS=(Ø6,56),LENGTH=11,COLOR=BLUE,                      *
                            INITIAL=') TPurge  ('
TPURG    DFHMDF POS=(Ø6,68),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='Y'
         DFHMDF POS=(Ø6,7Ø),LENGTH=1,COLOR=BLUE,                       *
                            INITIAL=')'
         DFHMDF POS=(Ø7,41),LENGTH=12,COLOR=BLUE,                      *
                            INITIAL='TClass     ('
TCLAS    DFHMDF POS=(Ø7,54),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='N'
         DFHMDF POS=(Ø7,56),LENGTH=11,COLOR=BLUE,                      *
                            INITIAL=') TWASize ('
TWASZ    DFHMDF POS=(Ø7,68),LENGTH=4,ATTRB=(FSET),COLOR=YELLOW,        *
                            INITIAL='ØØØØ'
         DFHMDF POS=(Ø7,73),LENGTH=1,COLOR=BLUE,                       *
                            INITIAL=')'
         DFHMDF POS=(Ø9,1),LENGTH=8,COLOR=BLUE,                        *
                            INITIAL='Fileid ('
FILEID   DFHMDF POS=(Ø9,1Ø),LENGTH=8,ATTRB=(FSET),COLOR=RED,          *
                            INITIAL='___     '
         DFHMDF POS=(Ø9,19),LENGTH=1Ø,COLOR=BLUE,                      *
                            INITIAL=') DSName ('
DSNAME   DFHMDF POS=(Ø9,3Ø),LENGTH=44,ATTRB=(FSET),COLOR=RED,         *
                            INITIAL='___                              '
         DFHMDF POS=(Ø9,75),LENGTH=1,COLOR=BLUE,                       *
                            INITIAL=')'
         DFHMDF POS=(1Ø,21),LENGTH=13,COLOR=BLUE,                      *
```

```
                                INITIAL='Recordformat('
RECFM     DFHMDF POS=(10,35),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='F'
          DFHMDF POS=(10,37),LENGTH=8,COLOR=BLUE,                       *
                                INITIAL=') STRNO('
STRNO     DFHMDF POS=(10,46),LENGTH=2,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='20'
          DFHMDF POS=(10,49),LENGTH=10,COLOR=BLUE,                      *
                                INITIAL=') LSRPool('
LSRP      DFHMDF POS=(10,60),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='1'
          DFHMDF POS=(10,62),LENGTH=1,COLOR=BLUE,                       *
                                INITIAL=')'
          DFHMDF POS=(11,21),LENGTH=7,COLOR=BLUE,                       *
                                INITIAL='Status('
STATUS    DFHMDF POS=(11,29),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='E'
          DFHMDF POS=(11,31),LENGTH=11,COLOR=BLUE,                      *
                                INITIAL=') Recovery('
RCVRY     DFHMDF POS=(11,43),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='B'
          DFHMDF POS=(11,45),LENGTH=1,COLOR=BLUE,                       *
                                INITIAL=')'
          DFHMDF POS=(12,21),LENGTH=7,COLOR=BLUE,                       *
                                INITIAL='Browse('
BROWSE    DFHMDF POS=(12,29),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='Y'
          DFHMDF POS=(12,31),LENGTH=6,COLOR=BLUE,                       *
                                INITIAL=') Add('
ADD       DFHMDF POS=(12,38),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='Y'
          DFHMDF POS=(12,40),LENGTH=9,COLOR=BLUE,                       *
                                INITIAL=') Delete('
DELETE    DFHMDF POS=(12,50),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='Y'
          DFHMDF POS=(12,52),LENGTH=9,COLOR=BLUE,                       *
                                INITIAL=') Update('
UPDATE    DFHMDF POS=(12,62),LENGTH=1,ATTRB=(FSET),COLOR=YELLOW,        *
                                INITIAL='Y'
          DFHMDF POS=(12,64),LENGTH=1,COLOR=BLUE,                       *
                                INITIAL=')'
          DFHMDF POS=(14,1),LENGTH=8,COLOR=BLUE,                        *
                                INITIAL='Remote ('
REMTID    DFHMDF POS=(14,10),LENGTH=8,ATTRB=(FSET),COLOR=RED,           *
                                INITIAL='___     '
          DFHMDF POS=(14,19),LENGTH=9,COLOR=BLUE,                       *
                                INITIAL=') SYSid ('
SYSID     DFHMDF POS=(14,29),LENGTH=4,ATTRB=(FSET),COLOR=RED,           *
                                INITIAL='____'
          DFHMDF POS=(14,34),LENGTH=10,COLOR=BLUE,                      *
                                INITIAL=') KEYLen ('
```

```
KEYLEN    DFHMDF POS=(14,45),LENGTH=3,ATTRB=(FSET),COLOR=RED,          *
                              INITIAL='255'
          DFHMDF POS=(14,49),LENGTH=10,COLOR=BLUE,                     *
                              INITIAL=') RECLen ('
RECLEN    DFHMDF POS=(14,60),LENGTH=4,ATTRB=(FSET),COLOR=RED,          *
                              INITIAL='9999'
          DFHMDF POS=(14,65),LENGTH=1,COLOR=BLUE,                      *
                              INITIAL=')'
MESS1     DFHMDF POS=(15,21),LENGTH=40,COLOR=RED,                      *
                              INITIAL=' '
MESS2     DFHMDF POS=(16,21),LENGTH=40,COLOR=RED,INITIAL=' '
          DFHMDF POS=(17,21),LENGTH=40,COLOR=BLUE,                     *
                        INITIAL='Enter Transaction, Fileid or Remote.'
          DFHMDF POS=(24,21),LENGTH=41,COLOR=NEUTRAL,                  *
                         INITIAL='Clear or PF3 to EXIT, PF4 to CANCEL'
          DFHMSD TYPE=FINAL
          END
```


Simplifying AOR/FOR creation

In the world of CICS operations, there comes a time when a systems programmer is faced with the task of taking an AOR with resident (local) file entries and separating it into an AOR/FOR pair, with the resident (local) file entries residing on the FOR and the file entries on the AOR becoming remote entries.

The tasks of moving the resident (local) file entries are simple enough – merely the creation of a FOR CICS region and the inclusion of the DFHCSD group entries of the resident (local) file entries in the DFHLIST for the FOR. However, the changes involved for the AOR are a bit more complicated because what used to be local file entries now become remote file entries. This by itself is not bad, it can be accomplished easily with the use of DFHCSDUP customized output and the use of any text manipulation algorithm to generate the DFHCSDUP control cards. It's the task of locating the LRECL/KEYLENGTH for the datasets (and AIX/PATHs) that is labour-intensive.

SASFCTRM JCL/SAS jobstream was created with this in mind. The

general idea is to generate DFHCSDUP control statement to define remote file entries from the resident (local) file entries in the DFHCSD resource – GROUP or LIST – specified by the user. The jobstream does the following:

- UTIL – generate (extract) list of DFHCSD resource.
- GETDSN – extract GROUP, FILE, DSNAME and generate IDCAMS LISTC for each file.
- IDCAMS1 – execute IDCAMS LISTC against DSNAMEs of files.
- GETCLUST – extract LRECL and KEYLENGTH if file is a CLUSTER; extract AIX DSN if file is PATH; generate IDCAMS LISTC for AIX.
- IDCAMS2 – execute IDCAMS LISTC against AIX.
- GETAIX – extract AIXKEYLENGTH and base CLUSTER dataset name; generate IDCAMS LISTC for AIX base CLUSTER.
- IDCAMS3 – execute IDCAMS LISTC against AIX base cluster.
- GETAIXCL – extract AIX base CLUSTER LRECL.
- GENERATE – merge all information extracted so that each file entry definition has correct LRECL/KEYLENGTH.
- PRINT – use IEBEGNER to print generate DFHCSDUP control statements
- UTIL – apply DFHCSDUP to create remote file definition in new group.

The job stream is as follows:

```
//JOBCARD JOB
//*-------------------------------------------------------------------
//* SASFCTRM JOB STREAM IS DESIGNED TO GENERATE DFHCSDUP DEFINE
//* STATEMENTS FOR REMOTE DFHCSD FCT ENTRIES FROM LOCAL DFHCSD
//* ENTRIES.  THIS IS NECESSARY IF THE CICS REGION WAS AN AOR WITH
//* LOCAL FCT ENTRIES BUT NOW MUST BE DIVIDED INTO MULTIPLE AORS
//* WITH REMOTE FCT ENTRIES POINTING TO A FOR BECAUSE OF PERFORMANCE/
//* RECOVERY REASONS.
//*-------------------------------------------------------------------
//* FIRST STEP IS DFHCSDUP UTILITY STEP.  NOTICE THE FOLLOWING:
```

```
//*
//* 1) DFHCSD IS SET FOR READ-ONLY
//* 2) CICS SUPPLIED DFHCSDUP EXIT DFH$FORA IS USED AND MUST RESIDE
//*    IN THE STEPLIB
//* 3) EXTRACT CAN BE FOR EITHER LIST -AND/OR- GROUP.
//*
//* THE RESULT OF THIS STEP IS TO PRODUCE A LIST OF RESOURCES
//* (INCLUDING FCT ENTRIES) IN A DATASET, ONE RESOURCE ENTRY PER LINE.
//*-------------------------------------------------------------------
//UTIL   EXEC PGM=DFHCSDUP,PARM='CSD(READONLY)'
//STEPLIB  DD DSN=CICS.SDFHLOAD,DISP=SHR
//DFHCSD   DD DSN=CICS.DFHCSD,DISP=SHR
//CRFINPT  DD *
//CRFOUT   DD SYSOUT=*
//FOROUT   DD DISP=(MOD,PASS),DSN=&&CSDOUT,
//         UNIT=SYSALLDA,SPACE=(CYL,(1Ø,1Ø),RLSE)
//CBDOUT   DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD *
EXTRACT LIST(DFHLIST) OBJECTS USERPROGRAM(DFH$FORA)
//*-------------------------------------------------------------------
//*  THIS STEP FIRST READS THE FOROUT DATASET FOR FCT ENTRIES AND
//*  PRODUCES A LIST OF FCT NAMES ASSOCIATED WITH THE GROUP NAME AND
//*  DATASET NAME IN A SAS DATASET
//*
//*  SAS DATASET DSNS
//*      VARIABLES   FCT GROUP DSN
//*
//*  THIS STEP THEN SORTS THE CREATED SAS DATASET DSNS IN DATASET NAME
//*  ORDER, THEN GENERATES IDCAMS LISTC ENT(DATASET NAME) ALL COMMAND
//*  FOR EACH FCT ENTRY WITH A DATASET NAME INTO IDCAMSI1 TEMPORARY
//*  DATASET
//*-------------------------------------------------------------------
//GETDSN EXEC SAS
//SAS.WORK DD DISP=(,PASS),DSN=&&FCTDATA
//CSDDATA  DD DISP=(OLD,DELETE),DSN=&&CSDOUT
//IDCAMSI1 DD DISP=(,PASS),DSN=&&IDCAMSI1,UNIT=SYSALLDA,
//         SPACE=(CYL,(1,1)),DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=616Ø)
//SYSIN    DD *

/* GENERATE LIST OF FCT ENTRIES AND THE DSN ASSOCIATED          */

DATA DSNS (KEEP = FCT GROUP DSN);
INFILE CSDDATA;
INPUT @1  TYPE      $4.
@5  FCT       $8.
@13 GROUP     $8.
@79 DSN       $44. ;
IF TYPE NE 'FILE' THEN RETURN;
```

```
IF DSN  =: ' ' THEN RETURN;
OUTPUT;
RETURN;

/* PROC PRINT NOOBS UNIFORM; */
PROC SORT DATA=DSNS;
BY DSN;

DATA _NULL_;
FILE  IDCAMSI1;
SET DSNS;
PUT ' LISTC ENT(' DSN') ALL';
RETURN;

//*-------------------------------------------------------------------
//* THIS STEP USES IDCAMS TO PROCESS INPUT FROM IDCAMSI1 AND PLACES
//* THE OUTPUT INTO TEMPORARY DATASET IDCAMSO1
//*-------------------------------------------------------------------
//IDCAMS1  EXEC PGM=IDCAMS
//SYSPRINT DD   DISP=(,PASS),DSN=&&IDCAMSO1,UNIT=SYSALLDA,
//         SPACE=(CYL,(10,10),RLSE)
//SYSIN    DD   DISP=(OLD,DELETE),DSN=&&IDCAMSI1
//*-------------------------------------------------------------------
//*
//* THIS STEP USES THE IDCAMS OUTPUT FROM IDCAMS1 STEP AND EXTRACTS
//* TWO NEW SAS DATSETS:
//*
//* CLUSTER  ==> DSN KEYL LRECL
//* AIX      ==> DSN AIX
//*
//* UPON COMPLETION OF DATA EXTRACTION, THE CLUSTER DATASET IS SORTED
//* IN THE DATASET NAME ORDER.  THE AIX DATASET IS SORTED IN THE AIX
//* DATASETNAME ORDER, AND ONCE AGAIN IDCAMS LISTC ENT(AIX NAME) ALL
//* COMMAND IS GENERATED FOR EACH AIX ENTRY INTO IDCAMSI2 TEMPORARY
//* DATASET
//*-------------------------------------------------------------------
//GETCLUST EXEC SAS
//SAS.WORK DD DISP=(NEW,PASS),DSN=&&CLUSTER
//FCTDATA  DD DISP=(OLD,PASS),DSN=&&FCTDATA
//IDCAMSIN DD DISP=(OLD,DELETE),DSN=&&IDCAMSO1
//IDCAMSI2 DD DISP=(,PASS),DSN=&&IDCAMSI2,UNIT=SYSALLDA,
//         SPACE=(CYL,(1,1)),DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160)
//SYSIN    DD *

/* GENERATES CLUSTER DATASETS WITH DSN, KEY LENGTH AND LRECL     */
/* GENERATES AIX DATASET FOR PATH WITH DSN AND AIX               */
/* VARIABLE NAME OUTPUT IS USED TO ONLY OUTPUT ONCE FOR CLUSTER - */
/*  SINCE INDEX COMPONENT HAS LRECL PARAMETER ALSO               */

DATA CLUSTER (KEEP = DSN KEYL LRECL)
```

```
AIX (KEEP = DSN AIX);
INFILE IDCAMSIN RECFM=VB PAD;
INPUT @2 TYPE $4. @;
RETAIN DSN;
RETAIN KEYL;
RETAIN OUTPUT;
RETAIN CURRENT;
IF  TYPE =: 'CLUS' OR
TYPE =: 'PATH' THEN DO;
INPUT @18 DSN $44.;
OUTPUT = 'YES';
CURRENT = TYPE;
RETURN;
END;
INPUT @9 MISC $8. @;
IF MISC =: 'AIX' AND CURRENT =: 'PATH' THEN DO;
INPUT @18 AIX $44.;
OUTPUT AIX;
RETURN;
END;
ELSE IF MISC =: 'KEYLEN' THEN DO;
INPUT @3Ø KEYL $3.;
KEYL = TRANSLATE(KEYL,' ','-');
RETURN;
END;
ELSE DO;
INPUT @38 MISC $8. @;
IF MISC =: 'MAXLRECL' THEN DO;
INPUT @57 LRECL $5. ;
LRECL = TRANSLATE(LRECL,' ','-');
IF OUTPUT = 'YES' THEN
OUTPUT CLUSTER;
OUTPUT = 'NO' ;
RETURN;
END;
ELSE RETURN;
END;

PROC SORT DATA=CLUSTER;
BY DSN;

/* PROC PRINT DATA=CLUSTER UNIFORM; */

PROC SORT DATA=AIX;
BY AIX;

/* PROC PRINT DATA=AIX UNIFORM; */

DATA _NULL_;
FILE  IDCAMSI2;
```

```
SET AIX;
PUT ' LISTC ENT(' AIX') ALL';
RETURN;
//*-------------------------------------------------------------------
//* THIS STEP USES IDCAMS TO PROCESS INPUT FROM IDCAMSI2 AND PLACES
//* THE OUTPUT INTO TEMPORARY DATASET IDCAMSO2.
//*-------------------------------------------------------------------
//IDCAMS2  EXEC PGM=IDCAMS
//SYSPRINT DD   DISP=(,PASS),DSN=&&IDCAMSO2,UNIT=SYSALLDA,
//         SPACE=(CYL,(1Ø,1Ø),RLSE)
//SYSIN    DD   DISP=(OLD,DELETE),DSN=&&IDCAMSI2
//*-------------------------------------------------------------------
//* THIS STEP FIRST USES IDCAMS OUTPUT FROM IDCAMS2 STEP TO GENERATE
//* TWO SAS DATASETS:
//*
//* AIXKEY  => AIX KEYL
//* AIXCL   => AIX AIXCL
//*
//* THE TWO DATASETS ARE THEN SORTED BY DATASET (AIX) NAME AND MERGED
//* TO PRODUCE ANOTHER SAS DATASET:
//*
//* AIXKELY => AIX AIXCL KEYL
//*
//* IDCAMS LISTC ENT(AIXCL DATASET) ALL COMMAND IS THEN GENERATED
//* AND PLACED INTO IDCAMSI3 DATASET.
//*-------------------------------------------------------------------
//GETAIX   EXEC SAS
//SAS.WORK DD DISP=(,PASS),DSN=&&AIX
//CLUSTER  DD DISP=(OLD,PASS),DSN=&&CLUSTER
//IDCAMSAX DD DISP=(OLD,DELETE),DSN=&&IDCAMSO2
//IDCAMSI3 DD DISP=(,PASS),DSN=&&IDCAMSI3,UNIT=SYSALLDA,
//         SPACE=(CYL,(1,1)),DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=616Ø)
//SYSIN    DD *

/* GENERATES AIX DATASETS WITH DSN, KEY LENGTH, AND LRECL         */
/* VARIABLE NAME OUTPUT IS USED TO ONLY OUTPUT ONCE FOR CLUSTER - */
/* SINCE INDEX COMPONENT HAS LRECL PARAMETER ALSO.                */

DATA AIXKEY (KEEP = AIX KEYL )
AIXCL (KEEP = AIX AIXCL);
INFILE IDCAMSAX RECFM=VB PAD;
INPUT @2 TYPE $4. @;
RETAIN AIX;
RETAIN KEYL;
RETAIN OUTPUT;
IF  TYPE =: 'AIX' THEN DO;
INPUT @18 AIX $44.;
OUTPUT = 'YES';
RETURN;
END;
```

```
INPUT @9 MISC $8. @;
IF MISC =: 'CLUSTER-' THEN DO;
INPUT @18 AIXCL $44.;
     OUTPUT AIXCL;
     RETURN;
   END;
   ELSE IF MISC =: 'KEYLEN' THEN DO;
     INPUT @3Ø KEYL $3.;
     KEYL = TRANSLATE(KEYL,' ','-');
     IF OUTPUT = 'YES' THEN
       OUTPUT AIXKEY ;
     OUTPUT = 'NO' ;
     RETURN;
   END;
   ELSE RETURN;

PROC SORT DATA=AIXKEY;
   BY AIX;

PROC SORT DATA=AIXCL;
   BY AIX;

DATA AIXKEYL (KEEP=AIX AIXCL KEYL) ;
   MERGE AIXKEY AIXCL;
   BY AIX;

PROC SORT DATA=AIXKEYL;
   BY AIXCL;

DATA _NULL_;
   FILE  IDCAMSI3;
   SET AIXCL;
   PUT ' LISTC ENT(' AIXCL') ALL';
   RETURN;
//*-------------------------------------------------------------------
//* THIS STEP USES IDCAMS TO PROCESS INPUT FROM IDCAMSI3 AND PLACES
//* THE OUTPUT INTO TEMPORARY DATASET IDCAMSO3.
//*-------------------------------------------------------------------
//IDCAMS3  EXEC PGM=IDCAMS
//SYSPRINT DD   DISP=(,PASS),DSN=&&IDCAMSO3,UNIT=SYSALLDA,
//         SPACE=(CYL,(1Ø,1Ø),RLSE)
//SYSIN    DD   DISP=(OLD,DELETE),DSN=&&IDCAMSI3
//*-------------------------------------------------------------------
//* OUT OF THE IDCAMS OUTPUT FROM STEP IDCAMS3 SAS EXTRACTS DATASET:
//*
//* AXCDATA => AIXCL (BASE CLUSTER NAME FOR THE AIX)
//*            LRECL (LRECL FOR THE BASE CLUSTER)
//*
//* AND THE DATASET IS THEN SORTED IN THE AIX BASE CLUSTER NAME ORDER.
//*-------------------------------------------------------------------
```

```
//GETAIXCL EXEC SAS
//SAS.WORK DD DISP=(,PASS),DSN=&&AIXCL
//IDCAMSAC DD DISP=(OLD,DELETE),DSN=&&IDCAMSO3
//SYSIN    DD *

/* GENERATES BASE CLUSTER DSN LRECL                                  */
/* VARIABLE NAME OUTPUT IS USED TO OUTPUT ONLY ONCE FOR CLUSTER -  */
/* SINCE INDEX COMPONENT HAS LRECL PARAMETER ALSO                    */

DATA AXCDATA (KEEP = AIXCL LRECL) ;
   INFILE IDCAMSAC RECFM=VB PAD;
   INPUT @2 TYPE $4. @;
   RETAIN AIXCL;
   RETAIN LRECL;
   RETAIN OUTPUT;
   IF  TYPE =: 'CLUS' THEN DO;
     INPUT @18 AIXCL $44.;
     OUTPUT = 'YES';
     RETURN;
   END;
   INPUT @38 MISC $8. @;
   IF MISC =: 'MAXLRECL' THEN DO;
     INPUT @57 LRECL $5.;
     LRECL = TRANSLATE(LRECL,' ','-');
     IF OUTPUT = 'YES' THEN
       OUTPUT AXCDATA;
     OUTPUT = 'NO' ;
     RETURN;
   END;
   ELSE RETURN;

PROC SORT DATA=AXCDATA;
   BY AIXCL;

//*-------------------------------------------------------------------
//* THIS IS WHERE IT GETS TRICKY.
//*
//* 1) SAS DATASET AIXDATA IS CREATED BY MERGING SAS DATASETS AXCDATA
//*    AND AIXDATA, KEEPING VARIABLES AIX, LRECL, KEYL, KEYED ON AIX
//* 2) SAS DATASET PATHDATA IS CREATED BY MERGING SAS DATASETS AIX
//*    AND AIXDATA, KEEPING VARIABLES DSN, LRECL, KEYL, KEYED ON AIX
//* 3) SAS DATASET DSNDATA IS CREATED BY MERGING SAS DATASETS PATHDATA
//*    AND CLUSTER, KEEPING ALL VARIBLES DSN, PATH, LRECL, AND KEYL,
//*    KEYED ON DSN
//* 4) SAS DATASET MERGED IS CREATED BY MERGING THE SAS DATASETS DSNS
//*    AND DSNDATA, KEEPING VARIABLES GROUP, FCT, DSN, LRECL, KEYL -
//*    ALL THE INFORMATION NECESSARY TO GENERATE DFHCSDUP STATEMENTS
//* 5) MERGED DATAST IS SORTED IN GROUP/FCT ORDER
//* 6) _NULL_ DATA STEP IS USED TO READ IN THE MERGED SAS DATASET
//*    AND GENERATE DFHCSDUP CONTROL STATEMENTS.
```

```
//* NOTE:
//*    1) THE GROUP NAME FOR THE FILE DEFINITION BEING GENERATED IS
//*       THE ORIGINAL GROUP NAME WITH '$' SUBSITITUION IN BYTE 1
//*    2) THE REMOTE SYSID IS CODED AS '$RMT'
//*-------------------------------------------------------------------
//GENERATE  EXEC SAS
//SAS.WORK  DD DISP=(,PASS),DSN=&&COMPLETE
//AIX       DD DISP=(OLD,DELETE),DSN=&&AIX
//AIXCL     DD DISP=(OLD,DELETE),DSN=&&AIXCL
//CLUSTER   DD DISP=(OLD,DELETE),DSN=&&CLUSTER
//FCTDATA   DD DISP=(OLD,DELETE),DSN=&&FCTDATA
//CSDCARDS  DD DISP=(,PASS),DSN=&&CSDCARDS,
//          DCB=(RECFM=FB,LRECL=8Ø,BLKSIZE=616Ø),
//          UNIT=SYSALLDA,SPACE=(CYL,(1,1))
//AUDIT     DD SYSOUT=*
//SYSIN     DD *

/* AIX DATA HAS AIX OF PATH, AIX KEYL AND LRECL OF BASE CL.  */

DATA AIXDATA (KEEP=AIX LRECL KEYL);
   MERGE AIXCL.AXCDATA
         AIX.AIXKEYL ;
     BY AIXCL;

PROC SORT DATA=AIXDATA ;
   BY AIX;

/* PATHDATA HAS DSN OF PATH, AIX KEYL AND LRECL OF BASE CL.  */

DATA PATHDATA (KEEP=DSN LRECL KEYL);
   MERGE CLUSTER.AIX
         WORK.AIXDATA;
     BY AIX;

PROC SORT DATA=PATHDATA ;
   BY DSN;

/* DSN DATA MERGES CLUSTER AND PATH DSN, LRECL AND KELY      */

DATA DSNDATA ;
   MERGE WORK.PATHDATA
         CLUSTER.CLUSTER;
   BY DSN;

PROC SORT DATA=DSNDATA ;
   BY DSN;

/* MERGED HAS ALL DATA NECESSARY TO GENERATE DFHCSDUP DEFINE */
/*  REMOTE FCT ENTRY STATEMENTS.                             */
```

```
DATA MERGED (KEEP=GROUP FCT DSN LRECL KEYL) ;
   MERGE FCTDATA.DSNS
         WORK.DSNDATA;
     BY DSN;

PROC SORT DATA=MERGED;
   BY GROUP FCT;

PROC PRINT DATA=MERGED;
   ID GROUP FCT DSN LRECL KEYL;

DATA _NULL_;
   SET MERGED;
   IF LRECL = '     ' THEN DO;
     FILE AUDIT;
     PUT 'GROUP('GROUP') FCT('FCT') **NOT** DEFINED';
   END;
   ELSE DO;
     GROUPN = '$'||SUBSTR(GROUP,2,7);
     FILE CSDCARDS;
     PUT 'DEFINE FILE('FCT')  GROUP('GROUPN') REMOTESYSTEM($RMT)';
     PUT '  RECORDSIZE('LRECL')' ;
     IF KEYL NE '  Ø' THEN PUT '  KEYLENGTH('KEYL')';
     FILE AUDIT;
     PUT 'GROUP('GROUP') FCT('FCT') DEFINED';
   END;
   RETURN;
//*-------------------------------------------------------------------
//* PRINT THE GENERATE STATEMENTS
//*-------------------------------------------------------------------
//PRINT  EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1   DD DISP=(OLD,PASS),DSN=&&CSDCARDS
//SYSUT2   DD SYSOUT=*
//SYSIN    DD DUMMY
//*-------------------------------------------------------------------
//* RUN THE GENERATED DFHCSDUP CONTROL STATEMENTS.  NOTICE I HAVE
//* COND=(Ø,LE) SET UP TO BYPASS EXECUTION - I WOULD RUN IT AGAINST
//* A COPY OF DFHCSD FIRST!                                    CSH
//*-------------------------------------------------------------------
//UTIL   EXEC PGM=DFHCSDUP,COND=(Ø,LE)
//STEPLIB  DD DSN=CICS.SDFHLOAD,DISP=SHR
//DFHCSD   DD DSN=CICS.DFHCSD,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSIN    DD DISP=(OLD,DELETE),DSN=&&CSDCARDS
//
```

Chorng S (Jack) Hwang
Principal
HSA Systems (USA)

TMONCICS transaction record detailed analysis – part 1

The Monitor for CICS/ESA (TMCE) from Landmark Systems is widely used at CICS installations for monitoring, accounting, and problem analysis. The on-line component of TMCE offers excellent facilities for examining CICS transactions as they execute, and to review them after they have completed. TMCE keeps a VSAM database of all completed transactions – the only limit to how much data you can keep on-line is the size allocated to the VSAM datasets.

Once the VSAM datasets fill up, they can be archived, using TMCE program $CRUTIL, to a tape-based GDG or other media as appropriate. This archived data can then be post-processed using the Landmark Report Writer program LMRK700.

Although these facilities do an excellent job, I found that I wanted to extend the functionality of the product in two areas to assist in debugging problems encountered on production CICS systems.

The first problem I had was in dealing with the huge quantities of raw data that TMCE is capable of producing if all its monitoring facilities are switched on. In a CICS environment executing over 1 million transactions per day, the daily GDG cycle of the TMCE transaction (TA) records spanned several cartridges. Running the Report Writer against these files was extremely time-consuming, especially when multiple passes were required to display different parts of the TA record according to the specific problem at hand.

This led to the development of the program TMTAEXTR, which reads archived TMCE TA records and writes records matching its selection criteria to another file in decompressed format. This second file can then be processed by the Report Writer, or used as input to the REXX program described below.

The selection criteria are CICS job name, transaction code, and user-id. Any of these can be defaulted to all by using the string XXXXXXXX. If specified, the transaction code must be entered as four characters followed by four bytes of X'00'. This is the way TMCE stores the CICS transaction code internally, presumably to allow for future codes that

exceed four characters.

The selection is further refined using date and starting/ending time. These must be specified, unlike in the Report Writer where they are optional.

The program decompresses the TMCE TA records, if they are not already decompressed. This is optional during a TMCE archive run, but most installations probably use compression. This would not be necessary for using the output from TMTAEXTR with the Report Writer, but it does not prevent this use either. It is necessary for the REXX program to have decompressed data. This decompression is handled by a call to the routine $CRCPRS, supplied by Landmark along with sample calling code.

The second problem was that, once data has been archived from the VSAM database, the only way to interpret it is by using the Report Writer as mentioned above. While the Report Writer is very flexible, I wanted to be able to view every single field in a TA record and to be able to jump quickly between views of different records in a file.

Data can actually be loaded back into the VSAM database with TMCE Version 2, if it is archived in an appropriate format: but even the on-line analysis screens do not give all the data that I wanted to have access to in the format required.

As a result, I developed the REXX program TMTA, which reads decompressed data, either directly from the archive program or from TMTAEXTR. It lists the records on an ISPF panel very similar to the main TMCE option 6.TA panel; the main difference being that I included the abend code field on the screen.

Further record filtering is available, either from the primary panel where the dataset name to be examined is entered, or on the list of tasks panel.

Records can be selected on the list panel with an ‘X’ for expansion into either a full DSECT format display of every field in the record, or a Hex dump view of the record. Any segments that are present after the main body of the record can be displayed from the DSECT display by entering a 3-character command corresponding to the segment type.

Also from the DSECT screen, command ‘SAVE’ can be entered to

save a formatted listing of the DSECT view of the record, including all the segments, as a member of a PDS (userid.TMTA.SAVE), where the member name is TAnnnnn (nnnnn being the task number of the transaction).

The commands available are all documented in panel TMTAPH, which is invoked by the HELP PF key at any time in any panel.

TMTAEXTR SOURCE CODE

```
TMTAEXTR CSECT
*-------------------------------------------------------------------
*  EXTRACT TMONCICS V2.Ø TA RECORD TYPE BY JOBNAME, TRANS, AND
*  START/END TIME FROM DETAIL RECORD.
*-------------------------------------------------------------------
         COPY REGS                      <==========  Your Macro Here
         SAVE  (14,12),,*
         BALR  R9,Ø
         USING *,R9
         ST    R13,REGISTER_SAVE_AREA+4
         LA    R8,REGISTER_SAVE_AREA
         ST    R8,8(R13)
         LR    R13,R8
*-------------------------------------------------------------------
*  READ PARAMETERS FOR EXTRACTION.
*-------------------------------------------------------------------
         OPEN  (SYSIN,INPUT)
         GET   SYSIN               READ PARAMETER FILE
         LR    R3,R1               SAVE INPUT RECORD ADDRESS
         MVC   INPUT_JOB_NAME,Ø(R3)
         MVC   INPUT_TRANSACTION,8(R3)
         MVC   INPUT_USERID,16(R3)
         MVC   INPUT_DATE,24(R3)
         MVC   INPUT_START_TIME,32(R3)
         MVC   INPUT_END_TIME,4Ø(R3)
@CLPRM   CLOSE (SYSIN)
*-------------------------------------------------------------------
*  GET BUFFER AREA FOR RECORD EXPANSION
*-------------------------------------------------------------------
         GETMAIN RU,LV=32768       SET MAXIMUM SIZE
         MVC   Ø(4,R1),=F'32768'   SET 4-BYTE RDW IN EXPANSION BUFFER
         LA    R1,8(,R1)           STEP PAST RDW...
         ST    R1,PARMPEX@         .. AND SET IN PARM LIST
*-------------------------------------------------------------------
*  OPEN FILES
*-------------------------------------------------------------------
         OPEN  (LMRKIN,INPUT)
         OPEN  (LMRKOUT,OUTPUT)
```

```
         OPEN  (SYSPRINT,OUTPUT)
*
         MVC   OUTREC+2(31),=CL31'TMTAEXTR - EXTRACT SPECIFIC JOB'
         MVC   OUTREC+33(31),=CL31'/TRANSACTION/USERID RECORDS'
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT JOB         :'
         MVC   OUTREC+37(8),INPUT_JOB_NAME
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT TRANSACTION :'
         MVC   OUTREC+37(4),INPUT_TRANSACTION
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT USERID      :'
         MVC   OUTREC+37(8),INPUT_USERID
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT DATE        :'
         MVC   OUTREC+37(8),INPUT_DATE
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT START TIME  :'
         MVC   OUTREC+37(8),INPUT_START_TIME
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - EXTRACT END TIME    :'
         MVC   OUTREC+37(8),INPUT_END_TIME
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
*--------------------------------------------------------------------
*  CONVERT START/END TIME
*--------------------------------------------------------------------
*
         MVC   TIME_HHMMSS,INPUT_START_TIME
         BAL   R6,CONVERT_TIME_TO_BINARY
         ST    R7,CONVERTED_START_TIME
         MVC   TIME_HHMMSS,INPUT_END_TIME
         BAL   R6,CONVERT_TIME_TO_BINARY
         ST    R7,CONVERTED_END_TIME
*--------------------------------------------------------------------
*  MAIN PROCESSING LOOP
*--------------------------------------------------------------------
GET_NEXT_RECORD DS ØH
*
         GET   LMRKIN
         LA    R2,4(,R1)           LOOK BEYOND QSAM LLBB
         TM    Ø(R2),X'8Ø'         IS RECORD COMPRESSED?
         BNO   RECORD_ALREADY_DECOMPRESSED
*--------------------------------------------------------------------
*  CALL LANDMARK DECOMPRESSION ROUTINE
*--------------------------------------------------------------------
         ST    R2,PARMPRC@         POINT TO COMPRESSED RECORD
         LA    R1,PARMCPRS         PARAMETER LIST
         L     R15,=V($CRCPRS)     ADDRESS LANDMARK DECOMPRESSION PROG
         BALR  R14,R15             GO DECOMPRESS
         L     R2,PARMPEX@         POINT DECOMPRESSED RECORD
*
```

```
RECORD_ALREADY_DECOMPRESSED DS ØH
*
         USING TAMAIN,R2
         AP    COUNTER_TOTAL_RECORDS,=P'1'
*
CHECK_RECORD_TYPE DS ØH
         CLC   TMHDREC,=CL2'TA'
         BNE   GET_NEXT_RECORD
*
CHECK_JOBNAME DS ØH
         CLC   TMHDJOBC,INPUT_JOB_NAME
         BNE   GET_NEXT_RECORD
*
CHECK_TRANSACTION DS ØH
         CLC   INPUT_TRANSACTION,=CL8'XXXXXXXX'
         BE    CHECK_USERID
         CLC   TAPTRAN,INPUT_TRANSACTION
         BNE   GET_NEXT_RECORD
*
CHECK_USERID DS ØH
         CLC   INPUT_USERID,=CL8'XXXXXXXX'
         BE    CHECK_TIME_RANGE
         CLC   TAUSERID,INPUT_USERID
         BNE   GET_NEXT_RECORD
*
CHECK_DATE DS ØH
         CLC   TMHDCDAT,INPUT_DATE       CCYYMMDD
         BNE   GET_NEXT_RECORD
*
CHECK_TIME_RANGE DS ØH
         CLC   TMHDTIME,CONVERTED_START_TIME
         BL    GET_NEXT_RECORD
         CLC   TMHDTIME,CONVERTED_END_TIME
         BH    GET_NEXT_RECORD
*
         AP    COUNTER_SELECTED_RECORDS,=P'1'
*
WRITE_RECORD_TO_LMRKOUT DS ØH
*
         LH    R1,Ø(,R2)           GET RECORD LENGTH
         LA    R1,4(,R1)           ADD 4 BYTES FOR RDW
         S     R2,=F'4'            BACK-UP 4 BYTES FOR RDW
         STH   R1,Ø(R2)            AND MAKE INTO STANDARD RDW
         XC    2(2,R2),2(R2)       DITTO
         PUT   LMRKOUT,(R2)        OUTPUT EXPANDED RECORD
         B     GET_NEXT_RECORD
*--------------------------------------------------------------------
*  END OF JOB PROCESSING
*--------------------------------------------------------------------
EOJ      DS    ØH
```

```
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - TOTAL RECS PROCESSED:'
         ED    MASK_FIELD,COUNTER_TOTAL_RECORDS
         MVC   OUTREC+37(8),MASK_FIELD+8
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
*
         MVC   OUTREC+2(32),=CL32'TMTAEXTR - TOTAL RECS SELECTED :'
         ED    MASK_FIELD,COUNTER_SELECTED_RECORDS
         MVC   OUTREC+37(8),MASK_FIELD+8
         BAL   R6,WRITE_RECORD_TO_SYSPRINT
*
         CLOSE (SYSPRINT)
         CLOSE (LMRKIN)
         CLOSE (LMRKOUT)
         L     R13,REGISTER_SAVE_AREA+4
         RETURN (14,12),RC=Ø
*--------------------------------------------------------------------
*  SUBROUTINE:
*    WRITE TO SYSPRINT, CLEAR OUTREC AND PRIME MASK_FIELD
*--------------------------------------------------------------------
WRITE_RECORD_TO_SYSPRINT DS ØH
*
         PUT   SYSPRINT,OUTCARD
         MVI   OUTREC,C' '
         MVC   OUTREC+1(132),OUTREC
         MVC   MASK_FIELD,EDIT_MASK
         BR    R6                  RETURN TO CALLER
*--------------------------------------------------------------------
*  SUBROUTINE:
*    CONVERT START/END TIME TO FULLWORD 64 MICRO-SECOND UNITS
*--------------------------------------------------------------------
CONVERT_TIME_TO_BINARY DS ØH
*
         PACK  DOUBLE_WORD_WORK_1,TIME_HH
         MP    DOUBLE_WORD_WORK_1,PACKED_VALUE_36ØØ
         ZAP   DOUBLE_WORD_WORK_2,DOUBLE_WORD_WORK_1
         PACK  DOUBLE_WORD_WORK_1,TIME_MM
         MP    DOUBLE_WORD_WORK_1,PACKED_VALUE_6Ø
         AP    DOUBLE_WORD_WORK_2,DOUBLE_WORD_WORK_1
         PACK  DOUBLE_WORD_WORK_1,TIME_SS
         AP    DOUBLE_WORD_WORK_2,DOUBLE_WORD_WORK_1
         MP    DOUBLE_WORD_WORK_2,PACKED_VALUE_15625
         CVB   R7,DOUBLE_WORD_WORK_2
         BR    R6                  RETURN TO CALLER
*--------------------------------------------------------------------
*  STORAGE AREA
*--------------------------------------------------------------------
REGISTER_SAVE_AREA        DS 18F
*
MASK_FIELD                DS CL16
EDIT_MASK                 DC XL16'4020202020202020202020202020212Ø'
```

```
*
COUNTER_TOTAL_RECORDS     DC PL8'Ø'    TOTAL RECORDS
COUNTER_SELECTED_RECORDS  DC PL8'Ø'    SELECTED RECORDS
*
TIME_HHMMSS               DS ØD
TIME_HH                   DS XL2
                          DS XL1
TIME_MM                   DS XL2
                          DS XL1
TIME_SS                   DS XL2
*
PACKED_VALUE_36ØØ         DC PL4'36ØØ'
PACKED_VALUE_6Ø           DC PL4'6Ø'
PACKED_VALUE_15625        DC PL4'15625'  = 1ØØØØØØ / 64
DOUBLE_WORD_WORK_1        DS D
DOUBLE_WORD_WORK_2        DS D
*
CONVERTED_START_TIME      DS F
CONVERTED_END_TIME        DS F
*
INPUT_JOB_NAME            DS D
INPUT_TRANSACTION         DS D
INPUT_USERID              DS D
INPUT_DATE                DS D
INPUT_START_TIME          DS D
INPUT_END_TIME            DS D
*
OUTCARD                   DC    AL2(137),AL2(Ø)
OUTREC                    DC    CL133' '
         ORG   OUTREC+133
*
SYSIN    DCB   DDNAME=SYSIN,MACRF=GL,DSORG=PS,EODAD=@CLPRM,RECFM=FB
LMRKIN   DCB   DDNAME=LMRKIN,MACRF=GL,EODAD=EOJ,DSORG=PS,RECFM=VB
LMRKOUT  DCB   DDNAME=LMRKOUT,MACRF=PM,DSORG=PS,RECFM=VB
SYSPRINT DCB   DDNAME=SYSPRINT,DSORG=PS,MACRF=PM,                      X
               LRECL=137,BLKSIZE=137Ø,RECFM=VB
*-------------------------------------------------------------------
*  PARAMETER AREA FOR $CRCPRS (RECORD EXPANSION)
*-------------------------------------------------------------------
         CNOP  Ø,4
PARMCPRS DS    ØCL(3*4)
PARMPCMD DC    F'2'
PARMPRC@ DS    F
PARMPEX@ DS    F
*-------------------------------------------------------------------
*  TRANSACTION PERFORMANCE DETAIL (TA) RECORD LAYOUT
*-------------------------------------------------------------------
         TMCETA DSECT=YES,CRHDR=YES   TA MAIN PORTION
*
         END
```

TMTAEXTR ASSEMBLY AND LINK EDIT

```
//ASMLJOB  JOB
//ASSEM    EXEC PGM=ASMA90,
//         PARM='DECK,NOOBJECT,LIST,XREF(FULL),ALIGN'
//SYSLIB   DD DSN=TMONCICS.V20.TCESAMP,DISP=SHR
//         DD DSN=TMONCICS.V20.LMKSAMP,DISP=SHR
//         DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//SYSUT1   DD UNIT=TEMPO,SPACE=(1700,(400,400))
//SYSUT2   DD UNIT=TEMPO,SPACE=(1700,(400,400))
//SYSUT3   DD UNIT=TEMPO,SPACE=(1700,(400,400))
//SYSPUNCH DD DSN=&&LOADSET,
//            UNIT=TEMPO,DISP=(,PASS),
//            SPACE=(400,(100,100,1)),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=400)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DSN=TMTA.SOURCE(TMTAEXTR),DISP=SHR
//*
//LNK      EXEC PGM=IEWL,COND=(7,LT)
//SYSUT1   DD UNIT=TEMPO,SPACE=(1024,(100,10))
//SYSLIB   DD DSN=TMONCICS.V20.LMKAUTH,DISP=SHR
//SYSLMOD  DD DSN=USERID.LOADLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSLIN   DD DSN=&&LOADSET,DISP=(OLD,DELETE)
//         DD *
      ENTRY TMTAEXTR
      INCLUDE SYSLIB($CRCPRS)
      NAME TMTAEXTR(R)
//
```

TMTAEXTR EXECUTION JCL

```
//TMTAEXTR JOB
//*--------------------------------------------------------------------
//* Format of SYSIN:
//* Jobname  CL8
//* Tran     ØCL8            or CL8'XXXXXXXX'
//*          CL4
//*          XL4'00000000'   ====================> NB
//* User     CL8             or CL8'XXXXXXXX'
//* Date     CL8             CCYYMMDD
//* Start    CL8             HH:MM:SS
//* End      CL8             HH:MM:SS
//*--------------------------------------------------------------------
//*
//TMTAEXTR EXEC PGM=TMTAEXTR
//STEPLIB  DD DSN=USERID.LOADLIB,DISP=SHR
//LMRKIN   DD DSN=TMONCICS.V20.ARCHIVE,DISP=SHR
//LMRKOUT  DD DSN=USERID.TMTAEXTR,DISP=(,CATLG),
```

```
//            SPACE=(CYL,(5,1)),UNIT=SYSDA,
//            DCB=(RECFM=VB,LRECL=27994,BLKSIZE=27998)
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
CICS0001TRAN    USERID  19990920070725:0007:3000
//
```

TMTA REXX SOURCE

```
/*------------------------------ Rexx -------------------------------*/
/* Display TMCE archived and decompressed TA records in a similar    */
/* fashion to on-line TMON screens. Select to display every field in */
/* the main body of the record, then specify segment type to display */
/*-------------------------------------------------------------------*/
numeric digits 21
tranmask = 'XXXX'
termmask = 'XXXX'
address ispexec "libdef ispplib dataset id('your.ispf.panel.library')"
GET_DSN:
address ispexec "display panel(tmtap1)"
if rc ¬= 0 then do
  signal EXIT_PROG
end
address ispexec "vget (dsn)"
"ALLOC FI(TMTA) DA('"dsn"') SHR REUSE"
address ispexec "tbcreate tmtatab nowrite replace,
  names(k date time jobn tran tskn term resp cpu page ioct abnd)"
i = 0; done = 'n'
do until done = 'y'
  "execio 1 diskr tmta"
  if rc = 0 then do
    parse pull tmtarec
    sel = 'y'
    if tranmask = 'XXXX' then nop
    else
      if substr(tmtarec,53,length(tranmask)) = tranmask then nop
      else
        sel = 'n'
    if termmask = 'XXXX' then nop
    else
      if substr(tmtarec,69,length(termmask)) = termmask then nop
      else
        sel = 'n'
    if sel = 'y' then do
      i = i + 1
      tmrec.i = tmtarec
    end
    if i = 16 then do
      j = 16
      call proc_recs
```

```
    end
  end
  else do
    done = 'y'
    j = i
    call proc_recs
  end                                                  /* end if          */
end                                                    /* end do until    */
"execio Ø diskr tmta (finis"
"FREE FI(TMTA)"
signal GET_DSN
EXIT_PROG:
address ispexec "libdef ispplib"
exit
/* Process the records                                                     */
proc_recs:
do k = 1 to j
  tmnrec = tmrec.k
  call proc_ta
end
address ispexec "tbtop tmtatab"
tbdispl = 'y'
do until tbdispl = 'n'
  tbdispl = 'n'
  address ispexec "tbdispl tmtatab panel(tmtap2)"
  if rc ¬= Ø then do
    address ispexec "tbclose tmtatab"
    "execio Ø diskr tmta (finis"
    "FREE FI(TMTA)"
    signal GET_DSN
  end
  if exp = 'x' then do
    call exp_rec
    tbdispl = 'y'
  end
  exp = ''
end                                                    /* end do until    */
address ispexec "tbclose tmtatab"
address ispexec "tbcreate tmtatab nowrite replace,
  names(k date time jobn tran tskn term resp cpu page ioct abnd)"
i = Ø
return
/* Process TA record                                                       */
proc_ta:
date = substr(tmnrec,13,2)    '/'    substr(tmnrec,15,2)
hdus = c2d(substr(tmnrec,17,4))
hdus = hdus * 64
hdhh = format(hdus%(3600000000),2,Ø)
hdus = hdus-(hdhh*3600000000)
hdmm = format(hdus%(60000000),2,Ø)
hdus = hdus-(hdmm*60000000)
```

```
hdss = format(hdus%1000000,2,0)
time = hdhh    ':'    hdmm    ':'    hdss
time = translate(time,'0',' ')
jobn = substr(tmnrec,21,8)
tran = substr(tmnrec,53,4)
tskn = substr(c2x(substr(tmnrec,46,4)),1,7)+0
term = substr(tmnrec,69,4)
resp = format(c2d(substr(tmnrec,299,8))/1000000,5,3)
cpu  = format(c2d(substr(tmnrec,283,8))/1000000,5,3)
page = c2d(substr(tmnrec,467,4))
ioct = c2d(substr(tmnrec,435,4))
abnd = substr(tmnrec,171,4)
address ispexec "tbadd tmtatab"
return
/*--------------------------------------------------------------------*/
/* Process TOD                                                        */
/*--------------------------------------------------------------------*/
proc_tod:
sec  = c2d(tod) / (4096 * 1000 * 1000)
day  = sec % (24 * 60 * 60)
sec  = sec - (24 * 60 * 60 * day)
hr   = sec % (60 * 60)
sec  = sec - (60 * 60 * hr)
min  = sec % 60
sec  = sec - (60 * min)
hr   = format(hr,2,0)
min  = format(min,2,0)
sec  = format(sec,2,6)
ttod = hr    ':'    min    ':'    sec
ttod = translate(ttod,'0',' ')
return
/*--------------------------------------------------------------------*/
/* Expand a TA record                                                 */
/*--------------------------------------------------------------------*/
exp_rec:
call init_des
tmnrec  = tmrec.k
val.1   = substr(tmnrec,3,3)                    /* Product ID      */
val.2   = c2x(substr(tmnrec,6,1))               /* Global flg 1    */
val.3   = c2d(substr(tmnrec,7,2))               /* Ver & rel       */
val.4   = substr(tmnrec,9,8)                    /* Date rec prod   */
hdus    = c2d(substr(tmnrec,17,4))
hdus    = hdus * 64
hdhh    = format(hdus%(3600000000),2,0)
hdus    = hdus-(hdhh*3600000000)
hdmm    = format(hdus%(60000000),2,0)
hdus    = hdus-(hdmm*60000000)
hdss    = format(hdus%1000000,2,0)
hdus    = format(hdus-(hdss*1000000),6,0)
hdtime  = hdhh    ':'    hdmm    ':'    hdss    '.'    hdus
val.5   = translate(hdtime,'0',' ')             /* Time rec prod   */
```

```
val.6   = substr(tmnrec,21,8)                     /* Jobname         */
val.7   = substr(tmnrec,37,4)                     /* CICS sysid      */
val.8   = substr(tmnrec,41,4)                     /* CICS level      */
val.9   = c2d(substr(tmnrec,45,1))                /* Mon version     */
val.10  = substr(c2x(substr(tmnrec,46,4)),1,7)+0  /* Task number     */
val.11  = substr(tmnrec,50,3)                     /* Opid            */
val.12  = substr(tmnrec,53,4)                     /* Prim tranid     */
val.13  = substr(tmnrec,61,4)                     /* Orig tranid     */
val.14  = substr(tmnrec,69,4)                     /* Termid          */
val.15  = substr(tmnrec,73,4)                     /* Termid owning   */
val.16  = substr(tmnrec,77,4)                     /* Sysid owning    */
val.17  = substr(tmnrec,81,8)                     /* VTAM LU         */
val.18  = substr(tmnrec,89,8)                     /* VTAM Applid     */
val.19  = substr(tmnrec,97,8)                     /* Base program    */
val.20  = substr(tmnrec,105,4)                    /* SMF sysid       */
val.21  = substr(tmnrec,109,8)                    /* MVS imageid     */
val.22  = substr(tmnrec,117,8)                    /* Job id          */
val.23  = c2x(substr(tmnrec,133,2))               /* ASID            */
val.24  = substr(tmnrec,135,20)                   /* UOW luname      */
val.25  = c2x(substr(tmnrec,155,6))               /* UOW clock       */
val.26  = c2x(substr(tmnrec,161,2))               /* UOW seq num     */
val.27  = c2x(substr(tmnrec,163,8))               /* Local UOW       */
val.28  = substr(tmnrec,171,4)                    /* Abend code      */
val.29  = substr(tmnrec,175,8)                    /* Abend pgm       */
val.30  = substr(tmnrec,191,8)                    /* WLM srv cls     */
val.31  = substr(tmnrec,199,8)                    /* WLM rep cls     */
val.32  = substr(tmnrec,207,8)                    /* WLM wkld        */
val.33  = substr(tmnrec,215,8)                    /* Tran class      */
val.34  = substr(tmnrec,223,8)                    /* TMON group      */
val.35  = c2d(substr(tmnrec,231,4))               /* Tran priority   */
val.36  = c2x(substr(tmnrec,235,1))               /* Fac type        */
val.37  = c2d(substr(tmnrec,237,8))               /* GMT offset      */
val.38  = c2d(substr(tmnrec,245,4))               /* Task orig cnt   */
val.39  = substr(tmnrec,249,8)                    /* Userid          */
val.40  = c2x(substr(tmnrec,257,2))               /* EIP lim excd    */
val.41  = substr(c2x(substr(tmnrec,259,4)),1,7)+0 /* Start day       */
tod     = substr(tmnrec,263,8)                    /*                 */
call      proc_tod                                /*                 */
val.42  = ttod                                    /* Start time      */
val.43  = substr(c2x(substr(tmnrec,271,4)),1,7)+0 /* End day         */
tod     = substr(tmnrec,275,8)                    /*                 */
call      proc_tod                                /*                 */
val.44  = ttod                                    /* End time        */
val.45  = c2d(substr(tmnrec,283,8))/1000000       /* Real cpu        */
val.46  = c2d(substr(tmnrec,291,8))/1000000       /* Dbctl th        */
val.47  = c2d(substr(tmnrec,299,8))/1000000       /* Resp            */
val.48  = c2d(substr(tmnrec,307,4))               /* Resp count      */
val.49  = c2d(substr(tmnrec,311,8))/1000000       /* Wait            */
val.50  = c2d(substr(tmnrec,319,4))               /* Tran count      */
val.51  = c2d(substr(tmnrec,323,8))/1000000       /* Wait            */
val.52  = c2d(substr(tmnrec,339,8))/1000000       /* Disp            */
```

```
val.53  = c2d(substr(tmnrec,347,4))              /* Disp count      */
val.54  = c2d(substr(tmnrec,351,4))              /* Wmvs count      */
val.55  = c2d(substr(tmnrec,355,8))/1000000      /* Wmvs time       */
val.56  = c2d(substr(tmnrec,363,4))              /* Woldc count     */
val.57  = c2d(substr(tmnrec,367,8))/1000000      /* Woldc time      */
val.58  = c2d(substr(tmnrec,375,4))              /* Woldw count     */
val.59  = c2d(substr(tmnrec,379,8))/1000000      /* Woldw time      */
val.60  = c2d(substr(tmnrec,387,4))              /* Susp  count     */
val.61  = c2d(substr(tmnrec,391,8))/1000000      /* Susp  time      */
val.62  = c2d(substr(tmnrec,399,4))              /* Iwait count     */
val.63  = c2d(substr(tmnrec,403,8))/1000000      /* Iwait time      */
val.64  = c2d(substr(tmnrec,411,4))              /* Tcbsw count     */
val.65  = c2d(substr(tmnrec,415,8))/1000000      /* Tcbsw time      */
val.66  = c2d(substr(tmnrec,423,4))              /* Exwt  count     */
val.67  = c2d(substr(tmnrec,427,8))/1000000      /* Exwt  time      */
val.68  = c2d(substr(tmnrec,435,4))              /* Io    count     */
val.69  = c2d(substr(tmnrec,439,8))/1000000      /* Io    time      */
val.70  = c2d(substr(tmnrec,447,4))              /* Mro   count     */
val.71  = c2d(substr(tmnrec,451,8))/1000000      /* Mro   time      */
val.72  = c2d(substr(tmnrec,459,4))              /* Pageins         */
val.73  = c2d(substr(tmnrec,463,4))              /* Pageouts        */
val.74  = c2d(substr(tmnrec,467,4))              /* Totpage         */
val.75  = c2d(substr(tmnrec,471,4))              /* Tioa hwm        */
val.76  = c2d(substr(tmnrec,475,4))              /* U24 hwm         */
val.77  = c2d(substr(tmnrec,479,4))              /* U31 hwm         */
val.78  = c2d(substr(tmnrec,483,4))              /* C24 hwm         */
val.79  = c2d(substr(tmnrec,487,4))              /* C31 hwm         */
val.80  = c2d(substr(tmnrec,491,4))              /* Getmains        */
val.81  = c2d(substr(tmnrec,495,8))/1000000      /* St susp tim     */
val.82  = c2d(substr(tmnrec,503,4))              /* St susp cnt     */
val.83  = c2d(substr(tmnrec,507,8))/1000000      /* Teri suspwt     */
val.84  = c2d(substr(tmnrec,515,4))              /* Teri suspwc     */
val.85  = c2d(substr(tmnrec,519,4))              /* Term i  cnt     */
val.86  = c2d(substr(tmnrec,523,4))              /* Term i  len     */
val.87  = c2d(substr(tmnrec,527,4))              /* Term o  cnt     */
val.88  = c2d(substr(tmnrec,531,4))              /* Term o  len     */
val.89  = c2x(substr(tmnrec,535,1))              /* Term devtyp     */
val.90  = c2x(substr(tmnrec,536,1))              /* Tctte aid       */
val.91  = c2x(substr(tmnrec,537,1))              /* Flag 01         */
val.92  = c2x(substr(tmnrec,538,1))              /* Flag 02         */
val.93  = c2x(substr(tmnrec,539,1))              /* Flag 03         */
val.94  = c2x(substr(tmnrec,540,1))              /* Flag 04         */
val.95  = c2x(substr(tmnrec,541,1))              /* Flag 05         */
val.96  = c2x(substr(tmnrec,542,1))              /* Flag 06         */
val.97  = c2x(substr(tmnrec,543,1))              /* Flag 07         */
val.98  = c2x(substr(tmnrec,544,1))              /* Flag 08         */
val.99  = c2x(substr(tmnrec,545,1))              /* Flag 09         */
val.100 = c2x(substr(tmnrec,546,1))              /* Flag 10         */
val.101 = c2x(substr(tmnrec,547,1))              /* Flag 11         */
val.102 = c2x(substr(tmnrec,548,1))              /* Flag 12         */
val.103 = c2x(substr(tmnrec,549,1))              /* Flag 13         */
```

```
val.104 = c2x(substr(tmnrec,550,1))              /* Flag 14         */
val.105 = c2x(substr(tmnrec,551,1))              /* Flag 15         */
val.106 = c2x(substr(tmnrec,552,1))              /* Flag 16         */
val.107 = c2x(substr(tmnrec,553,1))              /* Flag 17         */
val.108 = c2x(substr(tmnrec,554,1))              /* Flag 18         */
val.109 = c2x(substr(tmnrec,555,1))              /* Flag 19         */
val.110 = c2x(substr(tmnrec,556,1))              /* Flag 20         */
val.111 = c2d(substr(tmnrec,557,8))/1000000      /* Treq time       */
val.112 = c2d(substr(tmnrec,565,4))              /* Treq count      */
val.113 = c2d(substr(tmnrec,569,8))/1000000      /* Treq wtime      */
val.114 = c2d(substr(tmnrec,577,4))              /* Treq wcount     */
val.115 = c2d(substr(tmnrec,581,8))/1000000      /* Freq time       */
val.116 = c2d(substr(tmnrec,589,4))              /* Freq count      */
val.117 = c2d(substr(tmnrec,593,8))/1000000      /* Freq wtime      */
val.118 = c2d(substr(tmnrec,601,4))              /* Freq wcount     */
val.119 = c2d(substr(tmnrec,605,8))/1000000      /* Dreq time       */
val.120 = c2d(substr(tmnrec,613,4))              /* Dreq count      */
val.121 = c2d(substr(tmnrec,617,8))/1000000      /* Dreq wtime      */
val.122 = c2d(substr(tmnrec,625,4))              /* Dreq wcount     */
val.123 = c2d(substr(tmnrec,629,8))/1000000      /* Preq time       */
val.124 = c2d(substr(tmnrec,637,4))              /* Preq count      */
val.125 = c2d(substr(tmnrec,641,8))/1000000      /* Preq wtime      */
val.126 = c2d(substr(tmnrec,649,4))              /* Preq wcount     */
val.127 = c2d(substr(tmnrec,653,8))/1000000      /* Jrreq time      */
val.128 = c2d(substr(tmnrec,661,4))              /* Jrreq count     */
val.129 = c2d(substr(tmnrec,665,8))/1000000      /* Jrreq wtime     */
val.130 = c2d(substr(tmnrec,673,4))              /* Jrreq wcount    */
val.131 = c2d(substr(tmnrec,677,8))/1000000      /* Tsreq time      */
val.132 = c2d(substr(tmnrec,685,4))              /* Tsreq count     */
val.133 = c2d(substr(tmnrec,689,8))/1000000      /* Tsreq wtime     */
val.134 = c2d(substr(tmnrec,697,4))              /* Tsreq wcount    */
val.135 = c2d(substr(tmnrec,701,8))/1000000      /* Tdreq time      */
val.136 = c2d(substr(tmnrec,709,4))              /* Tdreq count     */
val.137 = c2d(substr(tmnrec,713,8))/1000000      /* Tdreq wtime     */
val.138 = c2d(substr(tmnrec,721,4))              /* Tdreq wcount    */
val.139 = c2d(substr(tmnrec,725,8))/1000000      /* Gdreq time      */
val.140 = c2d(substr(tmnrec,733,4))              /* Gdreq count     */
val.141 = c2d(substr(tmnrec,737,8))/1000000      /* Gdreq wtime     */
val.142 = c2d(substr(tmnrec,745,4))              /* Gdreq wcount    */
val.143 = c2d(substr(tmnrec,749,8))/1000000      /* Spreq time      */
val.144 = c2d(substr(tmnrec,757,4))              /* Spreq count     */
val.145 = c2d(substr(tmnrec,761,8))/1000000      /* Spreq wtime     */
val.146 = c2d(substr(tmnrec,769,4))              /* Spreq wcount    */
val.147 = c2d(substr(tmnrec,773,8))/1000000      /* Rdreq time      */
val.148 = c2d(substr(tmnrec,781,4))              /* Rdreq count     */
val.149 = c2d(substr(tmnrec,785,8))/1000000      /* Rdreq wtime     */
val.150 = c2d(substr(tmnrec,793,4))              /* Rdreq wcount    */
val.151 = c2d(substr(tmnrec,797,8))/1000000      /* Fereq time      */
val.152 = c2d(substr(tmnrec,805,4))              /* Fereq count     */
val.153 = c2d(substr(tmnrec,809,8))/1000000      /* Fereq wtime     */
val.154 = c2d(substr(tmnrec,817,4))              /* Fereq wcount    */
```

```
val.155 = c2d(substr(tmnrec,821,8))/1000000       /* Sqlcl time      */
val.156 = c2d(substr(tmnrec,829,4))               /* Sqlcl count     */
val.157 = c2d(substr(tmnrec,833,8))/1000000       /* Db2nsql time    */
val.158 = c2d(substr(tmnrec,841,4))               /* Db2nsql cnt     */
val.159 = c2d(substr(tmnrec,845,8))/1000000       /* Db2    wtime    */
val.160 = c2d(substr(tmnrec,853,4))               /* Db2    wcnt     */
val.161 = c2d(substr(tmnrec,857,8))/1000000       /* Udb    rtime    */
val.162 = c2d(substr(tmnrec,865,4))               /* Udb    rcnt     */
val.163 = c2d(substr(tmnrec,869,8))/1000000       /* Wlm ex resp     */
val.164 = c2d(substr(tmnrec,877,4))               /* Wlm percent     */
val.165 = c2x(substr(tmnrec,881,1))               /* Wlm tgt type    */
val.166 = c2x(substr(tmnrec,882,1))               /* Wlm importnc    */
val.167 = substr(tmnrec,883,4)                    /* Db2 subsys      */
val.168 = substr(tmnrec,887,16)                   /* User data       */
val.169 = c2d(substr(tmnrec,903,4))               /* Tcb switches    */
val.170 = c2d(substr(tmnrec,907,8))/1000000       /* Tcb sw time     */
val.171 = c2d(substr(tmnrec,915,4))               /* Mqseries cnt    */
val.172 = c2d(substr(tmnrec,919,8))/1000000       /* Mqseries tim    */
val.173 = c2d(substr(tmnrec,927,4))               /* Mqseries cnt    */
val.174 = c2d(substr(tmnrec,931,8))/1000000       /* Mqseries tim    */
tod     = substr(tmnrec,939,8)                    /*                 */
call      proc_tod                                /*                 */
val.175 = ttod                                    /* Norm intv       */
val.176 = substr(tmnrec,947,8)                    /* Dbctl regn      */
val.177 = c2x(substr(tmnrec,1125,2))              /* Fat offset      */
val.178 = c2x(substr(tmnrec,1127,2))              /* Fat length      */
val.179 = c2d(substr(tmnrec,1129,2))              /* Fat number      */
val.180 = c2x(substr(tmnrec,1131,2))              /* Uvl offset      */
val.181 = c2x(substr(tmnrec,1133,2))              /* Uvl length      */
val.182 = c2d(substr(tmnrec,1135,2))              /* Uvl number      */
val.183 = c2x(substr(tmnrec,1137,2))              /* Utg offset      */
val.184 = c2x(substr(tmnrec,1139,2))              /* Utg length      */
val.185 = c2d(substr(tmnrec,1141,2))              /* Utg number      */
val.186 = c2x(substr(tmnrec,1143,2))              /* Mro offset      */
val.187 = c2x(substr(tmnrec,1145,2))              /* Mro length      */
val.188 = c2d(substr(tmnrec,1147,2))              /* Mro number      */
val.189 = c2x(substr(tmnrec,1149,2))              /* Req offset      */
val.190 = c2x(substr(tmnrec,1151,2))              /* Req length      */
val.191 = c2d(substr(tmnrec,1153,2))              /* Req number      */
val.192 = c2x(substr(tmnrec,1155,2))              /* Wait offset     */
val.193 = c2x(substr(tmnrec,1157,2))              /* Wait length     */
val.194 = c2d(substr(tmnrec,1159,2))              /* Wait number     */
val.195 = c2x(substr(tmnrec,1161,2))              /* Tcbu offset     */
val.196 = c2x(substr(tmnrec,1163,2))              /* Tcbu length     */
val.197 = c2d(substr(tmnrec,1165,2))              /* Tcbu number     */
```

Editor's note: this article will be concluded next month.

Patrick Mullen
System Software Consultant (Canada)

H&W Computer Systems has announced Theme Manager for CICS, which extends the capabilities of CICS Transaction Server (TS) by allowing corporate presentation Web elements (company logos, hotlinks, animated graphics, etc) called 'Themes' to be created with Web development tools. Once developed, the Themes are uploaded to the mainframe. At run-time, CICS application output and Themes are merged and served directly from CICS TS to the Web.

Many applications can share the same Theme and different Themes can be presented to different users based on user-definable rules. Integration flexibility is extended by allowing other HTML sources to be merged automatically with the CICS application and the Theme(s) on the final Web page that is presented to the user.

Theme Manager for CICS supports non-traditional file formats within CICS, such as GIFs, JPEGs, PDF, JavaScript, HTML, etc.

For further information contact:
H&W Computer Systems, PO Box 46019, Boise, ID 83711, USA.
Tel: (208) 377 0336.
URL: http://www.hwcs.com/cicsweb/theme_manager2.html.

* * *

Software AG has launched its TIEMA mainframe CICS adapter for Microsoft's Host Integration Server 2000. Using the COM Transaction Integrator, TIEMA is said to give access to 3270-based CICS transactions from Windows 2000 or NT systems.

This approach is said to eliminate the need for a 3270 emulator while enabling the co-existence of both 3270 and Windows devices for application access.

For further information contact:
Software AG, Uhlandstr 12, 64297 Darmstadt, Germany.
Phone: (6151) 920.
URL: http://www.softwareag.com/corporat/news/july2000/tiema.htm.

* * *

PeerLogic has introduced its Java CICS (JCICS) software, which directly links production run-time CICS applications with Java on open systems. As a new component within LiveContent TRANS, it can access the locked information from mainframe applications through an open systems platform.

It allows reuse of legacy assets in new Java applications on open systems servers and migrates old COBOL code and business rules to a Unix server with LiveContent TRANS. It also enables integration of existing enterprise systems with new Java and EJB applications.

CICS applications can be directly called from Java programs for full access to all the process logic and data in these legacy assets. This enables creation of simple Java programs that call CICS applications and extends the existing CICS applications by calling Java programs, exposing methods that can be accessed by Java and integrated via IIOP interfaces.

For further information contact:
PeerLogic, 555 De Haro Street, San Francisco, CA 94107-2348, USA.
Tel: (415) 626 4545.
URL: http://www.peerlogic.com/news/javacics.html.