206

CICS update

January 2003

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 December 1999 issue, are available separately to subscribers for £16.00 ($24.00) each including postage.

CICS Update on-line

Code from *CICS Update*, and complete issues in Acrobat PDF format, can be downloaded from our Web site at http://www.xephon.com/cics; you will need to supply a word from the printed issue.

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

When Xephon is given copyright, 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 download a copy of our *Notes for Contributors* from www.xephon.com/nfc.

Printed in England.

Create COBOL copy from a BMS source

The standard IBM method used to create file descriptions, or copybooks, from a BMS source is to assemble it specifying PARM=SYSPARM(DSECT).

However, the code generated is far from elegant. The names are misaligned and the syntax varies from 'PIC' to 'PICTURE' – why, I never understood. And I also never accepted those copybooks as produced. In the early days of my life as a programmer, I fixed them by hand, aligning things and standardizing the syntax to PIC. Later, I created an XEDIT REXX macro to do that work automatically, and finally I dropped the assembled output and wrote an EXEC to create the copybook directly from the BMS source.

The EXEC presented here is the MVS version. It uses an ISPF panel, shown below, for the specification of the input and output files, and a few other things I always felt would be useful.

```
+-------------------- Create Cobol copy from BMS --------------------+
|                                                                     |
|   Input BMS...: SOURCE.CICS7.BMS(BDBTPR1)                           |
|   Output COPY.: SOURCE.CICS7.COPY(BDBTPR1)                          |
|                                                                     |
|   Default COBOL levels.........: Ø2   Ø4   Ø6                       |
|                                                                     |
|   Choose model (Full or Short).: S                                  |
|                                                                     |
|    Full model                        Short model                    |
|      Ø4   FIELDL PIC S9(4) COMP        Ø4   FIELDL PIC S9(4) COMP   |
|      Ø4   FIELDF PIC X                 Ø4   FIELDF PIC X            |
|      Ø4   FILLER REDEFINES FIELDF      Ø4   FIELDA REDEFINES FIELDF |
|        Ø6 FIELDA PIC X                 Ø4   FIELDI PIC X...         |
|      Ø4   FIELDI PIC...                                             |
+---------------------------------------------------------------------+
```

The default COBOL levels 01, 02, and 03 are never used by me, because a copybook is normally part of a larger structure (a COMMAREA, most of the time) and so the levels must be different. For that reason, my EXEC proposes as default levels 02, 04, and 06, but one can change that on the entry panel.

Each BMS labelled field is turned into three COBOL fields, suffixed L, F (or A), and I (or O). L is the length field, with a two-byte binary picture. F or A refer to the same byte (the attribute byte with an alphanumeric picture), and I or O refer to the input or output (again the same area, but that can have different pictures). For me, the strange part is the way A redefines F with a FILLER in between that does not seem to have any purpose. I consider that FILLER completely useless, and I always discard it. So, my EXEC gives you the choice of two models. There's the *Full* model, which does things the IBM way, only more cleanly and aligned. There's the Short model, which drops the FILLER, and A redefines F directly. Both models are illustrated in the input panel – the default is the short model.

The program supports PICIN and PICOUT specifications, as well as OCCURS. Besides that, it also adds a commented line at the top of the copy with the number of bytes it contains. This can be useful for COMMAREA length calculations.

Below is a small example of the code produced in the short model:

```
    02   BTBS020I.
*                                        === 96 ===
       04    FILLER       PIC X(12).
       04    CODIGL       PIC S9(4) COMP.
       04    CODIGF       PIC X.
       04    CODIGA    REDEFINES CODIGF PIC X.
       04    CODIGI       PIC X(10).
       04    CAB1L        PIC S9(4) COMP.
       04    CAB1F        PIC X.
       04    CAB1A     REDEFINES CAB1F PIC X.
       04    CAB1I        PIC 99999.
       04    DESIG1L      PIC S9(4) COMP.
       04    DESIG1F      PIC X.
       04    DESIG1A   REDEFINES DESIG1F PIC X.
       04    DESIG1I      PIC X(60).

    02   BTBS020O  REDEFINES BTBS020I.

       04    FILLER       PIC X(12).
       04    FILLER       PIC X(03).
       04    CODIGO       PIC X(10).
       04    FILLER       PIC X(03).
       04    CAB1O        PIC X(05).
       04    FILLER       PIC X(03).
```

```
           Ø4     DESIG1O      PIC X(6Ø).
```

Be aware that the BMS source must be syntactically correct. My program does not check for missing commas, wrong continuation lines, and other source of problems. If the BMS contains syntax errors, the produced copy may be inaccurate. My advice is to assemble the BMS to CICS prior to creating the copy, in order to ensure that it contains no errors.

BMSCOPY REXX SOURCE

```
/* REXX / TSO *=========================================================*/
/*    BMSCOPY creates a COBOL COPY from a BMS source.                  */
/*    Uses same name ISPF panel.                                       */
/*=====================================================================*/
model = "S"                                  /* default model short */
L1    = "Ø2"                                 /* default levels      */
L2    = "Ø4"
L3    = "Ø6"

arg infile .
infile = strip(infile,,"'")
call display_panel

field0 = "              "L2"     FILLER       PIC X(12)."
field1 = "              "L2"     FILLER       PIC X(Ø3)."
field2 = "              "L2"     FILLER    REDEFINES"
field3 = "                 "L3"  FILLER       PIC X(Ø2)."
level1 = "          "L1"   "
level2 = "              "L2"    "
level3 = "                 "L3" "
x = Ø
k = Ø
length = 12
if L1 = "Ø1" then level1 = substr(level1,4)
actual = ""

do alpha = 1 to 9999
   execio 1 diskr ddin
   if rc = Ø then parse pull linha
   else do
      call close_ddin
      leave alpha
   end
   if left(linha,1) = "*" then do
      iterate alpha
   end
   p = pos("DFHMDI",linha)
   if p > Ø then do
```

```
   mapname = left(linha,8)
   mapname = space(mapname,Ø)
   iterate alpha
end
p = pos("DFHMDF",linha)
if p > Ø then do
   if left(linha,1) = " " then iterate alpha
   actual = left(linha,71)
end
continue = substr(linha,72,1)
if actual <> "" then do
   if p = Ø then actual = actual || substr(linha,16)
   if continue = " " then do
      x = x + 1
      name.x = left(actual,8)
      name.x = space(name.x,Ø)
      len = pos("LENGTH",actual) + 7
      len = substr(actual,len)
      len = strip(len)
      parse var len leng.x "," .
      pic = pos("PICIN",actual)
      if pic > Ø then do
         pic = pic + 6
         pic = substr(actual,pic)
         parse var pic "'" picin.x "'"
      end
      else do
         picin.x = "X("right(leng.x,2,'Ø')")"
      end
      pic = pos("PICOUT",actual)
      if pic > Ø then do
         pic = pic + 7
         pic = substr(actual,pic)
         parse var pic "'" picout.x "'"
      end
      else do
         picout.x = "X("right(leng.x,2,'Ø')")"
      end
      occ = pos("OCCURS",actual)
      if occ > Ø then do
         occ = occ + 7
         occ = substr(actual,occ)
         occ = strip(occ)
         parse var occ occur.x "," .
      end
      else do
         occur.x = 1
      end
      length = length + (leng.x + 3) * occur.x
      actual = ""
```

```
      end
   end
end alpha

dropbuf
queue level1 mapname"I."
queue "      *" copies(" ",42) "=== " length " ==="
queue field0
do y = 1 to x
   nametem1 = name.y"L"
   nametem1 = left(nametem1,12)
   nametem2 = name.y"F"
   nametem2 = left(nametem2,12)
   nametem3 = name.y"A"
   nametem3 = left(nametem3,12)
   nametem4 = name.y"I"
   nametem4 = left(nametem4,12)
   nametem5 = name.y"O"
   nametem5 = left(nametem5,12)
   if occur.y = 1 then do
      queue level2 nametem1 "PIC S9(4) COMP."
      queue level2 nametem2 "PIC X."
      if model = "F" then do
         queue field2 name.y"F."
         queue level3 nametem3 "PIC X."
      end
      if model = "S" then do
         queue level2 left(nametem3,9) "REDEFINES "name.y"F PIC X."
      end
      queue level2 nametem4 "PIC" picin.y"."
   end
   else do
      nametem6 = name.y"D"
      nametem6 = left(nametem6,9)
      queue level2 nametem6 "OCCURS "occur.y" TIMES."
      queue level3 nametem1 "PIC S9(4) COMP."
      queue level3 nametem2 "PIC X."
      queue level3 nametem4 "PIC" picin.y"."
   end
end
queue " "
queue level1 mapname"O  REDEFINES "mapname"I."
queue " "
queue field0
do y = 1 to x
   nametem5 = name.y"O"
   nametem5 = left(nametem5,12)
   if occur.y = 1 then do
      queue field1
      queue level2 nametem5 "PIC" picout.y"."
```

```
   end
   else do
      k = k + 1
      nametem3 = name.y"A"
      nametem3 = left(nametem3,12)
      nametem7 = "DFHMS"k
      nametem7 = left(nametem7,9)
      queue level2 nametem7 "OCCURS "occur.y" TIMES."
      queue field3
      queue level3 nametem3 "PIC X".
      queue level3 nametem5 "PIC" picout.y"."
   end
end

execio queued() diskw ddout "(finis"
call close_ddout
exit
/*================================================================*/
/*  Subroutines                                                   */
/*================================================================*/
display_panel:
 M = model
 address ispexec
 'addpop row(1) column(1)'
 'display panel(bmscopy)'
 if rc=8 then exit
 'rempop'
 address tso
 call alloc_file infile  ddin
 call alloc_file outfile ddout
 model = M
return
/*================================================================*/
/*  Allocate input and output files                               */
/*================================================================*/
alloc_file:
 arg file dd
 xx = msg(off)
 "free  dd("dd")"
 "alloc dd("dd") da('"file"') shr"
 if rc <>Ø then do
    say "Error allocating" file rc
    exit
 end
 xx = msg(on)
return
/*================================================================*/
/*  Close and deallocate files                                    */
/*================================================================*/
close_ddin:
```

```
 execio Ø diskr ddin "(finis"
 "free dd(ddin)"
return

close_ddout:
 execio Ø diskw ddout "(finis"
 "free dd(ddout)"
return
```

BMSCOPY PANEL SOURCE

```
)ATTR
  _ TYPE(INPUT) CAPS(ON) JUST(LEFT)  COLOR(RED)
  # TYPE(INPUT) CAPS(ON) JUST(RIGHT) COLOR(RED)
  ? TYPE(TEXT) INTENS(HIGH) SKIP(ON) COLOR(PINK)
  % TYPE(TEXT) INTENS(HIGH) SKIP(ON) COLOR(YELLOW)
  $ TYPE(TEXT) INTENS(LOW)  SKIP(ON) COLOR(BLUE)
  + TYPE(TEXT) INTENS(LOW)  SKIP(ON) COLOR(GREEN)
  ! TYPE(OUTPUT) CAPS(OFF)  SKIP(ON) COLOR(WHITE)
)BODY WINDOW(7Ø,18)
+
?  Input BMS...:_INFILE                                    +
?  Output COPY.:_OUTFILE                                   +
?
?  Default COBOL levels.........:_L1+ _L2+ _L3+
?
%  Choose model (Full or Short).:_M
?
%   Full model                       Short model
+     Ø4   FIELDL PIC S9(4) COMP       Ø4   FIELDL PIC S9(4) COMP
+     Ø4   FIELDF PIC X                Ø4   FIELDF PIC X
+     Ø4   FILLER REDEFINES FIELDF     Ø4   FIELDA REDEFINES FIELDF
+       Ø6 FIELDA PIC X                Ø4   FIELDI PIC X...
+     Ø4   FIELDI PIC...
+
+
)INIT
&ZWINTTL = 'Create COBOL copy from BMS'
)PROC
&Mver='F S'
VER(&infile,nonblank,dsname)
VER(&outfile,nonblank,dsname)
VER(&L1,nonblank,num)
VER(&L2,nonblank,num)
VER(&L3,nonblank,num)
VER(&M,nonblank,listv,&Mver)
)END
```

CICSPlex SM dynamic workload management – balancing

In the last two articles (see *CICS Update* Issues 204 and 205, November and December 2002 – *CICS dynamic workload management – concepts* and *CICS and the CICSPlex SM dynamic workload management model*), we have talked about the factors affecting dynamic workload balancing, and the techniques used for affinity management, health detection, and abend avoidance. This article will discuss the load-balancing algorithms provided by CICSPlex SM, namely queue and goal. The techniques used both involve calculating a weight for each candidate target region for this work request. The region with the lowest weight becomes the target selected.

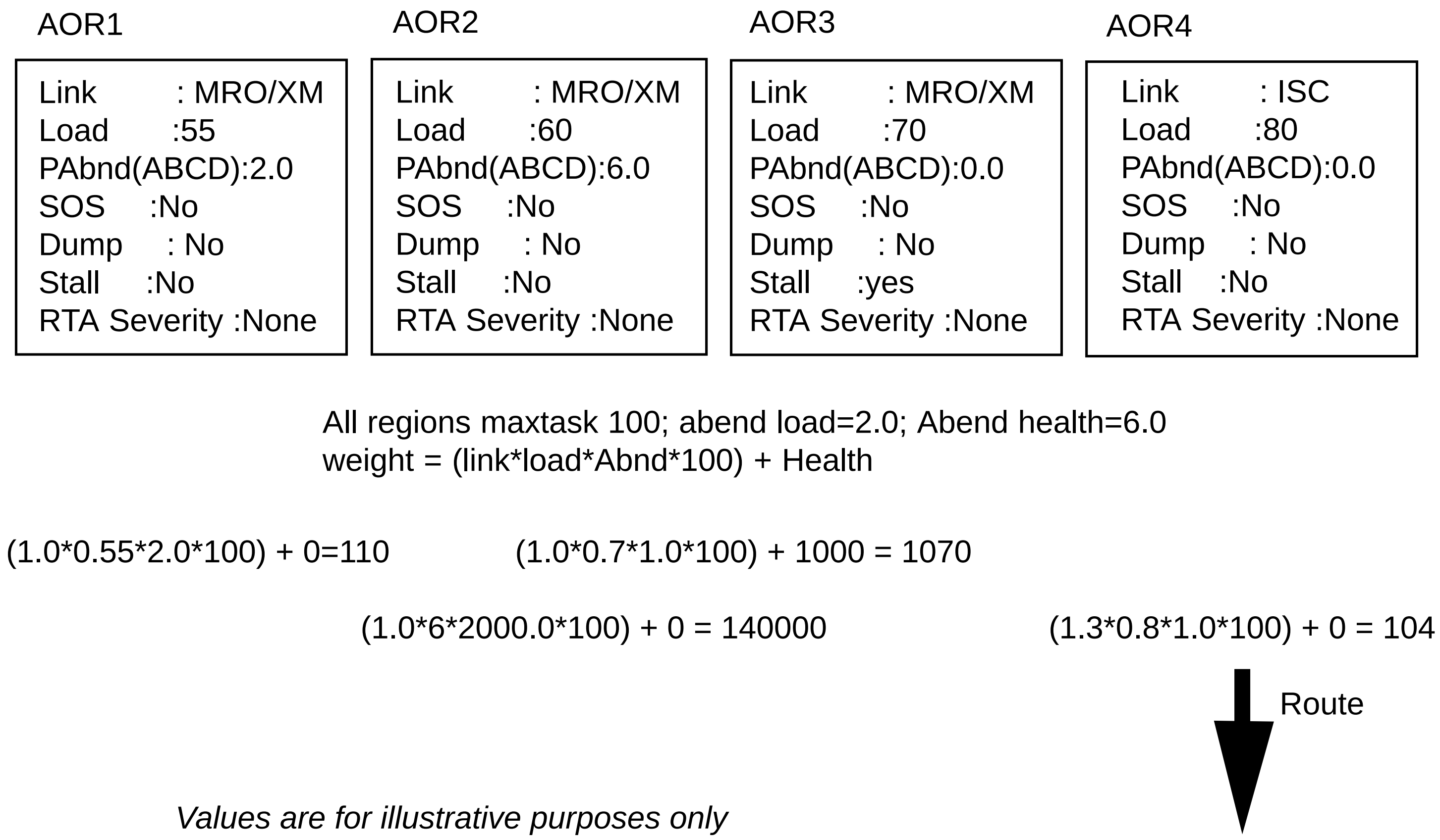
It should be noted that CICSPlex SM makes use of the no queue option provided by CICS. When a candidate is selected, CICSPlex SM returns the target and specifies noqueue. This means that if no sessions are available to that target region, CICS will recall CICSPlex SM immediately to suggest an alternative target, rather than wait for a session to become available. The next lowest weighted target is then tried. If all potential targets return with session not available, CICSPlex SM will finally try the system with the lowest weight again using the queue mechanism.

QUEUE-BASED BALANCING

CICSPlex SM provides a relatively simple balancing algorithm called queue. It calculates weights for each candidate region based on the health criteria and abend factor described in the previous article, along with two further values – normalized load and linkType.

- Normalized load – currentTask/maxTask.
- LinkType:
  - Two sets of values based on percentage load – one for

Figure 1: Queue based balancing



low, one for high.

- Local.
- MRO.
- MRO/XCF.
- ISC.

The weight for each region is calculated using the following formula:

```
weight=(normalisedLoad*linkType*abendFactor*100) + healthFactor
```

The region with the lowest weight becomes the target. In the event of a tie, one of the regions with the lowest weight is randomly chosen. An example of this process is shown in Figure 1.

GOAL-BASED BALANCING

CICSPlex SM also provides a balancing algorithm, which works in conjunction with MVS workload manager. In order to use this algorithm, all MVS images where the CICS regions reside must be in goal mode.

MVS workload manager definitions are maintained via base TSO facilities, not CICSPlex SM. MVS workload manager provides the ability to define a serviceDefinition (one per sysplex). A serviceDefinition can contain several servicePolicies (one of which can be active). A servicePolicy contains workloadDefinitions, serviceClasses, reportClasses, goals, and workloadClassifications.

A goal can be a velocity goal (typically used during address space initialization); discretionary, or response time. Response time goals can be either average or percentile. CICSPlex SM supports only average response time goals.

MVS WLM also provides definition capabilities for classification

rules (what goal to use with a piece of work). For CICS the classification scheme is:

```
(subsystemtype,subsysteminstance,userid,trangroup,Luname)-> serviceclass
```

In order to understand goal-based balancing a new concept must be introduced – that of a Performance Index (PI). The PI of a service class is simply the ratio of the actualAverageResponseTime to the averageResponseTimeGoal. Service classes with a PI of less than 1 are exceeding their goal; those greater than 1 are missing their goal. The aim of goal based routing is to steal resource from service classes exceeding their goal to give to service classes that are missing their goal. Hopefully, the result will be a system with all PIs below or equal to 1 (ie exceeding or achieving their goal).

$$PI_{ServiceClassn} = averageResponseTime_{ServiceClassn} / averageResponseTimeGoal_{ServiceClassn}$$

The goal-based algorithm takes into account performance indexes, transaction arrival rates, and activity of service classes. Service classes that have had no transaction arrivals for a while are removed from the active service class list and eliminated. In order to take more account of the recent past, both averageResponseTimes and arrival rates (AR) are exponentially faded using the following formula.

```
Value = value*alpha + (1-alpha)*value
```

Obviously all of this calculation (and more, as we shall see) would cause a large overhead if it were inline with a routing call. Obtaining a goal for a service class from MVS WLM, allocating targets to service classes, and calculating various values is all performed in parallel to routing requests (actually in the CMAS). Response-time goals are cached by CICSPlex SM and reused. MVS WM provides a signalling mechanism to detect when goals have been changed. CICSPlex SM listens for such events and clears its cache when such an event occurs. Furthermore, using MVS WLMs Performance Indices would also be prohibitive.

CICSPlex SM therefore calculates its own, working in a sympathetic relationship with MVS WLM. To understand this, we need to learn a little about MVS WLM.

MVS WLM calculates Performance Indexes for MVS address spaces. For an address space that has only one service class running within it, this is simple. However, for address spaces like CICS, which sub-dispatch, several service classes can be running in the same address space (CICS allocates maxtask MVS WLM Performance Blocks during CICS initialization, which are used for managing workloads by MVS WLM).

Since MVS WLM can allocate only storage and dispatch priority for an address space in order to attain the desired goal, a mechanism is needed for reducing the set of PIs in such address spaces to a single value. MVS WLM's mechanism is to calculate individual PIs and then choose the worst PI as the address space's PI. By default, we would end up with each address space having a spectrum of PI's, ending up with every address space having an address space PI about the same. This would give MVS WLM no clear 'Peter' to rob to 'pay Paul'. The trick that CICSPlex SM plays is to allocate service classes with similar PI to a given address space. MVS WLM therefore has a simple choice to make. It is via this relationship, not via a program call, that CICSPlex SM and MVS WLM work in partnership.

Allocation of service classes to targets

The first stage of the algorithm is to allocate targets to service classes. This is achieved by creating a list of targets on each MVS image (sorted by sysid), and a corresponding list of service classes (sorted by PI). Allocation of a service class to a (set of) targets(s) is then performed. Since the number of targets on each MVS image could be different, normalized values are calculated and then scaled to the number of targets on a given image. Given an arrival rate of AR:

First the percentage of targets to be allocated to a given service class is calculated:

$$target\%_{serviceclassn} = PI_{serviceclassn} * AR_{serviceclassn} / (\Sigma_{serviceclass}(PI_{serviceclassn} * AR_{serviceclassn}))$$

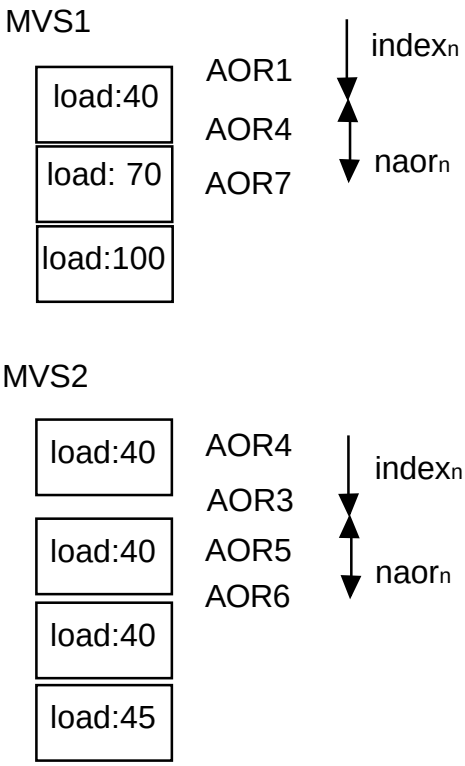


Figure 2: Goal indices and allocated AORs

From these values, a normalized depth can be calculated (starting at 0). It is merely a convenience for the following calculations:

$$depth_1 = \emptyset$$

$$depth_{serviceclassn} = depth_{serviceclassn-1} + target\%_{serviceclassn}$$

Now that we have these values, the number of targets allocated, an array index is easily calculated by multiplying the above by the number of targets on that MVS image:

$$Ntarget_{serviceclassn} = ROUNDUP\ (target\%_{serviceclassn} * Ntarget)$$

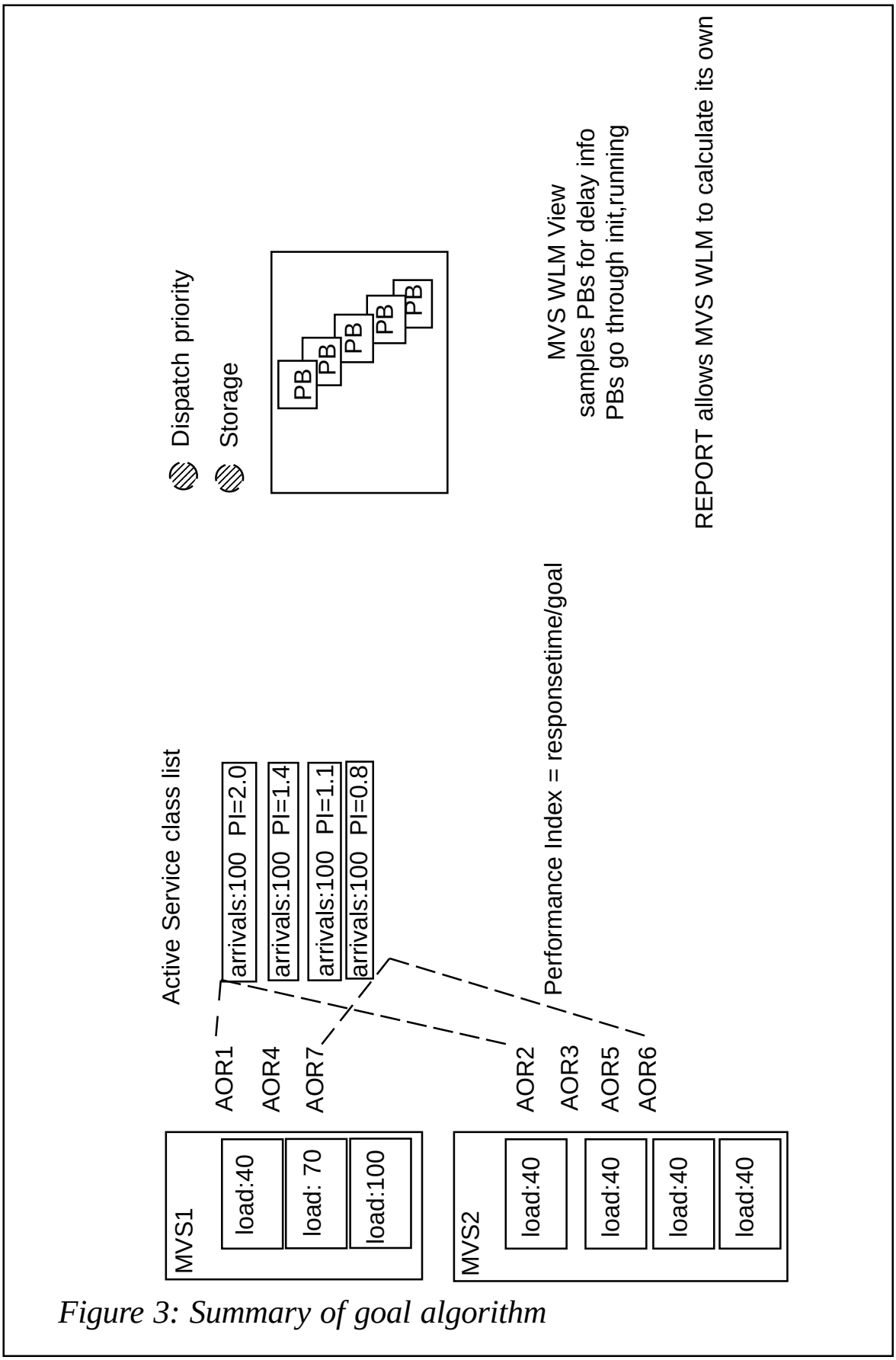


Figure 3: Summary of goal algorithm

$$Index_{serviceclassn} = TRUNCATE(depth_{serviceclassn} * Ntarget)$$

These values are illustrated in Figure 2.

Now we have calculated all the necessary values, we can get on with choosing a target.

SELECTION OF TARGET

First the MVS image is selected. It is the MVS image with the lowest load:

$$MVSLoad = \Sigma(curtask_n / maxtask_n)$$

where summation is over $Ntarget_{serviceclassn}$ starting at $Index_{serviceclassn}$.

The array of targets in each MVS image are then sorted according to normalized load. The best region is the one with the lowest load on the lowest loaded MVS image. We now have a series of targets sorted by load on each MVS image that are allocated to the service class. We now allocate a goal weight to the best region, a higher weight to the next and so on. When we have finished with allocated targets, we continue this sequence with the non-allocated targets. Finally we calculate the target weight in a similar fashion to queue:

```
Weight = (goalWeight*linkType*abend*100) + health
```

The region with the lowest weight is chosen as target. A summary of this processing is shown in Figure 3.

NEXT ARTICLE

We have now seen how CICS and CICSPlex SM provide dynamic workload management capabilities, and how affinity management, abend avoidance and balancing are achieved. In the next article (in next month's issue of *CICS Update*) we will look at the various types of workload that can be managed in this way, and how administratively the workload criteria are defined.

Dr Paul Johnson
CICS Transaction Server Systems Management Planning/Development
IBM (UK)

Browse and edit files under CICS – part 2

This month we conclude the code for a tool that allows users to browse through the keys of a KSDS file and to select one record to view its contents and, optionally, to modify it.

```
 DATASET-READ.
*=============*
     MOVE LOW-VALUES TO FILE-IN
     EXEC CICS IGNORE CONDITION LENGERR
     END-EXEC
     EXEC CICS HANDLE CONDITION NOTOPEN(ERRO-NOTOPEN)
     END-EXEC
     EXEC CICS READ DATASET (DDNAMEØI)
                    RIDFLD  (CURRKEY)
                    INTO    (FILE-IN)
                    LENGTH  (RECLEN)
     END-EXEC.
     DIVIDE RECLEN  BY 256 GIVING PAG-MAX  REMAINDER REMBYTES
     DIVIDE REMBYTES BY 16 GIVING REMLINES REMAINDER REMCHAR
     MULTIPLY REMCHAR BY 2 GIVING REMCHAR2
     IF REMCHAR > Ø
        ADD 1 TO REMLINES
     ELSE
        MOVE 99 TO REMCHAR REMCHAR2
     END-IF
     IF REMBYTES = Ø
        MOVE 256 TO REMBYTES
        MOVE 17  TO REMLINES
     ELSE
        ADD 1 TO PAG-MAX
     END-IF.
*
 DATASET-WRITE.
*==============*
     EXEC CICS IGNORE CONDITION LENGERR
     END-EXEC
     EXEC CICS HANDLE CONDITION ILLOGIC (ERRO-ILLOGIC)
     END-EXEC
     EXEC CICS READ DATASET (DDNAMEØI)
                    RIDFLD  (CURRKEY)
                    INTO    (FILE-IN)
                    LENGTH  (RECLENDUMMY)
                    UPDATE
     END-EXEC
     EXEC CICS REWRITE DATASET (DDNAMEØI)
                       FROM    (FILE-IN)
```

```
                        LENGTH  (RECLEN)
     END-EXEC.
*
 CHECK-LINES.
*============*
     IF DTHEXAL(K) > Ø
        PERFORM CHECK-HEXCHARS-LINE
                VARYING R FROM 1 BY 1 UNTIL R > 32
     END-IF
     IF DTCHARL(K) > Ø OR DTHEXAL(K) > Ø
        MOVE Ø TO DTCHARL(K) DTHEXAL(K)
        MOVE 1 TO RECALTERED
        PERFORM CHECK-LINES-EACH-CHAR
                VARYING X FROM 1 BY 1 UNTIL X > 16
        MOVE FILECO16(K)  TO  DTCHARI(K)
        MOVE FILEHO16(K)  TO  DTHEXAI(K)
     END-IF.
*
 CHECK-LINES-EACH-CHAR.
*======================*
     IF PAG = PAG-MAX AND K = REMLINES AND X > REMCHAR
        MOVE SPACES TO FILECO161(K, X) FILEHO161(K, X)
     ELSE
        IF DTHEXAI1(K, X) NOT EQUAL LOW-VALUE AND
                          NOT EQUAL FILEHO161(K, X)
           MOVE DTHEXAI1(K, X)   TO FILEHO161(K, X)
           PERFORM TRANSLATE-TO-CHAR-LINE
                   VARYING Y FROM 1 BY 1 UNTIL Y > 256
           MOVE FILECO161(K, X)  TO  DTCHARI1(K, X)
        END-IF
        IF DTCHARI1(K, X) NOT EQUAL LOW-VALUE AND
                          NOT EQUAL FILECO161(K, X)
           MOVE DTCHARI1(K, X)   TO FILECO161(K, X)
                                    FILEC161(PAG, K, X)
           PERFORM TRANSLATE-TO-HEXA-LINE
                   VARYING Y FROM 1 BY 1 UNTIL Y > 256
           MOVE FILEHO161(K, X)  TO  DTHEXAI1(K, X)
        END-IF
     END-IF.
*
 TRANSLATE-TO-CHAR-LINE.
*=======================*
     IF FILEHO161(K, X) = HEXTAB(Y)
        MOVE  CHARTABD(Y) TO FILECO161(K, X)
        MOVE  CHARTABF(Y) TO FILEC161(PAG, K, X)
        MOVE  257 TO Y
     END-IF.
*
 TRANSLATE-TO-HEXA-LINE.
*=======================*
```

```
     IF FILECO161(K, X) = CHARTABD(Y)
        MOVE HEXTAB(Y) TO FILEHO161(K, X)
        MOVE 257 TO Y
     END-IF.
*
 CHECK-HEXCHARS-LINE.
*====================*
     IF NOT ( PAG = PAG-MAX AND K = REMLINES AND R > REMCHAR2 )
        EVALUATE TRUE
         WHEN DTHEXAI11(K, R) = 'a' MOVE 'A' TO DTHEXAI11(K, R)
         WHEN DTHEXAI11(K, R) = 'b' MOVE 'B' TO DTHEXAI11(K, R)
         WHEN DTHEXAI11(K, R) = 'c' MOVE 'C' TO DTHEXAI11(K, R)
         WHEN DTHEXAI11(K, R) = 'd' MOVE 'D' TO DTHEXAI11(K, R)
         WHEN DTHEXAI11(K, R) = 'e' MOVE 'E' TO DTHEXAI11(K, R)
         WHEN DTHEXAI11(K, R) = 'f' MOVE 'F' TO DTHEXAI11(K, R)
        END-EVALUATE
        IF DTHEXAI11(K, R) < 'A' OR  > '9' OR
         ( DTHEXAI11(K, R) > 'F' AND < 'Ø' )
           MOVE -1 TO DTHEXAL(K)
           GO TO ERRO-HEXA-LINE
        END-IF
     END-IF.
*
 CHECK-HEXCHARS-STARTKEYH.
*=========================*
     EVALUATE TRUE
       WHEN STARTKHI11(R) = 'a' MOVE 'A' TO STARTKHI11(R)
       WHEN STARTKHI11(R) = 'b' MOVE 'B' TO STARTKHI11(R)
       WHEN STARTKHI11(R) = 'c' MOVE 'C' TO STARTKHI11(R)
       WHEN STARTKHI11(R) = 'd' MOVE 'D' TO STARTKHI11(R)
       WHEN STARTKHI11(R) = 'e' MOVE 'E' TO STARTKHI11(R)
       WHEN STARTKHI11(R) = 'f' MOVE 'F' TO STARTKHI11(R)
     END-EVALUATE
     IF STARTKHI11(R) NOT = SPACE AND NOT = LOW-VALUE
        IF STARTKHI11(R) < 'A' OR  > '9' OR
         ( STARTKHI11(R) > 'F' AND < 'Ø' )
           MOVE -1 TO STARTKHL
           GO TO ERRO-HEXA-STARTK
        END-IF
     END-IF.
*
 TRANSLATE-TO-CHAR-STARTKEYH.
*============================*
     IF STARTKHI1(R) = HEXTAB(Y)
        MOVE CHARTABF(Y) TO STARTKEY(R:1)
        MOVE CHARTABD(Y) TO STARTKCI(R:1)
        MOVE 257 TO Y
     END-IF.
*
 TRANSLATE-TO-HEXA-KEY.
```

```
*=====================*
     IF FILEKEY(X)(Y:1) = CHARTABF(Z)
        COMPUTE Y2 = Y * 2 - 1
        MOVE HEXTAB(Z)   TO KEYHEXI(X)(Y2:2)
        MOVE CHARTABD(Z) TO KEYCHAI(X)(Y:2)
        MOVE 257 TO Z
     END-IF.
*
 FILEIN-TO-SCREENOUT.
*====================*
     IF PAG = PAG-MAX AND X > REMBYTES
        MOVE 257 TO X
     ELSE
        IF FILEC(PAG, X) = CHARTABF(Y)
           MOVE CHARTABD(Y) TO FILECO(X)
           MOVE HEXTAB(Y)   TO FILEHO(X)
           MOVE 257 TO Y
        END-IF
     END-IF.
*
 LOAD-MAP1.
*==========*
     IF PAG = PAG-MAX AND K > REMLINES
        MOVE 'U' TO DTCHARF(K) DTHEXAF(K)
     ELSE
        MOVE  SPACE TO DTCHARF(K) DTHEXAF(K)
        MOVE  FILECO16(K)  TO  DTCHARI(K)
        MOVE  FILEHO16(K)  TO  DTHEXAI(K)
        ADD CO(K) PAGOFFSET GIVING LINEOFFSET
        MOVE  LINEOFFSET   TO  OFSETDI(K)
        MOVE  HO(K)        TO  OFSETHI(K)(3:2)
        MOVE  HP(PAG)      TO  OFSETHI(K)(2:1)
        MOVE  Ø            TO  OFSETHI(K)(1:1)
     END-IF.
*
 RECEIVE-MAPØ.
*=============*
     EXEC CICS HANDLE CONDITION MAPFAIL (CLEARØ)
     END-EXEC
     EXEC CICS HANDLE AID CLEAR (CLEARØ)
                         PF3   (CLEARØ)
                         PF15  (CLEARØ)
     END-EXEC
     MOVE LOW-VALUES TO VEDISØØI
     EXEC CICS RECEIVE MAP('VEDISØØ')
     END-EXEC
     MOVE SPACES TO ERROØI.
*
 RECEIVE-MAP1.
*=============*
```

```
     EXEC CICS IGNORE CONDITION MAPFAIL
     END-EXEC
     EXEC CICS HANDLE AID CLEAR (CLEAR1)
     END-EXEC
     MOVE LOW-VALUES TO VEDISØ1I
     EXEC CICS RECEIVE MAP('VEDISØ1')
     END-EXEC.
     MOVE SPACES TO ERRO1I.
*
 SEND-MAPØ.
*==========*
     EXEC CICS SEND MAP ('VEDISØØ')
                    ERASE
                    CURSOR
     END-EXEC.
*
 SEND-MAP1.
*==========*
     MOVE -1 TO DTCHARL(1)
     EXEC CICS SEND MAP ('VEDISØ1')
                    ERASE
                    CURSOR
     END-EXEC.
*
 SEND-MAP1-DATAONLY.
*==================*
     MOVE -1 TO DTCHARL(1)
     EXEC CICS SEND MAP ('VEDISØ1')
                    DATAONLY
                    FRSET
                    CURSOR
     END-EXEC.
*
 SEND-ALARMØ.
*============*
     EXEC CICS SEND MAP ('VEDISØØ')
                    DATAONLY
                    CURSOR
                    ALARM
                    FREEKB
     END-EXEC.
*
 SEND-ALARM1.
*============*
     EXEC CICS SEND MAP ('VEDISØ1')
                    DATAONLY
                    CURSOR
                    ALARM
                    FREEKB
     END-EXEC.
```

```
*
 ERRO-INPUTØ.
*============*
     MOVE 'PLEASE ENTER THE CURSOR FIELD' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-FILENOTFND.
*================*
     MOVE 'FILE NOT FOUND' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-NOTOPEN.
*=============*
     MOVE 'FILE NOT OPENED' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-NOTENABLED.
*================*
     MOVE 'FILE IS NOT ENABLED' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-NOTKSDS.
*=============*
     MOVE 'FILE IS NOT KSDS' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-NOT-FOUND.
*===============*
     MOVE 'KEY NOT FOUND' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-ILLOGIC.
*=============*
     MOVE 'FILE KEY CHANGED- CANNOT REWRITE' TO ERRO1I.
     MOVE SPACES TO ANSWERI
     MOVE 2      TO INDICA
     PERFORM SEND-ALARM1
     GO TO RETURN-TRANSID.
*
 ERRO-RECSIZE.
*=============*
     MOVE 'RECORD SIZE IS GREATER THAN 4Ø96' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
```

```
*
 ERRO-HEXA-STARTK.
*=================*
     MOVE 'INVALID HEXADECIMAL' TO ERROØI.
     PERFORM SEND-ALARMØ
     GO TO RETURN-TRANSID.
*
 ERRO-HEXA-LINE.
*===============*
     MOVE 'INVALID HEXADECIMAL' TO ERRO1I.
     PERFORM SEND-ALARM1
     GO TO RETURN-TRANSID.
*
 ERRO-PAG-MAX.
*=============*
     MOVE 'LAST PAGE ATTAINED' TO ERRO1I.
     PERFORM SEND-MAP1-DATAONLY
     GO TO RETURN-TRANSID.
*
 ERRO-PAG-MIN.
*=============*
     MOVE 'FIRST PAGE ATTAINED' TO ERRO1I.
     PERFORM SEND-MAP1-DATAONLY
     GO TO RETURN-TRANSID.
*
 MESSAGE-QUESTION.
*=================*
     MOVE 'WRITE TO FILE? ENTER YES OR NO ==> ' TO ERRO1I.
     MOVE 'A' TO ANSWERF
     MOVE -1  TO ANSWERL
     PERFORM SEND-ALARM1
     GO TO RETURN-TRANSID.
*
 SET-LOWERCASE.
*==============*
     EXEC CICS SET TERMINAL(EIBTRMID)
                   NOUCTRAN
     END-EXEC.
*
 SET-UPPERCASE.
*==============*
     EXEC CICS SET TERMINAL(EIBTRMID)
                   UCTRAN
     END-EXEC.
*
 RETURN-TRANSID.
*===============*
     EXEC CICS RETURN TRANSID  (TRANSID)
                      COMMAREA (COMMAREA)
                      LENGTH   (EIBCALEN)
```

```
           END-EXEC.
      *
       CLEAR1.
      *=======*
           MOVE  1  TO PAG INDICA
           MOVE  Ø  TO PAGOFFSET RECALTERED
           MOVE -1  TO SELCØL(1)
           MOVE 'Z' TO DDNAMEØF
           MOVE LOW-VALUES TO VEDISØ1I
           MOVE SPACES TO ANSWERI
           PERFORM SEND-MAPØ
           GO TO RETURN-TRANSID.
      *
       CLEARØ.
      *=======*
           PERFORM SET-UPPERCASE
           EXEC CICS SEND FROM(MSG-END)
                          ERASE
           END-EXEC
           EXEC CICS RETURN
           END-EXEC.
      *
       ABEND-PROGRAM.
      *==============*
           MOVE EIBRESP TO MSG-ABEND-EIBRESP
           EXEC CICS HANDLE ABEND NOHANDLE
           END-EXEC
           EXEC CICS SEND FROM(MSG-ABEND)
                          ERASE
           END-EXEC
           PERFORM SET-UPPERCASE
           EXEC CICS RETURN
           END-EXEC
           GOBACK.
```

VEDIS00 SOURCE CODE

```
MAPSET   DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=(FREEKB),               *
               LANG=COBOL,TIOAPFX=YES,EXTATT=MAPONLY
VEDISØØ  DFHMDI SIZE=(24,8Ø)
         DFHMDF POS=(Ø1,Ø1),LENGTH=Ø5,ATTRB=(ASKIP,PROT),             *
               COLOR=YELLOW,INITIAL='File:'
DDNAMEØ  DFHMDF POS=(Ø1,Ø7),LENGTH=Ø8,ATTRB=(UNPROT,FSET,IC),        *
               COLOR=TURQUOISE
         DFHMDF POS=(Ø1,16),LENGTH=Ø1,ATTRB=(ASKIP,PROT)
DSNAMEØ  DFHMDF POS=(Ø1,18),LENGTH=44,ATTRB=(ASKIP,PROT,FSET),        *
               COLOR=PINK
         DFHMDF POS=(Ø1,67),LENGTH=Ø6,ATTRB=(ASKIP,PROT),             *
               COLOR=YELLOW,INITIAL='Lrecl:'
```

```
RECSIZØ  DFHMDF POS=(Ø1,74),LENGTH=Ø4,ATTRB=(ASKIP,PROT,FSET)
         DFHMDF POS=(Ø2,Ø1),LENGTH=Ø5,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Attr:'
STATUS   DFHMDF POS=(Ø2,Ø7),LENGTH=3Ø,ATTRB=(ASKIP,PROT,FSET)
         DFHMDF POS=(Ø2,67),LENGTH=Ø7,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Keylen:'
KEYLENØ  DFHMDF POS=(Ø2,75),LENGTH=Ø3,ATTRB=(ASKIP,PROT,FSET)
         DFHMDF POS=(Ø3,Ø1),LENGTH=24,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Start browse key (char):'
STARTKC  DFHMDF POS=(Ø3,26),LENGTH=38,ATTRB=(UNPROT),                  *
               COLOR=RED
         DFHMDF POS=(Ø3,65),LENGTH=Ø9,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='  Offset:'
KEYPOSØ  DFHMDF POS=(Ø3,75),LENGTH=Ø3,ATTRB=(ASKIP,PROT,FSET)
         DFHMDF POS=(Ø4,18),LENGTH=Ø7,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='(hexa):'
STARTKH  DFHMDF POS=(Ø4,26),LENGTH=5Ø,ATTRB=(UNPROT),                  *
               COLOR=RED
         DFHMDF POS=(Ø4,77),LENGTH=Ø1,ATTRB=(ASKIP,PROT)
         DFHMDF POS=(Ø5,Ø1),LENGTH=77,ATTRB=(ASKIP,PROT),              *
               COLOR=RED,INITIAL='-----------------------------------*
               ---------------------------------------'
SELECØ1  DFHMDF POS=(Ø6,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ1  DFHMDF POS=(Ø6,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ1  DFHMDF POS=(Ø6,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ2  DFHMDF POS=(Ø7,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ2  DFHMDF POS=(Ø7,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ2  DFHMDF POS=(Ø7,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ3  DFHMDF POS=(Ø8,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ3  DFHMDF POS=(Ø8,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ3  DFHMDF POS=(Ø8,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ4  DFHMDF POS=(Ø9,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ4  DFHMDF POS=(Ø9,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ4  DFHMDF POS=(Ø9,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ5  DFHMDF POS=(1Ø,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ5  DFHMDF POS=(1Ø,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ5  DFHMDF POS=(1Ø,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ6  DFHMDF POS=(11,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ6  DFHMDF POS=(11,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ6  DFHMDF POS=(11,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ7  DFHMDF POS=(12,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ7  DFHMDF POS=(12,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ7  DFHMDF POS=(12,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
```

```
SELECØ8  DFHMDF POS=(13,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ8  DFHMDF POS=(13,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ8  DFHMDF POS=(13,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELECØ9  DFHMDF POS=(14,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHAØ9  DFHMDF POS=(14,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEXØ9  DFHMDF POS=(14,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC1Ø  DFHMDF POS=(15,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA1Ø  DFHMDF POS=(15,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX1Ø  DFHMDF POS=(15,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC11  DFHMDF POS=(16,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA11  DFHMDF POS=(16,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX11  DFHMDF POS=(16,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC12  DFHMDF POS=(17,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA12  DFHMDF POS=(17,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX12  DFHMDF POS=(17,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC13  DFHMDF POS=(18,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA13  DFHMDF POS=(18,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX13  DFHMDF POS=(18,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC14  DFHMDF POS=(19,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA14  DFHMDF POS=(19,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX14  DFHMDF POS=(19,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC15  DFHMDF POS=(2Ø,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA15  DFHMDF POS=(2Ø,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX15  DFHMDF POS=(2Ø,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC16  DFHMDF POS=(21,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA16  DFHMDF POS=(21,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX16  DFHMDF POS=(21,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
SELEC17  DFHMDF POS=(22,Ø1),LENGTH=Ø1,ATTRB=(ASKIP,PROT,FSET)
KYCHA17  DFHMDF POS=(22,Ø3),LENGTH=24,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=TURQUOISE
KYHEX17  DFHMDF POS=(22,3Ø),LENGTH=48,ATTRB=(ASKIP,PROT,FSET)
ERROØ    DFHMDF POS=(23,2Ø),LENGTH=5Ø,ATTRB=(ASKIP,PROT,BRT,FSET),     *
               COLOR=YELLOW
         DFHMDF POS=(24,Ø8),LENGTH=61,ATTRB=(ASKIP,PROT,BRT),          *
               COLOR=NEUTRAL,INITIAL='ENTER Display record       PF8/2Ø*
               Next page    PF3/15 Exit'
         DFHMSD TYPE=FINAL
         END
```

```
MAPSET   DFHMSD TYPE=&SYSPARM,MODE=INOUT,CTRL=(FREEKB),                *
               LANG=COBOL,TIOAPFX=YES,EXTATT=MAPONLY
VEDISØ1  DFHMDI SIZE=(24,8Ø)
         DFHMDF POS=(Ø1,Ø3),LENGTH=1Ø,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='File.....:'
DDNAME1  DFHMDF POS=(Ø1,14),LENGTH=Ø8,ATTRB=(ASKIP,PROT),              *
               COLOR=TURQUOISE
DSNAME1  DFHMDF POS=(Ø1,23),LENGTH=41,ATTRB=(ASKIP,PROT),              *
               COLOR=TURQUOISE
         DFHMDF POS=(Ø1,65),LENGTH=Ø6,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Lrecl:'
RECSIZ1  DFHMDF POS=(Ø1,72),LENGTH=Ø4,ATTRB=(ASKIP,PROT)
         DFHMDF POS=(Ø2,Ø3),LENGTH=1Ø,ATTRB=(ASKIP,PROT,FSET),         *
               COLOR=YELLOW,INITIAL='Rec Key..:'
REKEYD   DFHMDF POS=(Ø2,14),LENGTH=48,ATTRB=(ASKIP,PROT),COLOR=PINK
         DFHMDF POS=(Ø2,65),LENGTH=Ø7,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Keylen:'
KEYLEN1  DFHMDF POS=(Ø2,73),LENGTH=Ø3,ATTRB=(ASKIP,PROT)
REKEYH   DFHMDF POS=(Ø3,14),LENGTH=48,ATTRB=(ASKIP,PROT),COLOR=PINK
         DFHMDF POS=(Ø3,65),LENGTH=Ø7,ATTRB=(ASKIP,PROT),              *
               COLOR=YELLOW,INITIAL='Offset:'
KEYPOS1  DFHMDF POS=(Ø3,73),LENGTH=Ø3,ATTRB=(ASKIP,PROT)
         DFHMDF POS=(Ø4,Ø3),LENGTH=75,ATTRB=(ASKIP,PROT),COLOR=YELLOW, *
               INITIAL='-------------------------------------------*
               ------------------------'
         DFHMDF POS=(Ø5,Ø3),LENGTH=77,ATTRB=(ASKIP,PROT),COLOR=YELLOW, *
               INITIAL='Dec        Ø....+....1....+    Ø . . . . + . . *
               . . 1 . . . . +       Hex'
OFSETD1  DFHMDF POS=(Ø6,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR1  DFHMDF POS=(Ø6,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(Ø6,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA1  DFHMDF POS=(Ø6,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(Ø6,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH1  DFHMDF POS=(Ø6,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD2  DFHMDF POS=(Ø7,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR2  DFHMDF POS=(Ø7,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(Ø7,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA2  DFHMDF POS=(Ø7,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(Ø7,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH2  DFHMDF POS=(Ø7,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD3  DFHMDF POS=(Ø8,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR3  DFHMDF POS=(Ø8,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(Ø8,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA3  DFHMDF POS=(Ø8,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(Ø8,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH3  DFHMDF POS=(Ø8,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD4  DFHMDF POS=(Ø9,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR4  DFHMDF POS=(Ø9,14),LENGTH=16,ATTRB=(UNPROT)
```

```
         DFHMDF POS=(Ø9,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA4  DFHMDF POS=(Ø9,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(Ø9,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH4  DFHMDF POS=(Ø9,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD5  DFHMDF POS=(1Ø,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR5  DFHMDF POS=(1Ø,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(1Ø,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA5  DFHMDF POS=(1Ø,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(1Ø,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH5  DFHMDF POS=(1Ø,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD6  DFHMDF POS=(11,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR6  DFHMDF POS=(11,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(11,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA6  DFHMDF POS=(11,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(11,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH6  DFHMDF POS=(11,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD7  DFHMDF POS=(12,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR7  DFHMDF POS=(12,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(12,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA7  DFHMDF POS=(12,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(12,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH7  DFHMDF POS=(12,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD8  DFHMDF POS=(13,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR8  DFHMDF POS=(13,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(13,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA8  DFHMDF POS=(13,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(13,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH8  DFHMDF POS=(13,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETD9  DFHMDF POS=(14,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHAR9  DFHMDF POS=(14,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(14,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXA9  DFHMDF POS=(14,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(14,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETH9  DFHMDF POS=(14,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDA  DFHMDF POS=(15,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARA  DFHMDF POS=(15,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(15,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAA  DFHMDF POS=(15,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(15,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHA  DFHMDF POS=(15,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDB  DFHMDF POS=(16,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARB  DFHMDF POS=(16,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(16,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAB  DFHMDF POS=(16,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(16,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHB  DFHMDF POS=(16,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDC  DFHMDF POS=(17,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARC  DFHMDF POS=(17,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(17,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAC  DFHMDF POS=(17,34),LENGTH=32,ATTRB=(UNPROT)
```

```
         DFHMDF POS=(17,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHC  DFHMDF POS=(17,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDD  DFHMDF POS=(18,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARD  DFHMDF POS=(18,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(18,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAD  DFHMDF POS=(18,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(18,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHD  DFHMDF POS=(18,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDE  DFHMDF POS=(19,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARE  DFHMDF POS=(19,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(19,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAE  DFHMDF POS=(19,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(19,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHE  DFHMDF POS=(19,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDF  DFHMDF POS=(2Ø,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARF  DFHMDF POS=(2Ø,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(2Ø,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAF  DFHMDF POS=(2Ø,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(2Ø,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHF  DFHMDF POS=(2Ø,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
OFSETDG  DFHMDF POS=(21,Ø3),LENGTH=5,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
DTCHARG  DFHMDF POS=(21,14),LENGTH=16,ATTRB=(UNPROT)
         DFHMDF POS=(21,31),LENGTH=1,ATTRB=(ASKIP,PROT)
DTHEXAG  DFHMDF POS=(21,34),LENGTH=32,ATTRB=(UNPROT)
         DFHMDF POS=(21,67),LENGTH=1,ATTRB=(ASKIP,PROT)
OFSETHG  DFHMDF POS=(21,72),LENGTH=4,ATTRB=(ASKIP,PROT),COLOR=TURQUOISE
         DFHMDF POS=(22,Ø3),LENGTH=77,ATTRB=(ASKIP,PROT),COLOR=YELLOW, *
               INITIAL='-----------------------------------------*
               -------------------------'
ERRO1    DFHMDF POS=(23,15),LENGTH=35,ATTRB=(ASKIP,PROT),COLOR=NEUTRAL
ANSWER   DFHMDF POS=(23,51),LENGTH=Ø3,ATTRB=(ASKIP,PROT),COLOR=YELLOW
         DFHMDF POS=(23,55),LENGTH=Ø1,ATTRB=(ASKIP,PROT)
         DFHMDF POS=(24,Ø3),LENGTH=75,ATTRB=(ASKIP,PROT),COLOR=NEUTRAL,*
               INITIAL='F3-Save/return    CLEAR-Return w/o save   F8-Ne*
               xt   F7-Prev   ENTER-check'
         DFHMSD TYPE=FINAL
         END
```


Monitoring CICS resources online

We needed to be able to check on the status of various resources in our CICS systems and send a message to the appropriate people when they were not available (depending on specified standards of time and day of the week).

The types of resource and the characteristics that we monitor are:

- Transactions – not active for a given interval
- Connections – acquired
- Terminals – acquired
- Files – open or closed
- Internet servers connected to CICS – activity
- MQSeries message queues – upper threshold limit

The program that does this checking is run every minute, thus providing a minimum interval of one minute between resource checks. As we will see later, the maximum interval is 99 minutes.

The parameters for checking each resource are contained in an extra-partition transient data queue that is read by the program at start-up. Each resource is defined to the program using a minimum of two and a maximum of nine records.

STRUCTURE OF THE TD QUEUE

Each resource to be monitored is defined in the Transient Data queue RESCHEK. The definitions consist of at least two records, namely the NA record and the DA record. Either ‘NA’ or ‘DA’ in positions 1-2 identifies the records.

The NA record contains information about the resource, such as resource name and type.

The DA record contains interval information such as day of the week, from and to time, and frequency of checking. If several days have the same time characteristics, they can be combined into one record.

The NA record

The NA record contains NA in the first two positions.

The resource name is in positions 4-11, and the type of resource

is in positions 13-16.

All fields are left justified and blank filled.

The DA record

The DA record contains DA in the first 2 positions.

The day(s) of the week, numbered 1-8, start in position 4.

Day 1 = Sunday, through 7 = Saturday; 8 = Holiday.

Days with the same intervals may be combined into a single DA record.

Each record can have up to four time intervals (from HH, to HH, Frequency)

Frequency can range from 01 – 99 and is in minutes. A frequency of 00 means do not check.

A SAMPLE QUEUE

This file can be created and/or updated using any convenient TSO editor. Here is an example:

```
**********************************************************************
* RESOURCES TO BE CHECKED FOR IDLE TIMES BY PROG S5ØPRESC             *
* RECORDS ARE IN THE FORM                                            *
* REC-TYPE(2) NA OR DA                                               *
* REOURCE-NAME(8) RESOURCE TYPE(4)   THIS IS FOR NA TYPE RECORDS     *
* FOR DA RECORDS:                                                    *
* DAY-NUMBER(1-7) CAN BE UP TO 7  FOR DAYS WITH SAME TIME PARAMETERS *
* DAY-NUMBER CAN BE .HOLIDAY. FOR HOLIDAY  PARAMETERS                *
* FROM-HR(2) TO-HR(2) FREQUENCY IN MINUTES(2)  THESE CAN BE UP TO 4  *
* GROUPS PER RECORD.                                                 *
**********************************************************************
NA I9Ø4     TX
DA 123456   ØØ 17 15 18 24 ØØ
DA 78       ØØ 24 ØØ
NA CICX     CONN
DA 123456   Ø8 17 Ø5 18 24 15
DA 78       ØØ 24 3Ø
NA TAKALOT  MQ   ØØØ1
DA 135      Ø8 1Ø Ø5 11 12 Ø2 13 24 ØØ
DA 24       Ø8 1Ø Ø2 11 12 1Ø 13 24 ØØ
NA DFHCSD   FILE
```

```
DA 123456   Ø8 17 Ø1 18 24 15
DA 78       ØØ 24 3Ø
NA I9IH     TRAN
DA 123456   Ø8 17 Ø1 18 24 15
DA 78       ØØ 24 3Ø
NA UBQ4     PU
DA 123456   Ø8 17 15 18 24 6Ø
DA 78       ØØ 24 3Ø
```

PROGRAM CHARACTERISTICS

The program, S50PRESC, is started either from CECI (Start trans(RESC)) or from a PLTSI program issuing the START command. On initiation, the program determines whether it is running for the first time. If so, it reads the TD queue and formats a resource table for use in monitoring. The actual checking of each resource is done by the appropriate CICS command such an Inquire, or Collect Statistics.

A special case exists for MQSeries queues since the resource table is limited to eight-character resource names. We use a table look-up product to match the actual queue name with the eight-character abbreviation. If your installation does not have the product, a table can easily be built.

When a resource is in the desired state, the program does nothing and checks the next resource.

If, however, a resource in not in the desired state, a message is sent via MQSeries to Vantive and written to a TD queue for systems programmers' records. At the next interval for that resource, another message is written stating either that the resource is now in the desired state or still inactive.

The program should be stopped and restarted every 24 hours. This will allow for changes in the input file to be incorporated as well as releasing storage below the line that accumulate, with each delay command. We have a separate program that runs at midnight to restart the monitoring program as well as clean up any of the remnants that programmers and others may have inadvertently left in the system.

This program was originally written to run under CICS 4.1. However, we have been running it under Transaction Server 1.3 for over a year. Our operating system is OS390 R2.8.

The program does require MQSeries facilities for messaging, but this can be changed easily.

Compilation is normal in all respects and does not require an authorized library. Remember, however, to include CSQSTUB in the link-edit. The load module can be in any library defined in DFHRPL.

CSD entries for the program and transaction can use system defaults.

```
 IDENTIFICATION      DIVISION.
 PROGRAM-ID. S50PRESC.
 AUTHOR.  SHALOM WASSERMAN.
************************************************************
*   THIS PROGRAM WILL CHECK ON RESOURCE AVAILABILITY       *
*   ACCORDING TO THE VALUES IN TD QUEUE RESC               *
************************************************************
 ENVIRONMENT DIVISION.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01 ACCUM-FIELDS.
    03  NUM-ACTIVE                   PIC S9(4) COMP VALUE +0.
    03  H                            PIC S9(4) COMP VALUE +1.
    03  I                            PIC S9(4) COMP VALUE +1.
    03  J                            PIC S9(4) COMP VALUE +0.
    03  K                            PIC S9(4) COMP VALUE +0.
    03  Q                            PIC S9(4) COMP VALUE +0.
    03  X                            PIC S9(4) COMP VALUE +0.
    03  DELAY                        PIC 9(4)  VALUE 1.
    03  CURR-OBS                     PIC S9(8) COMP VALUE +0.
    03  TD-BUFFLEN                   PIC S9(4) BINARY.
    03 DAY-TIME                    PIC S9(9) BINARY VALUE +1.
    03 EVENING-TIME                PIC S9(9) BINARY VALUE +30.
    03 NIGHT-TIME                  PIC S9(9) BINARY VALUE +60.
    03 SYS-ID                      PIC X(4).
    03  FIRST-TIME-FLAG              PIC 9 VALUE 1.
     88  FIRST-TIME                 VALUE 1.
     88  NOT-FIRST-TIME             VALUE ZERO.
    03 LONG-RUNNING-FLAG           PIC 9 VALUE ZERO.
 01 W-RESP                 PIC S9(8) COMP.
 01 WORK-FIELDS.
    03 NUM-TRAN-X                 PIC X(4).
```

```
   03 NUM-TRAN-N REDEFINES NUM-TRAN-X PIC S9(7) COMP.
   03 TIMEAREA                    PIC S9(15) COMP-3.
    03 SHOWTIME.
     04 COMPTIME.
       05 SHOW-HR                    PIC XX.
       05 FILLER                     PIC X.
       05 SHOW-MIN                   PIC XX.
       05 FILLER                     PIC X.
       05 SHOW-SEC                   PIC XX.
    03 SHOWDATE.
       05  FILLER                   PIC XX.
       05  YY                       PIC XX.
       05  MM                       PIC XX.
       05  DD                       PIC XX.
    03 INITDATE                     PIC X(8).
    03 DAY-NUM                      PIC S9(8) COMP.
    03 STAT-1                       PIC S9(8) COMP.
    03 VALID-INTERVAL               PIC 999 COMP-3.
    03 DELAY-INTERVAL               PIC 999 COMP-3 VALUE 1.
    03 OLIBDAT-DATE                 PIC 9(6).
    03 TEST-FREQ                   PIC 99.
01  RESOURCE-RECORD.
      03  REC-TYPE                    PIC XX.
      03  FILLER                      PIC X.
    02  INPUT-DATA.
      03  REC-TITLE                   PIC X(9).
      03  RES-DAY-NUM REDEFINES REC-TITLE PIC 9
          OCCURS 9.
      03  FILLER                      PIC X(68).
01  FILLER                           PIC X(45)
    VALUE 'RESOURCE TABLE STARTS HERE'.
01  FILLER.
    02  RESOURCE-TABLE       OCCURS  100.
      03   RESOURCE-CHECKS.
       05  RES-NAME                          PIC X(8).
       05  FILLER           REDEFINES RES-NAME.
         07  RES-NAME-STAR           PIC X.
         07  FILLER                  PIC X(7).
       05  FILLER           REDEFINES RES-NAME.
         07  RES-NAME-4              PIC X(4).
         07  FILLER                  PIC X(4).
       05  FILLER           REDEFINES RES-NAME.
         07  RES-NAME-2              PIC X(2).
         07  FILLER                  PIC X(6).
       05  FILLER                          PIC X.
       05  RES-TYPE                        PIC X(4).
       05  FILLER                          PIC X.
       05  MQ-LIMIT                        PIC X(5).
       05  FILLER                          PIC X.
```

```
      03   RESOURCE-INFO.
       05  MQ-TRAN                        PIC X(4).
       05  LAST-OBS                       PIC S9(8) COMP.
       05  FILLER                         PIC X.
       05  TOTAL-ELAPSED                  PIC 9(7) COMP-3.
       05  FILLER                         PIC X.
       05  SINCE-CHECKED                  PIC 9(3) COMP-3.
       05  STATUS-FLAG                    PIC X.
       05  FILLER                         PIC X.
      03   RESOURCE-TIMES OCCURS 8 TIMES.
       05  DAY-NAME                       PIC X(9).
       05  FREQUENCIES OCCURS 4 TIMES.
           07 FROM-HR                     PIC 99.
           07 FILLER                      PIC X.
           07 TO-HR                       PIC 99.
           07 FILLER                      PIC X.
           07 FREQ                        PIC 99.
           07 FILLER                      PIC X.
    02  TS-TABLE.
       05  TS-RES-NAME              PIC X(14) VALUE SPACES.
       05  TS-DESIRED                     PIC X(5).
       05  FILLER                         PIC X(4).
       05  TS-LAST-OBS                    PIC X(8).
       05  FILLER                         PIC X.
       05  TS-TOTAL-ELAPSED               PIC X(7).
       05  FILLER                         PIC X.
       05  TS-SINCE-CHECKED               PIC X(3).
       05  FILLER                         PIC X.
       05  TS-STATUS-FLAG                 PIC X.
       05  FILLER                         PIC X(19).
01  TABLE-HEADER.
    03 FILLER                             PIC X(80)
    VALUE 'RES-NAME TYPE LIMIT    OBSERVED ELAPSED CHK STAT'.
01  TERM-ID.
    03  PU-ID                              PIC XX.
    03  PU-NO                              PIC 99.
01  RESOURCE-TYPE-NAMES.
    03  CONN                            PIC X(8) VALUE 'CONN    '.
    03  TRAN                            PIC X(8) VALUE 'TRAN    '.
    03  TX                              PIC X(8) VALUE 'TX      '.
    03  PU                              PIC X(8) VALUE 'PU      '.
    03  MQS                             PIC X(8) VALUE 'MQ      '.
    03  DDNAM                           PIC X(8) VALUE 'FILE    '.
01  SYSTEM-NAMES.
    03 TEST-SYS                    PIC X(4) VALUE 'CICT'.
    03 PROD-SYS                    PIC X(4) VALUE 'CICP'.
    03 QA-SYS                      PIC X(4) VALUE 'CICV'.
    03 SYST-SYS                    PIC X(4) VALUE 'CICU'.
    03 TS13-SYS                    PIC X(4) VALUE 'TS13'.
```

```
      03 Z-SYS                    PIC X(4) VALUE 'CICZ'.
   01  STATUS-NAMES.
      03 STATUS-OPEN              PIC X(1)  VALUE 'O'.
      03 STATUS-STILL-OPEN        PIC X(1)  VALUE 'S'.
      03 STATUS-CLOSED            PIC X(1)  VALUE 'C'.
   01 FREQUENCY-NAMES.
      03 B-DAY                    PIC 9 VALUE 1.
      03 B-EVE                    PIC 9 VALUE 2.
      03 B-NIGHT                  PIC 9 VALUE 3.
      03 NB-DAY                   PIC 9 VALUE 4.
      03 NB-EVE                   PIC 9 VALUE 5.
   01  DAY-NAMES.
      03  SUNDAY    PIC X(9)  VALUE 'SUNDAY   '.
      03  MONDAY    PIC X(9)  VALUE 'MONDAY   '.
      03  TUESDAY   PIC X(9)  VALUE 'TUESDAY  '.
      03  WEDNESDAY PIC X(9)  VALUE 'WEDNESDAY'.
      03  THURSDAY  PIC X(9)  VALUE 'THURSDAY '.
      03  FRIDAY    PIC X(9)  VALUE 'FRIDAY   '.
      03  SATURDAY  PIC X(9)  VALUE 'SATURDAY '.
      03  HOLIDAY   PIC X(9)  VALUE 'HOLIDAY  '.
      03  NA        PIC XX    VALUE 'NA'.
      03  DA        PIC XX    VALUE 'DA'.
 ************************************************************
 *   MQM API FIELDS                                         *
 ************************************************************
   01  SAVEQ                        PIC X(48).
   01  W03-HCONN                    PIC S9(9) BINARY VALUE ZERO.
   01  W03-OPTIONS                  PIC S9(9) BINARY.
**   OBJECT DESCRIPTOR
**   OBJECT HANDLE
   01  W03-HOBJ           PIC S9(9) BINARY.
   01  FILLER.
      03  W03-COMPCODE-X            PIC X(4).
      03  W03-COMPCODE REDEFINES W03-COMPCODE-X PIC S9(9) BINARY.
      03  W03-REASON-X              PIC X(4).
      03  W03-REASON REDEFINES W03-REASON-X PIC S9(9) BINARY.
   01  W03-BUFFLEN                  PIC S9(9) BINARY.
   01  W03-PUT-BUFFER.
      03  W03-VANTIVE.
          05  VANTIVE-MSG.
             10  MQ-DATE            PIC X(8) VALUE SPACES.
             10  FILLER             PIC X    VALUE SPACES.
             10  MQ-TIME            PIC X(8) VALUE SPACES.
             10  FILLER             PIC X VALUE SPACES.
             10  MQ-SYSID           PIC X(4).
             10  FILLER             PIC X(14)
             VALUE '    S50PRESC '.
             10  MQ-RES             PIC X(4).
             10  FILLER             PIC X    VALUE SPACES.
             10  MQ-EVENT           PIC X(5).
```

```
           10  FILLER             PIC X    VALUE SPACES.
           10  MQ-RES-NAME        PIC X(8).
           10  FILLER             PIC X    VALUE SPACES.
           10  MQ-MSG             PIC X(14).
           10  MQ-MQS-ONLY.
               15  MQ-DEPTH-CONST PIC X(10) VALUE SPACES.
               15  MQ-DEPTH       PIC Z(6)99.
************************************************************
*   API CONTROL BLOCKS                                     *
************************************************************
 01  FILLER.
     02  MQM-OBJECT-DESCRIPTOR.
     COPY CMQODV.
     02  MQM-MESSAGE-DESCRIPTOR.
     COPY CMQMDV.
     02  MQM-PUT-MESSAGE-OPTIONS.
     COPY CMQPMOV.
     02  MQM-MQV.
     COPY CMQV.
 01  FILLER       PIC X(17) VALUE 'END OF'.
************************************************************
* TD      STANDARD MESSAGE HEADER                          *
************************************************************
 01  LOGHEADR.
     03  LOGTIME                 PIC X(8).
     03  FILLER                  PIC X  VALUE SPACES.
     03  FILLER                  PIC X(8) VALUE 'S50PRESC'.
     03  FILLER                  PIC X  VALUE SPACES.
     03  LOGTERM                 PIC X(4) VALUE SPACES.
     03  FILLER                  PIC X  VALUE SPACES.
     03  LOGTRAN                 PIC X(4) VALUE 'RESC'.
     03  FILLER                  PIC X  VALUE SPACES.
     03  LOGTEXT                 PIC X(52) VALUE SPACES.
 01  TEXT1.
     03  FILLER.
         05  FILLER              PIC X(38)
         VALUE ' PROGRAM HAS BEEN INITIALIZED    DATE='.
         05  TEXTDATE            PIC X(8) VALUE SPACES.
         05  FILLER              PIC X(16) VALUE SPACES.
 01  TEXT2.
     03  FILLER.
         05  FILLER.
            10  FILLER           PIC X    VALUE SPACES.
            10  LOG-TYPE         PIC X(5) VALUE SPACES.
            10  FILLER           PIC X    VALUE SPACES.
            10  LOG-NAME         PIC X(8).
            10  FILLER           PIC X    VALUE SPACES.
            10  LOG-STAT         PIC X(23) VALUE SPACES.
            10  FILLER           PIC X    VALUE SPACES.
            10  LOG-TIME          PIC Z999.
```

```
           10  FILLER             PIC X(8) VALUE ' MINUTES'.
           10  FILLER             PIC X(2) VALUE SPACES.
01  FILLER.
    03 ALT-STAT                   PIC X(23)
    VALUE ' HAS BEEN INACTIVE FOR '.
01  TEXT3.
    03  FILLER.
        05 FILLER                 PIC X(24)
        VALUE 'MQPUT1 NOT OK COMPCODE= '.
        05 TEXT3-COMP             PIC X(4).
        05 FILLER                 PIC X(10)
        VALUE ' REASON = '.
        05 TEXT3-REASON           PIC X(4).
01  TEXT4.
    03  FILLER.
        05 FILLER                 PIC X(24)
        VALUE 'MQOPEN NOT OK COMPCODE= '.
        05 TEXT4-COMP             PIC ZZZZ9.
        05 FILLER                 PIC X(10)
        VALUE ' REASON = '.
        05 TEXT4-REASON           PIC ZZZ9.
01  TEXT4A.
    03  FILLER.
        05  FILLER                PIC X(8)
        VALUE 'QUEUE = '.
        05  TEXT4A-Q-NAME         PIC X(44).
        05  FILLER                PIC X VALUE SPACES.
01  TEXT5.
    03  FILLER.
        05 FILLER                 PIC X(24)
        VALUE 'MQINQ  NOT OK COMPCODE= '.
        05 TEXT5-COMP             PIC ZZZZ9.
        05 FILLER                 PIC X(10)
        VALUE ' REASON = '.
        05 TEXT5-REASON           PIC ZZZ9.
01  TEXT6.
    03  FILLER.
        05 FILLER                 PIC X(25)
        VALUE 'MQCLOSE NOT OK COMPCODE= '.
        05 TEXT6-COMP             PIC ZZZZ9.
        05 FILLER                 PIC X(10)
        VALUE ' REASON = '.
        05 TEXT6-REASON           PIC ZZZ9.
    COPY A05C002.
01  TRAN-ID                PIC X(4).
01  TASK-LIST-SIZE         PIC S9(8) COMP VALUE ZERO.
01  TASK-NUMB-POINTER      USAGE IS POINTER VALUE NULL.
01  TASK-LIST-POINTERS.
    05  TASK-LIST-POINTER     USAGE IS POINTER VALUE NULL.
    05  TASK-LIST-POINTER-REDEF
```

```
            REDEFINES
            TASK-LIST-POINTER PIC S9(8) COMP.
 01  WS-MQ-DEPTH-X            PIC X(6) JUST RIGHT.
 01  WS-MQ-DEPTH REDEFINES WS-MQ-DEPTH-X  PIC Z(5)9.
 01  WS-MQ-DEPT9 REDEFINES WS-MQ-DEPTH-X  PIC 9(6).
 01 SELECTORCOUNT             PIC S9(9) BINARY VALUE 3.
 01 SELECTORS-TABLE.
    05 SELECTORS              PIC S9(9) BINARY OCCURS 3 TIMES.
 01 INTATTRCOUNT              PIC S9(9) BINARY VALUE 2.
 01 INTATTRS-TABLE.
    05 INTATTRS               PIC S9(9) BINARY OCCURS 2 TIMES.
 01 CHARATTLENGTH             PIC S9(9) BINARY VALUE 100.
 01 CHARATTRS                 PIC X(100)       VALUE LOW-VALUES.
 01  KEY2402-KEY.
        03 KEY2402-ZIHUY-MASHAV  PIC XX.
        03 KEY2402-SHEM-LOGI     PIC X(8).
        COPY   TAB2402P.
 LINKAGE SECTION.
       COPY DFHA06DS.
       COPY DFHXMRDS.
   01  TRAN-ID-STR.
     03  WORK-TRAN-ID           PIC X(4).
 PROCEDURE  DIVISION.
 P0000-PROG.
     PERFORM P1000-INIT THRU P1000-INIT-EXIT.
     MOVE 1 TO I.
     PERFORM P2000-PROC THRU P2000-PROC-EXIT
             VARYING I FROM 1 BY 1
             UNTIL I > NUM-ACTIVE.
        MOVE ZERO TO FIRST-TIME-FLAG.
     PERFORM P2500-DELAY THRU P2500-DELAY-EXIT.
     GO TO P0000-PROG.
 P1000-INIT.
     EXEC CICS ASKTIME ABSTIME(TIMEAREA)
     END-EXEC.
     EXEC CICS FORMATTIME ABSTIME(TIMEAREA) TIME(SHOWTIME)
           TIMESEP YYYYMMDD(SHOWDATE) DDMMYY(OLIBDAT-DATE)
           DAYOFWEEK(DAY-NUM)
     END-EXEC.
     EXEC CICS FORMATTIME ABSTIME(TIMEAREA) TIME(LOGTIME)
           TIMESEP(' ') DATESEP DDMMYY(TEXTDATE)
     END-EXEC.
***************************************************************
* THIS CALL IS TO A ROUTINE TO DETERMINE WHETHER THE DAY IS*
* AN ISRAELI HOLIDAY SINCE THEY DO NOT FOLLOW THE          *
* GREGORIAN CALENDAR.                                      *
***************************************************************
     MOVE OLIBDAT-DATE TO A02K-TARICH.
     MOVE '2' TO A02K-KOD-SHANA.
     MOVE 'C' TO A02K-SUG-PEULA.
```

```
     EXEC CICS LINK PROGRAM('OLIBDAT') COMMAREA(A002-NETUNIM)
                                       LENGTH(76) END-EXEC.
     ADD 1 TO DAY-NUM GIVING K.
     IF A02P-YOM-SHABATON
        MOVE 8 TO K.
************************************************************
* END OF ISRAELI HOLIDAY DETERMINATION ROUTINE             *
************************************************************
     IF FIRST-TIME-FLAG = 1 PERFORM P8000-INIT
      THRU P8000-INIT-EXIT.
 P1000-INIT-EXIT.
     EXIT.
 P2000-PROC.
     MOVE ZERO TO LONG-RUNNING-FLAG.
************************************************************
* DO NOT CHECK SWIFT ON SUNDAY                             *
************************************************************
     EVALUATE DAY-NUM
     WHEN 0
       IF RES-NAME-2(I) = 'IW' OR
          RES-NAME-2(I) = 'OW'
          GO TO P2000-PROC-EXIT
       END-IF
     END-EVALUATE.
     MOVE 1 TO J.
  P2100-FREQ.
        IF FROM-HR(I, K, J) > TO-HR(I, K, J)
           ADD 24 TO TO-HR(I, K, J).
        IF SHOW-HR >= FROM-HR(I,K,J) AND SHOW-HR <= TO-HR(I,K,J)
           MOVE FREQ(I,K,J) TO TEST-FREQ
           GO TO P2100-FREQ-EXIT
        ELSE
        ADD 1 TO J.
        IF J > 4 MOVE ZERO TO TEST-FREQ
        ELSE GO TO P2100-FREQ
        END-IF.
  P2100-FREQ-EXIT.
     EXIT.
  P2200-PROC-CONTINUE.
     IF TEST-FREQ = ZERO GO TO P2000-PROC-EXIT.
*    IF RES-NAME(I) = 'TIMESKED' GO TO P2000-PROC-EXIT.
     EXEC CICS HANDLE CONDITION NOTFND(P2000-PROC-EXIT)
         SYSIDERR(P2000-PROC-EXIT)
     END-EXEC.
     EVALUATE RES-TYPE(I)
      WHEN CONN PERFORM P2010-CONN THRU P2010-CONN-EXIT
      WHEN TRAN PERFORM P2010-TRAN THRU P2010-TRAN-EXIT
      WHEN TX   PERFORM P2010-TX   THRU P2010-TX-EXIT
      WHEN PU   PERFORM P2010-PU THRU P2010-PU-EXIT
      WHEN MQS  PERFORM P2010-MQS THRU P2010-MQS-EXIT
```

```
         WHEN DDNAM PERFORM P2010-FILE THRU P2010-FILE-EXIT
         WHEN OTHER GO TO P2000-PROC-EXIT
       END-EVALUATE.
       IF FIRST-TIME-FLAG = 1
          MOVE CURR-OBS TO LAST-OBS(I)
      *   MOVE ZERO TO FIRST-TIME-FLAG
          MOVE DELAY-INTERVAL TO SINCE-CHECKED(I)
          GO TO P2000-PROC-EXIT.
       MOVE TEST-FREQ TO VALID-INTERVAL
       IF VALID-INTERVAL > SINCE-CHECKED(I)
         ADD DELAY-INTERVAL TO SINCE-CHECKED(I)
         GO TO P2000-PROC-EXIT
       ELSE
         MOVE DELAY-INTERVAL TO SINCE-CHECKED(I)
       END-IF.
       EVALUATE RES-TYPE(I)
       WHEN CONN
          IF   CURR-OBS NOT EQUAL DFHVALUE(ACQUIRED)
               PERFORM P2100-BUILD-MSG THRU P2100-BUILD-MSG-EXIT
          ELSE IF CURR-OBS = DFHVALUE(ACQUIRED) AND
               STATUS-FLAG(I) NOT = STATUS-CLOSED
               PERFORM P2200-OK THRU P2200-OK-EXIT
               END-IF
          END-IF
       WHEN MQS
            IF CURR-OBS            >= WS-MQ-DEPT9 AND
               CURR-OBS >= LAST-OBS(I) AND
               LONG-RUNNING-FLAG = ZERO
             PERFORM P2100-BUILD-MSG THRU P2100-BUILD-MSG-EXIT
             MOVE CURR-OBS TO LAST-OBS(I)
            ELSE MOVE CURR-OBS TO LAST-OBS(I)
             MOVE ZERO TO TOTAL-ELAPSED(I) LONG-RUNNING-FLAG
             PERFORM P2200-OK THRU P2200-OK-EXIT
           END-IF
```

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

Shalom Wasserman
CICS Systems Programmer (Israel)

Switching from CICS to another VTAM application

EXEC CICS ISSUE PASS is probably a little-known but very useful command when you have a program built around it. Basically, this command allows us to abandon the current CICS and move to another VTAM application, whether it is another CICS, a TSO, or whatever. Optionally, if the target application is able to receive an argument, we can also send it. This is the case with a TSO destination, where we can send the userid we wish to log on to. In such a case, we go straight to the 'enter password' screen, instead of the logon prompt.

The ISSUE PASS command has the following syntax:

```
EXEC CICS ISSUE PASS LUNAME (vtam_application)
                     FROM   (optional_argument)
                     LENGTH (optional_argument_length)
```

FROM and LENGTH are optional. We need to specify them only if we want to send an argument (the userid) to the VTAM application.

The program presented here, VTAMPASS, assigned to transaction VTPA, is just a front-end to this command. Within the program, you can pre-define up to 12 VTAM applications. You can invoke the program in two ways.

You can call the transaction and immediately pass it the destination name and optionally also the userid, for example:

```
VTPA R23CICSL

VTPA F23TSO USERØ97

VTPA F23TSO *
```

In this example, the userid argument is just an asterisk. In this case, the program will replace the asterisk with the currently signed-on CICS userid. This is practical if your userid is the same in CICS and in the TSO destination, so you don't have to type it fully.

Invoking the transaction without arguments, you will get a screen

with a list of the VTAM applications declared in the program, one per line. To switch to one, all you need do is place the cursor in the same line as it and press *Enter*. To pass an argument, tab the cursor to the respective line, in front of the name, and type the argument. The same method of just typing an asterisk is also valid.

The program does not use a BMS to present the screen, just a 3270 datastream. In front of each VTAM application name, there is an eight-byte unprotected area to type the optional argument.The choice of the destination is taken from the cursor line. If the cursor is not on a valid line, the screen will be displayed again. If you wish to exit instead of switching, you can use PF3.

Before using the program, you should fill the DEST11 fields in the working-storage with pre-defined destinations of your choice. If you do not want to use all the twelve available, just set those unused to spaces.

VTAMPASS SOURCE CODE

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. VTAMPASS.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
       WORKING-STORAGE SECTION.
      *=============================================================*
       77  Z           PIC S9(4) COMP VALUE +Ø.
       77  Z1          PIC S9(4) COMP VALUE +Ø.
       77  P1          PIC S9(4) COMP VALUE +Ø.
       77  P2          PIC S9(4) COMP VALUE +Ø.
       77  LRECEIVE    PIC S9(4) COMP VALUE +Ø.
       77  LSCREENAREA PIC S9(4) COMP VALUE +Ø.
      *
       Ø1  SCREEN-VALUES.
           Ø2 DEST11.
              Ø4  FILLER  PIC X(8) VALUE  'AØ3TSO'.
              Ø4  FILLER  PIC X(8) VALUE  'B13TSO'.
              Ø4  FILLER  PIC X(8) VALUE  'C23TSO'.
              Ø4  FILLER  PIC X(8) VALUE  'PØ2TSO'.
              Ø4  FILLER  PIC X(8) VALUE  'RØ3CICSL'.
              Ø4  FILLER  PIC X(8) VALUE  'R13CICSU'.
              Ø4  FILLER  PIC X(8) VALUE  'R23CICSD'.
              Ø4  FILLER  PIC X(8) VALUE  'CØ2CICSA'.
              Ø4  FILLER  PIC X(8) VALUE  'C12CICS7'.
```

```
           Ø4  FILLER  PIC X(8) VALUE  'D35CICS9'.
           Ø4  FILLER  PIC X(8) VALUE   SPACES.
           Ø4  FILLER  PIC X(8) VALUE   SPACES.
        Ø2 DEST1 REDEFINES DEST11 PIC X(8) OCCURS 12.
*
        Ø2 ADDR11.
           Ø4  FILLER  PIC X(5) VALUE X'1143C11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1144D11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1145E11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1146F11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1148C11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1149D11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'114AE11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'114BF11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'114DC11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'114ED11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'114FE11DF8'.
           Ø4  FILLER  PIC X(5) VALUE X'1150F11DF8'.
        Ø2 ADDR1 REDEFINES ADDR11 PIC X(5) OCCURS 12.
*
        Ø2 ADDR22.
           Ø4  FILLER  PIC X(7) VALUE X'1D401143D31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401144E31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401145F31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401147C31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401148D31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401149E31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D40114AF31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D40114CC31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D40114DD31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D40114EE31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D40114FF31DFØ'.
           Ø4  FILLER  PIC X(7) VALUE X'1D401151C31DFØ'.
        Ø2 ADDR2 REDEFINES ADDR22 PIC X(7) OCCURS 12.
*
         Ø1  RECEIVE-AREA.
        Ø2  RECEIVE-SCREEN.
          Ø4  FILLER                          PIC X(4).
          Ø4  RECEIVESC                       PIC X(18).
          Ø4  RECEIVE-S  REDEFINES RECEIVESC PIC X OCCURS 18.
        Ø2  RECEIVE-327Ø REDEFINES RECEIVE-SCREEN.
          Ø4  FILLER                          PIC X(3).
          Ø4  RECEIVE-3                       PIC X(19).
        Ø2  PARM11                            PIC X(8).
        Ø2  PARM1        REDEFINES PARM11     PIC X OCCURS 8.
        Ø2  PARM22                            PIC X(8).
        Ø2  PARM2        REDEFINES PARM22     PIC X OCCURS 8.
*
         Ø1  SCREENAREA.
        Ø2  FILLER PIC X(11) VALUE X'114040290341F242F3C0FØ'.
```

```
     Ø2  FILLER PIC X(28) VALUE '* * Put the cursor in front '.
     Ø2  FILLER PIC X(28) VALUE 'of the desired destinatinati'.
     Ø2  FILLER PIC X(22) VALUE 'on and press Enter * *'.
     Ø2  FILLER PIC X(1Ø) VALUE X'29Ø24100C0F01143CB13'.
     Ø2  SCREEN-LINES PIC X(24Ø).
     Ø2  SCREEN-LINE REDEFINES SCREEN-LINES OCCURS 12.
       Ø4  A1   PIC X(5).
       Ø4  D1   PIC X(8).
       Ø4  A2   PIC X(7).
*
 LINKAGE SECTION.
 Ø1  DFHCOMMAREA  PIC X(1).
*================================================================*
 PROCEDURE DIVISION.
*================================================================*
*
 RECEIVE-DATA.
*=============*
     EXEC CICS HANDLE AID CLEAR (PROGRAM-EXIT1)
                         PF3   (PROGRAM-EXIT1)
                         PF15  (PROGRAM-EXIT1)
     END-EXEC
     EXEC CICS IGNORE CONDITION LENGERR
                                ENDDATA
     END-EXEC
     MOVE 22 TO LRECEIVE
     MOVE SPACES TO RECEIVE-AREA
     EXEC CICS RECEIVE INTO   (RECEIVE-SCREEN)
                      LENGTH (LRECEIVE)
     END-EXEC
     IF EIBCALEN = Ø
        GO TO FIRST-TIME
     ELSE
        GO TO OTHER-TIMES
     END-IF.
*
 FIRST-TIME.
*===========*
     IF LRECEIVE > 5
        PERFORM SEPARATE-PARAMETERS THRU
                SEPARATE-PARAMETERS-EXIT
                VARYING Z FROM 1 BY 1 UNTIL Z > 18
        GO TO VTAM-PASS
     ELSE
        MOVE 99 TO LSCREENAREA
        MOVE SPACES TO SCREEN-LINES
        PERFORM LOAD-SCREEN-LINES
                VARYING Z FROM 1 BY 1 UNTIL Z > 12
        GO TO SEND-RETURN
     END-IF.
```

```
*
 OTHER-TIMES.
*============*
     COMPUTE Z = EIBCPOSN / 8Ø - 1
     IF Z > Ø AND Z < 13
        MOVE DEST1(Z)  TO PARM11
        MOVE RECEIVE-3 TO PARM22
        GO TO VTAM-PASS
     ELSE
        GO TO SEND-RETURN
     END-IF.
*
 SEPARATE-PARAMETERS.
*====================*
     IF RECEIVE-S(Z) = SPACE OR = LOW-VALUE
        IF PARM11 = SPACES OR LOW-VALUES
           MOVE 1 TO Z1
        ELSE
           MOVE 2 TO Z1
        END-IF
        GO TO SEPARATE-PARAMETERS-EXIT
     END-IF
     IF Z1 = 1
        ADD 1 TO P1
        MOVE RECEIVE-S(Z) TO PARM1(P1)
     END-IF
     IF Z1 = 2
        ADD 1 TO P2
        MOVE RECEIVE-S(Z) TO PARM2(P2)
     END-IF.
*
 SEPARATE-PARAMETERS-EXIT.
*=========================*
     EXIT.
*
 LOAD-SCREEN-LINES.
*==================*
     IF DEST1(Z) NOT EQUAL SPACES
        MOVE  DEST1(Z) TO  D1(Z)
        MOVE  ADDR1(Z) TO  A1(Z)
        MOVE  ADDR2(Z) TO  A2(Z)
        ADD 2Ø TO LSCREENAREA
     END-IF.
*
 SEND-RETURN.
*============*
     EXEC CICS SEND FROM   (SCREENAREA)
                    LENGTH (LSCREENAREA)
     END-EXEC
     MOVE 1 TO EIBCALEN
```

```
     EXEC CICS RETURN TRANSID  (EIBTRNID)
                      COMMAREA (PARM11)
                      LENGTH   (EIBCALEN)
     END-EXEC.
*
 VTAM-PASS.
*==========*
     INSPECT PARM11 REPLACING ALL LOW-VALUES BY SPACES
     INSPECT PARM22 REPLACING ALL LOW-VALUES BY SPACES
     IF PARM11 = SPACES
        MOVE Ø TO LRECEIVE
        GO TO FIRST-TIME
     END-IF
     IF PARM2(1) = '*'
        EXEC CICS ASSIGN USERID (PARM22)
        END-EXEC
     END-IF
     IF PARM22 = SPACES
        EXEC CICS ISSUE PASS LUNAME (PARM11)
        END-EXEC
     ELSE
        EXEC CICS ISSUE PASS LUNAME (PARM11)
                             FROM   (PARM22)
                             LENGTH (8)
        END-EXEC
     END-IF
     GO TO PROGRAM-EXIT.
*
 PROGRAM-EXIT1.
*=============*
     EXEC CICS SEND CONTROL ERASE
     END-EXEC.
*
 PROGRAM-EXIT.
*=============*
     EXEC CICS RETURN
     END-EXEC.
     GOBACK.
```


ClientSoft is teaming up with Actional to create Web services management products and will integrate its ClientSoft Tanit Objects (CTO) with Actional's SOAPswitch and SOAPstation products to target mainframe sites.

The CTO 3.4 mainframe integration software for high-performance transactions currently supports XML and Web services by generating instant WSDL and XML files from CICS and IMS applications without reliance on traditional emulation-based host connectivity.

It can also access any version of a mainframe application through IBM VSE, MQSeries, CTG, EXCI, and Microsoft COMTI.

For further information contact:
ClientSoft, 8323 Northwest 12 Street, Suite 216, Miami, FL 33126, USA.
Tel: (305) 716 1007.
URL: http://www.clientsoft.com/news/pr1002.htm.

* * *

IBM has announced CICS Transaction Server for Windows V5.0, which runs on Windows 2000 and NT, and is designed to offer a migration path to these platforms for users of CICS TS for OS/2.

It is said to provide a comprehensive CICS API on a slim easy-to-install codebase.

IBM CICS Transaction Server (CICS TS) for Windows V5.0 delivers a version of CICS that is compatible with CICS Transaction Server for OS/2 Warp V4.1, providing those users with a migration path to Windows platforms.

CICS TS for Windows V5.0 incorporates as its file manager Pervasive.SQL 2000i Server (at level SP4). Pervasive.SQL 2000i Server contains the Btrieve file manager, as used in CICS TS for OS/2 V4.1.

Specific functions include the Integrated Performance Analyzer, programming language support for COBOL, C, C++, and PL/I, support for the master terminal (CEMT) function for management of CICS resources, and interfaces for external security manager, external file manager, and resource definition.

There's support for dynamic resource installation (CEDA), MRO, FEPI LU0, and National Language.

For further information contact your local IBM representative.
URL: http://www.ibm.com.

* * *

Pervasive Software has announced that its embedded database engine, Pervasive.SQL, will ship with every IBM CICS for Windows.

For further information contact your local IBM representative.
URL: http://www.pervasive.com.