217

CICS update

December 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 £100 ($160) per 1000 words and £50 ($80) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £20 ($32) per 100 lines. 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.

Cold start next time – Outstanding UOW support

The introduction of Outstanding UOW indicators via PTF (CTS 1.3, PTF: UQ72907 (PQ60109), and CTS 2.2, PTF: UQ72639 (PQ60089)), has prompted an enhancement to the warm check program published in *CICS Update,* issue 209, April 2003 (see article entitled 'DFHRMUTL enhancements – or can I cold restart CICS?').

To support this new PTF, an update to the warm check program (see *CICS Update,* issue 180, November 2000, 'Cold start next time – revisited') is required. The program has been modified to work with or without the PTF applied. If the new GCD record is found, the program will interrogate the UOW indicators and set a return code accordingly. If the record is not found, the program functions as before and sets a return code based solely on the warm keypoint flag.

```
/*****************************************************************
 * LTWRMCK  : Check CICS DFHGCD for Warm keypoint/UOW indicators      *
 * v1.1.Ø.  :                                                        *
 * ----------------------------------------------------------------- *
 * (C) Copyright 2ØØ3 Lemon Tree                                     *
 *****************************************************************
 * Hardware and Software Requirements                                *
 * ----------------------------------------------------------------- *
 * Language: C (LE - use of flocate)                                 *
 * CPU: All that run MVS/ESA and C                                   *
 * Operating System: MVS/ESA                                         *
 * CICS Version/Release: CICS Transaction Server for OS/39Ø v1.3.    *
 *                       CICS Transaction Server for OS/39Ø v2.2.    *
 *****************************************************************
 * Function                                                          *
 * ----------------------------------------------------------------- *
 * This program reads the CICS DFHGCD file and checks the Restart    *
 * keypoint record: '....DFHRMDM DFHRMDM_RESTART' (.... = x'ØØØØØØ11')*
 * The Outstanding OUW indicators are at offsets Ø,4 5,4 9,4 (beyound *
 * the key), The program checks these values are zero.               *
 * If PTFs UQ729Ø7 or UQ72639 are not APPLY'd then the above will fail*
 * (record not found), in which case this program will read the Anchor*
 * keypoint record: '....DFHRMDM DFHRMDM_ANCHOR ' (.... = x'ØØØØØØ11')*
 * The Warm keypoint indicator is then at offset 2Ø,1 (beyound the   *
 * key):                                                             *
 *                                                                   *
```

```
 * x'Ø3' - Warm keypoint taken                                          *
 * x'Ø4' - Emergency restart required                                   *
 *                                                                      *
 * The program sets return codes:                                       *
 *                                                                      *
 * 'ØØ' - Warm keypoint found, no outstanding UOW, OK to cold start     *
 * 'Ø4' - Emergency restart required                                    *
 * '12' - Can't Open DFHGCD                                             *
 *                                                                      *
 * Notes                                                                *
 ************************************************************************/
/*----------------------------------------------------------------------*
 * #includes                                                            *
 *----------------------------------------------------------------------*/
#include <signal.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
/*----------------------------------------------------------------------*
 * #defines and typedefs                                                *
 *----------------------------------------------------------------------*/
#define INDD "DD:DFHGCD"
#define DFHGCD_KEYLEN 28
#define RC_OK Ø
#define RC_WARN 4
#define RC_FAILURE 12
#define KEYPOINT_WARM Øx3
#define KEYPOINT_EMER Øx4
/*----------------------------------------------------------------------*
 * global variables                                                     *
 *----------------------------------------------------------------------*/
/*----------------------------------------------------------------------*
 * function prototypes                                                  *
 *----------------------------------------------------------------------*/
void error_exit(int);
/*----------------------------------------------------------------------*
 * main()                                                               *
 *----------------------------------------------------------------------*/
int main()                              /* ==> Entry point              */
{                                       /*                              */
/*----------------------------------------------------------------------*
 * local variables                                                      *
 *----------------------------------------------------------------------*/
FILE *dfhgcd;                           /* Input file (DFHGCD)          */
                                        /*------------------------------*/
struct dfhgcd_rec                       /* DFHGCD record layout         */
{                                       /*                              */
  unsigned int key_hex;                 /* Record key hex               */
  char key_char??(24??);                /* Record key char              */
  signed long int indoubt_uows;         /* Indoubt UOWs                 */
```

```
  signed long int cfail_uows;          /* Commit failed UOWs          */
  signed long int bfail_uows;          /* Backout failed UOWs         */
  char filler??(7??);                  /* ** Filler **                */
  char keypoint;                       /* Keypoint indicator          */
};                                     /*                             */
                                       /*-----------------------------*/
_Packed struct dfhgcd_rec              /* DFHGCD anchor record        */
 dfhgcd_rec_anch =                     /*                             */
{                                      /*                             */
  Øx11,                                /* Record key hex              */
  "DFHRMDM DFHRMDM_ANCHOR  ",          /* Record key char             */
  Ø,                                   /* ** Filler **                */
  Ø,                                   /* ** Filler **                */
  Ø,                                   /* ** Filler **                */
  "       ",                           /* ** Filler **                */
  Ø                                    /* Keypoint indicator          */
};                                     /*                             */
                                       /*-----------------------------*/
_Packed struct dfhgcd_rec              /* Pointer at anchor record    */
 *dfhgcd_rec_anch_ptr =                /*                             */
 &dfhgcd_rec_anch;                     /*                             */
                                       /*-----------------------------*/
_Packed struct dfhgcd_rec              /* DFHGCD anchor record        */
 dfhgcd_rec_restart =                  /*                             */
{                                      /*                             */
  Øx11,                                /* Record key hex              */
  "DFHRMDM DFHRMDM_RESTART ",          /* Record key char             */
  Ø,                                   /* Indoubt UOW                 */
  Ø,                                   /* Commit failed UOW           */
  Ø,                                   /* Backout failed UOW          */
  "       ",                           /* ** Filler **                */
  Ø                                    /* ** Filler **                */
};                                     /*                             */
                                       /*-----------------------------*/
_Packed struct dfhgcd_rec              /* Pointer at restart record   */
 *dfhgcd_rec_restart_ptr =             /*                             */
 &dfhgcd_rec_restart;                  /*                             */
int rc_warm;                           /* Return code                 */
/*-------------------------------------------------------------------*
 * code                                                              *
 *-------------------------------------------------------------------*/
rc_warm = RC_WARN;                     /* Set return code             */
if(dfhgcd = fopen(INDD, "rb, type=record"))
{                                      /* Open DFHGCD                 */
  if(!flocate(dfhgcd,                  /* Locate Restart record       */
      &dfhgcd_rec_restart.key_hex,     /*                             */
      DFHGCD_KEYLEN,                   /*                             */
      __KEY_EQ))                       /*                             */
  {                                    /*                             */
    fread(dfhgcd_rec_restart_ptr,      /* Read Restart record         */
```

```
      1,                                /*                              */
      sizeof(struct dfhgcd_rec),        /*                              */
      dfhgcd);                          /*                              */
                                        /* Check UOW flags              */
    if(dfhgcd_rec_restart.indoubt_uows == Ø &&
     dfhgcd_rec_restart.cfail_uows == Ø &&
     dfhgcd_rec_restart.bfail_uows == Ø)
    {                                   /* ...if all zero               */
      rc_warm = RC_OK;                  /* ...set return code           */
    }                                   /*                              */
  }                                     /*                              */
  else                                  /*                              */
  {                                     /*                              */
    if(!flocate(dfhgcd,                 /* Locate Anchor record         */
        &dfhgcd_rec_anch.key_hex,       /*                              */
        DFHGCD_KEYLEN,                  /*                              */
        __KEY_EQ))                      /*                              */
    {                                   /*                              */
      fread(dfhgcd_rec_anch_ptr,        /* Read Anchor record           */
        1,                              /*                              */
        sizeof(struct dfhgcd_rec),      /*                              */
        dfhgcd);                        /*                              */
      if(dfhgcd_rec_anch.keypoint ==    /* Check Keypoint flag          */
       KEYPOINT_WARM)                   /* ...if WARM then...           */
      {                                 /*                              */
        rc_warm = RC_OK;                /* ...set return code           */
      }                                 /*                              */
    }                                   /*                              */
  }                                     /*                              */
}                                       /*                              */
else                                    /* Unable to open file          */
{                                       /*                              */
  rc_warm = RC_FAILURE;                 /* Set return code              */
}                                       /*                              */
return rc_warm;                         /* <== Exit point               */
}                                       /*                              */
/*--------------------------------------------------------------------*
 * function: error_exit                                               *
 *--------------------------------------------------------------------*/
void error_exit(int sig_num)            /* ==> Function Entry           */
{                                       /*                              */
/*--------------------------------------------------------------------*
 * local variables                                                    *
 *--------------------------------------------------------------------*/
/*--------------------------------------------------------------------*
 * code                                                               *
 *--------------------------------------------------------------------*/
exit(EXIT_FAILURE);                     /* <== Exit point               */
}                                       /* <== Function Exit            */
/*--------------------------------------------------------------------*
```

```
 * End of Program                                                    *
 *-------------------------------------------------------------------*/
```


An aid to subroutine program debugging

If you've ever tried to debug a linkable program that is driven by COMMAREA input, you'll know it has issues. You can define a transaction to run the program and run CEDF, but you can't supply a COMMAREA. You can CECI link to the program and supply a COMMAREA, but you can't run CEDF!

This simple program helps get round the problem. You supply the program you wish to link to, create an input temporary storage queue, which can be used to store your input parameters, and a separate output TSQ to store the return values.

Define a transaction to run LINKER and define the program LINKER with CEDF(No) to stop its calls getting in the way of debugging the real program when using CEDF with LINKER.

LINKER PROGRAM

```
#pragma XOPTS(NOEDF)
#include "zlinker.h"
#include "string.h"
#include "stdio.h"
#include "stdlib.h"
#include "dfhaid.h"
#include "dfhbmsca.h"
char commarea??(32767??);
char *commarea_ptr = &commarea??(Ø??);
short commarea_length = Ø;
int  field_len??(9??);
int  num_errors = Ø;
void do_commarea(void);
void do_output_commarea(void);
void do_char(char*, char*);
void display_tsq(char *);
```

```
void create_input_tsq();
int  isModified(char);

main()
{
   memset(&screen1.screen1i.dfhms1??(Ø??),Ø,sizeof(screen1));
   EXEC CICS ADDRESS EIB(dfheiptr)
                     NOHANDLE;
   switch(dfheiptr->eibaid)
   {
      case DFHPF3:
         EXEC CICS SEND CONTROL ERASE NOHANDLE;
         EXEC CICS RETURN NOHANDLE;
         break;
      case DFHPF5:
         EXEC CICS RECEIVE MAP("ZLINKER ")
                           INTO(&screen1.screen1i)
                           NOHANDLE;
         memcpy(screen1.screen1o.messageo,"BROWSED!",8);
         display_tsq(screen1.screen1i.intsqi);
         screen1.screen1i.intsql = -1;
         EXEC CICS SEND CONTROL ERASEAUP NOHANDLE;
         EXEC CICS SEND MAP("ZLINKER ")
                        FROM(screen1.screen1o)
                        CURSOR
                        NOHANDLE;
         EXEC CICS RETURN
                   TRANSID(dfheiptr->eibtrnid)
                   NOHANDLE;
         break;
      case DFHPF6:
         EXEC CICS RECEIVE MAP("ZLINKER ")
                           INTO(&screen1.screen1i)
                           NOHANDLE;
         display_tsq(screen1.screen1i.outtsqi);
         EXEC CICS SEND CONTROL ERASEAUP NOHANDLE;
         memcpy(screen1.screen1o.messageo,"BROWSED!",8);
         screen1.screen1i.outtsql = -1;
         EXEC CICS SEND MAP("ZLINKER ")
                        FROM(screen1.screen1o)
                        CURSOR
                        NOHANDLE;
         EXEC CICS RETURN
                   TRANSID(dfheiptr->eibtrnid)
                   NOHANDLE;
         break;
      case DFHPF9:
         EXEC CICS RECEIVE MAP("ZLINKER ")
                           INTO(&screen1.screen1i)
                           NOHANDLE;
```

```
      create_input_tsq(screen1.screen1i.intsqi);
      EXEC CICS SEND CONTROL ERASEAUP NOHANDLE;
      memcpy(screen1.screen1o.messageo,"CREATED!",8);
      screen1.screen1i.programl = -1;
      EXEC CICS SEND MAP("ZLINKER ")
                     FROM(screen1.screen1o)
                     CURSOR
                     NOHANDLE;

      EXEC CICS RETURN
                TRANSID(dfheiptr->eibtrnid)
                NOHANDLE;
      break;
   default:
      EXEC CICS RECEIVE MAP("ZLINKER ")
                        INTO(&screen1.screen1i)
                        NOHANDLE;
      if(dfheiptr->eibresp == DFHRESP(MAPFAIL))
      {
         memset(&screen1.screen1i.dfhms1??(Ø??),Ø,sizeof(screen1));
         break;
      }
      else
      {
        do_commarea();
        EXEC CICS LINK PROGRAM(screen1.screen1i.programi)
                       COMMAREA(commarea)
                       LENGTH(commarea_length)
                       NOHANDLE;
        switch(dfheiptr->eibresp)
        {
           case DFHRESP(NORMAL) :
             memcpy(screen1.screen1o.messageo,"ALL OK!!",8);
             do_output_commarea();
             break;
           case DFHRESP(PGMIDERR) :
             memcpy(screen1.screen1o.messageo,"Program?",8);
             break;
           default:
             memcpy(screen1.screen1o.messageo,"Error!!!",8);
        }
      }
}
EXEC CICS SEND CONTROL ERASE NOHANDLE;
      screen1.screen1i.programl = -1;
EXEC CICS SEND MAP("ZLINKER ")
               FROM(screen1.screen1o)
               CURSOR
               NOHANDLE;
EXEC CICS RETURN
```

```
            TRANSID(dfheiptr->eibtrnid)
            NOHANDLE;
}

void do_commarea(void)
{
     int item = 1;
     short len  = 32767;
     char buffer??(32767??);
     memset(commarea,Øx ØØ,sizeof(commarea));
     for(commarea_length = Ø;dfheiptr->eibresp != DFHRESP(ITEMERR)
     && commarea_length < 32767;item++)
     {
        len  = 32767;
        EXEC CICS READQ TS QUEUE(screen1.screen1i.intsqi)
                           ITEM(item)
                           LENGTH(len)
                           INTO(buffer)
                           NOHANDLE;
         if(dfheiptr->eibresp != DFHRESP(NORMAL)) break;
         memcpy((commarea_ptr + commarea_length),buffer,len);
         commarea_length += len;
      }
     return;
}
void do_output_commarea(void)
{
     EXEC CICS DELETEQ TS QUEUE(screen1.screen1i.outtsqi) NOHANDLE;
     EXEC CICS WRITEQ TS QUEUE(screen1.screen1i.outtsqi)
                         LENGTH(commarea_length)
                         FROM(commarea_ptr)
                         NOHANDLE;
     return;
}

/*********************************************************************
 * Function: create_input_tsq                                        *
 * Description: Create the Input TSq                                 *
 * Copyright Wayne Robinson All Rights Reserved                      *
 *********************************************************************/

void create_input_tsq()               /* ==> Function Entry          */
{
/*-------------------------------------------------------------------*
 * local variables                                                   *
 *-------------------------------------------------------------------*/

struct
{
  unsigned char  transid??(5??);
```

```
  unsigned char  static_parms1??(14??);
  unsigned char  tsq_name??(8??);
  unsigned char  static_parms2??(37??);
} ceci_input;

/*-------------------------------------------------------------------*
 * code                                                              *
 *-------------------------------------------------------------------*/

if (!num_errors)
{
  memcpy(ceci_input.transid, "CECI ", 5);
  memcpy(ceci_input.static_parms1,
         "WRITEQ TS QU('",
         sizeof(ceci_input.static_parms1));
  memcpy(ceci_input.tsq_name,
         screen1.screen1i.intsqi,
         screen1.screen1i.intsql);
  memcpy(ceci_input.static_parms2,
         "') FROM('Change your test data here')",
         sizeof(ceci_input.static_parms2));
  EXEC CICS LINK
       PROGRAM("DFHECIP ")
       INPUTMSG(&ceci_input)
       INPUTMSGLEN(sizeof(ceci_input))
       NOHANDLE;
}
return;                                 /* <== Exit point            */
}                                       /* <== Function Exit         */
/*********************************************************************
 * Function: display_tsq                                             *
 * Description: Invoke CEBR                                          *
 * Copyright Wayne Robinson All Rights Reserved                      *
 *********************************************************************/
void display_tsq(                       /* ==> Function Entry        */
                 unsigned char  tsq_name??(8??))
{
/*-------------------------------------------------------------------*
 * local variables                                                   *
 *-------------------------------------------------------------------*/
struct
{
  unsigned char  transid??(5??);
  unsigned char  tsq_name??(8??);
} cebr_input;
/*-------------------------------------------------------------------*
 * code                                                              *
 *-------------------------------------------------------------------*/
memcpy(cebr_input.transid, "CEBR ", 5);
memcpy(cebr_input.tsq_name, tsq_name, sizeof(cebr_input.tsq_name));
```

```
EXEC CICS LINK
     PROGRAM("DFHEDFBR")
     INPUTMSG(&cebr_input)
     INPUTMSGLEN(sizeof(cebr_input))
     NOHANDLE;

return;                                 /* <== Exit point              */
}                                       /* <== Function Exit           */

LINKER mapset ZLINKER:
ZDEBUG   DFHMSD TYPE=MAP,                                               X
               STORAGE=AUTO,                                            X
               MODE=INOUT,                                              X
               LANG=C,                                                  X
               DSATTS=(COLOR,HILIGHT),                                  X
               MAPATTS=(COLOR,HILIGHT),                                 X
               TIOAPFX=YES,                                             X
               CTRL=FREEKB,                                             X
               EXTATT=YES
SCREEN1  DFHMDI SIZE=(24,80),                                           X
               LINE=1,                                                  X
               COLUMN=1
         DFHMDF POS=(1,1),                                              X
               LENGTH=23,                                               X
               COLOR=YELLOW,                                            X
               ATTRB=(ASKIP,PROT),                                      X
               INITIAL='LINKer           '
         DFHMDF POS=(1,52),                                             X
               LENGTH=28,                                               X
               COLOR=YELLOW,                                            X
               ATTRB=(ASKIP,PROT),                                      X
               INITIAL='http://www.lemon-tree.co.uk'
         DFHMDF POS=(2,1),                                              X
               LENGTH=80,                                               X
               COLOR=BLUE,                                              X
               ATTRB=(ASKIP,PROT),                                      X
               INITIAL='________________________________________________X
               ________________________________________'

         DFHMDF POS=(3,1),                                              X
               LENGTH=10,                                               X
               COLOR=TURQUOISE,                                         X
               ATTRB=(ASKIP,PROT),                                      X
               INITIAL='Program: '
PROGRAM  DFHMDF POS=(3,14),                                             X
               LENGTH=8,                                                X
               COLOR=GREEN,                                             X
               HILIGHT=UNDERLINE,                                       X
               ATTRB=(NORM,FSET),                                       X
```

```
              INITIAL='         '
         DFHMDF POS=(3,23),                                            X
               LENGTH=1,                                               X
               COLOR=BLUE,                                             X
               ATTRB=(ASKIP,PROT),                                     X
               INITIAL=' '
         DFHMDF POS=(5,1),                                             X
               LENGTH=11,                                              X
               COLOR=TURQUOISE,                                        X
               ATTRB=(ASKIP,PROT),                                     X
               INITIAL='Input TSQ: '
INTSQ    DFHMDF POS=(5,14),                                            X
               LENGTH=8,                                               X
               COLOR=GREEN,                                            X
               ATTRB=(UNPROT,FSET),                                    X
               HILIGHT=UNDERLINE,                                      X
               INITIAL='         '
         DFHMDF POS=(5,23),                                            X
               LENGTH=1,                                               X
               COLOR=BLUE,                                             X
               ATTRB=(ASKIP,PROT),                                     X
               INITIAL=' '
         DFHMDF POS=(7,1),                                             X
               LENGTH=12,                                              X
               COLOR=TURQUOISE,                                        X
               ATTRB=(ASKIP,PROT),                                     X
               INITIAL='Output TSQ: '
OUTTSQ   DFHMDF POS=(7,14),                                            X
               LENGTH=8,                                               X
               COLOR=GREEN,                                            X
               ATTRB=(UNPROT,NORM,FSET),                               X
               HILIGHT=UNDERLINE,                                      X
               CASE=MIXED,                                             X
               INITIAL='         '
         DFHMDF POS=(7,23),                                            X
               LENGTH=1,                                               X
               COLOR=BLUE,                                             X
               ATTRB=(ASKIP,PROT),                                     X
               INITIAL=' '
MESSAGE  DFHMDF POS=(10,1),                                            X
               LENGTH=8,                                               X
               COLOR=YELLOW,                                           X
               ATTRB=(ASKIP,NORM,FSET),                                X
               INITIAL='         '
         DFHMDF POS=(10,10),                                           X
               LENGTH=1,                                               X
               COLOR=NEUTRAL,                                          X
               ATTRB=(ASKIP,NORM)
         DFHMDF POS=(24,1),                                            X
               LENGTH=79,                                              X
```

```
         COLOR=BLUE,                                                  X
         ATTRB=(ASKIP,PROT),                                          X
         INITIAL='PF3=End  PF5=Browse Input TSQ  PF6=Browse OutpuX
         t TSQ  PF9=Create/Add Input TSQ'
DFHMSD TYPE=FINAL
END
```


CICS Performance Monitor workstation client and browser interfaces

In the previous article ('CICS Performance Monitor for z/OS', *CICS Update*, issue 216, November 2003), we looked at the general facilities provided by CICS Performance Monitor for z/OS. In this article we will look at the facilities provided by the CICS PM end user interfaces, namely the CICS PM workstation client interface and the CICS PM browser interface, which utilize the CICSPlex SM Web user interface server.

CICS PM WORKSTATION CLIENT

The CICS PM workstation client is a Java swing application that relates scopes – CICS regions, resource classes, and events. The workstation client also provides facilities for the definition/modification of thresholds and can launch the WUI to view detailed information relative to the event being investigated.

Context and scope are automatically assigned by cursor positioning on the topology pane, and WUI launch can also be to the resourceclass or the resourceclass and instance (eg PROGRAM(*) or PROGRAM(ABC*)). When navigating from the client to the WUI, no additional sign-on by the user is required because the client/WUI utilizes RACF passticket support.

A Java Runtime Environment (JRE) is required on the client

workstation, and the client code is installable on Windows using the Microsoft Software Installer (MSI) in either attended or unattended modes (Windows NT4 service pack 6; Windows 2000 and XP are supported).

Observing triggered thresholds

After installing the workstation client and launching the URL of the WUI server address space, the screen in Figure 1 is presented. To the left of the screen is a CICSplex topology tree in familiar Windows explorer format. This tree is constructed through interrogation of topology data maintained by CICSPlex SM on what CICSplexes exist, what scopes are defined, and what systems constitute those scopes. You can set the context and scope by positioning the cursor at the relevant point on the tree.

At the bottom of the screen is an area for displaying events that

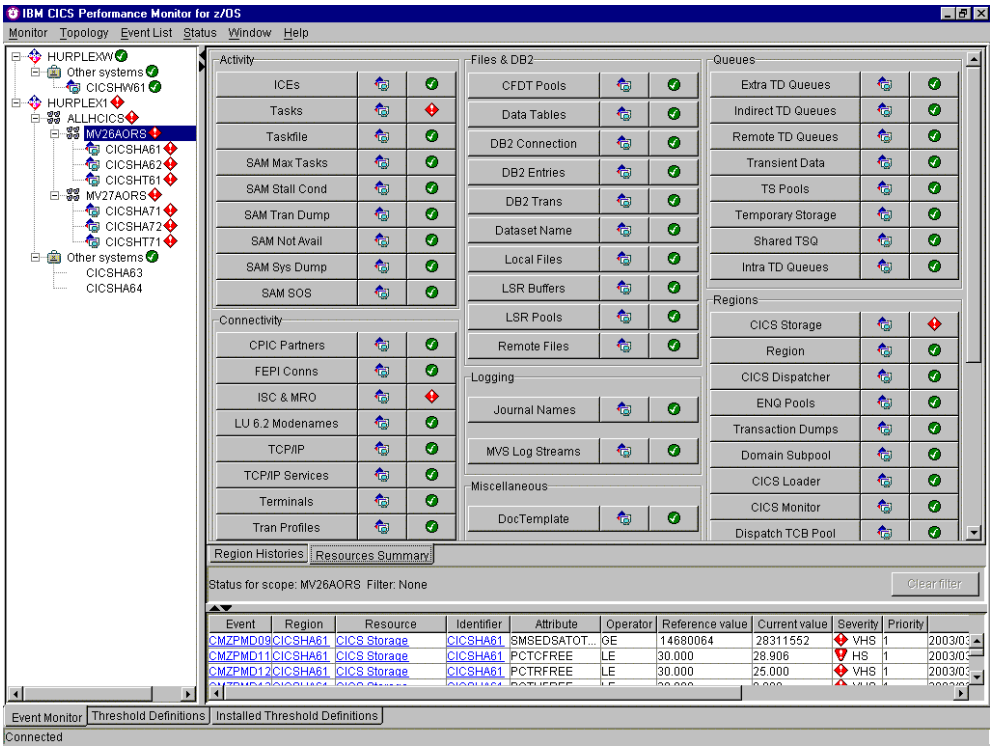

Figure 1: CICS Performance Monitor for z/OS

are currently active in the systems. The currently active events are determined by reading the EVENT records from CICSPlex SM for the current context and scope.

The event data is now summarized in two forms: a pass through the EVENT data will identify the highest severities against each CICS system in the topology tree. This tree can therefore be annotated with the highest severity, and percolated up through the scopes and contexts. Options exist to display only systems in the topology tree that have events associated with them. Similarly, the expanded/contracted state of the tree is remembered for any subsequent client launch.

A second pass through the EVENT data can identify which resources are affected in the current context and scope, and the severity. This allows you to build the resource summary pane, which occupies the central portion of the screen. Each entry on

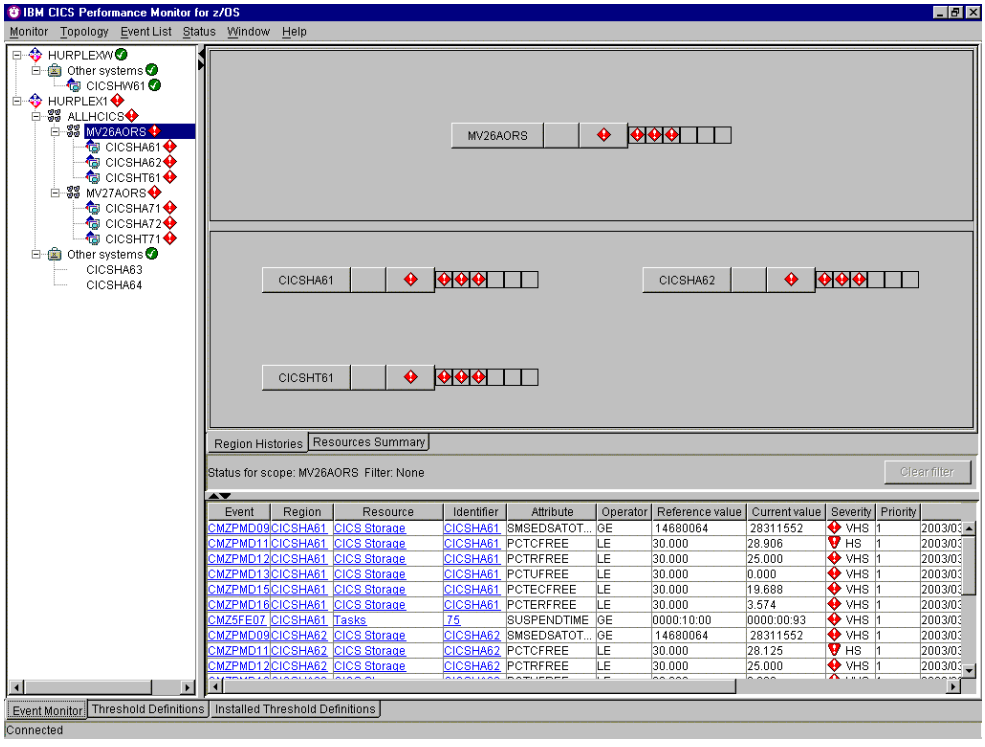

Figure 2: Observing event history

the screen is actually a triplet of buttons – resource class, history, and filter.

The resource class button enables you to launch the WUI for that resource class under the current context and scope. The history button displays a history pane (see Figure 2), and the filter button will further filter the EVENT pane to this resource class. To undo this, press the clear filter button.

Observing event history

You can observe the history of an event by selecting the region history tab. This shows the history of events over the last six sample periods. By looking in the EVENT pane, you can also see more clearly some events with their associated data and hyperlink fields. You can hyperlink from Event to the event definition in the client (CMZA381); from Region to the WUI for that CICS region (CICSREGION CICSHA61); from Resource to that resource

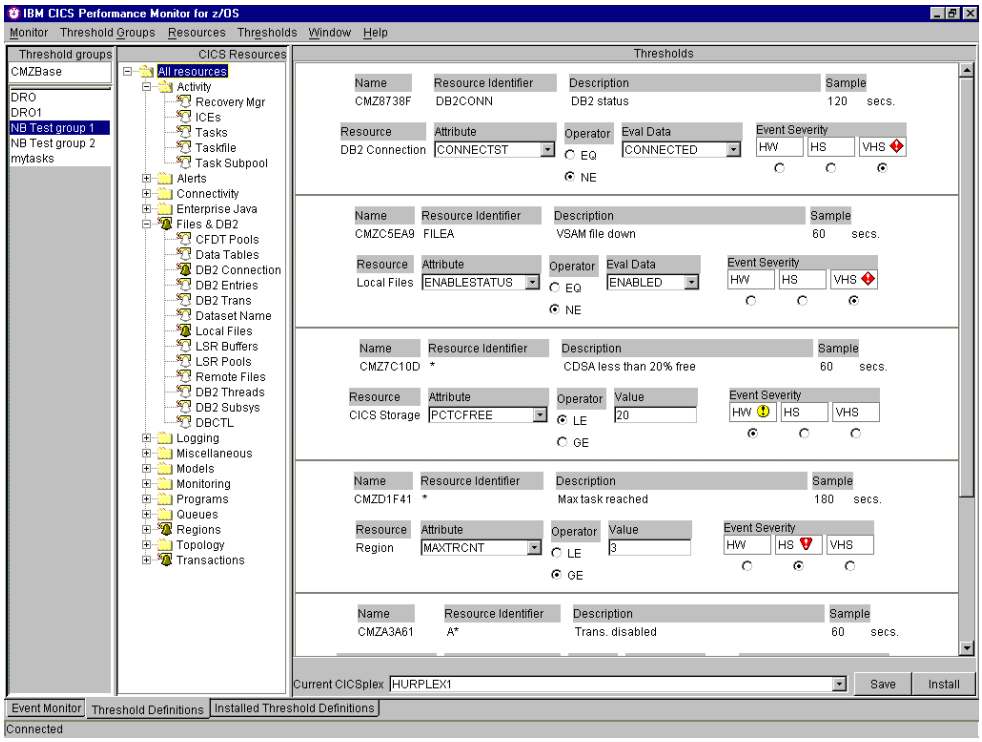

Figure 3: Threshold definitions

class in the given region (transaction(*)); and from Identifier to the resource class with that identifier (transaction(AORD)).

Defining and modifying thresholds for exception management

You can define and install thresholds simply by choosing the Threshold Definition tab. The view displayed is shown in Figure 3. CICS PM ships with a sample set of thresholds for instant use. In addition, users can define groups of their own thresholds. You can create thresholds by selecting a resource class in the tree view, selecting an attribute via drop-downs, and defining the comparative test to be performed. When definitions have been completed, they can be activated into a (set of) CICS systems via a single install command. Definitions can also be saved to the CICSPlex SM repository for future usage.

Observing currently active thresholds

To identify which thresholds are currently being tested across the

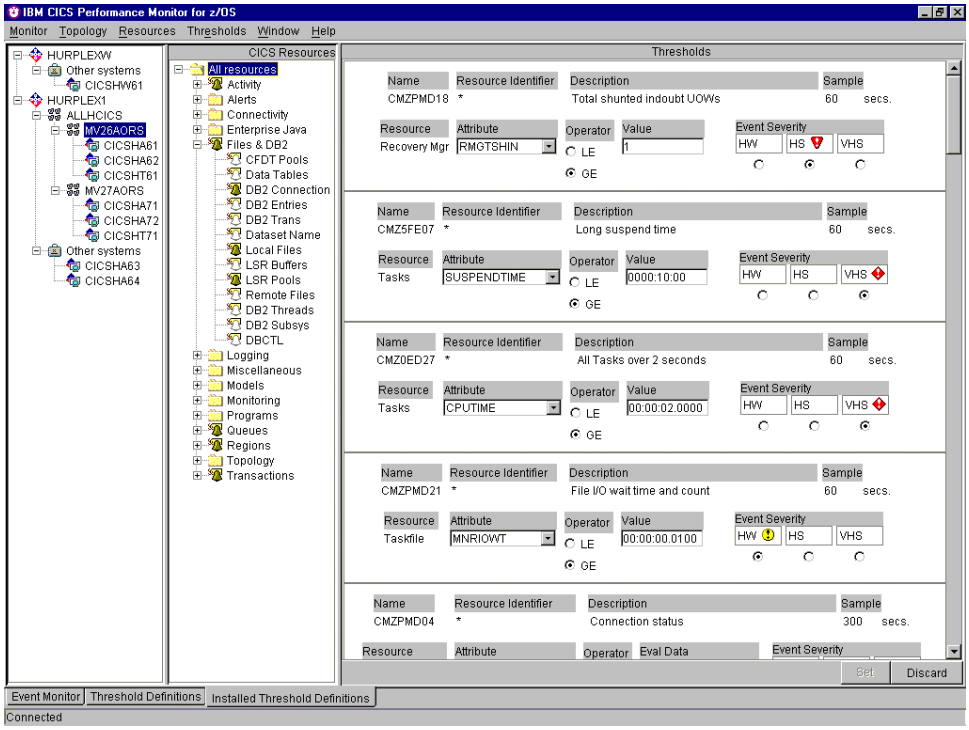

Figure 4: Installed thresholds

CICSplex, choose the Installed thresholds tab, and the screen in Figure 4 will be displayed. From here, you can discard active thresholds by selecting the threshold in the central pane and pressing the discard button.

CICS PERFORMANCE MONITOR SERVER PROGRAM

The server program that works with the CICS PM workstation client is installed on the WUI server region. These routines provide the ability to create/modify/delete/activate/list/copy appropriate CICSPlex SM definitions in response to client requests. This comprises:

- CMZSRVR – which analyses the request stream coming from the client program.
- CMZRTDP – which maps requests to RTAGROUP definitions.
- CMZGVAL – which maps requests to RTADEF/EVADEF definitions.
- CMZIQWUI – which provides service to interpret TCP/IP request streams.

Two service routines, CMZTREY and CMZNGEN, control EYUDA translation and name generation respectively.

Although not required in order to utilize the CICS PM client, for those familiar with CICSPlex SM, these routines also provide more complex activities such as install and discard of thresholds.

Install a group of definitions

Look up CSYSDEF to see if RTASPEC is already defined:

- Yes – add a group of definitions to this RTASPEC.
- No – create RTASPEC with the name of CSYSDEF. Add RTAGROUP to RTASPEC. Add RTASPEC to CSYSDEF. Install definitions.

Discard a group of definitions

Look up CSYSDEF to identify RTASPEC:

- CICS PM created RTASPEC. Remove RTASCOPE and delete RTASPEC. Remove from RTAACTV.
- Not created by CICS PM. Remove from users RTSPEC. Remove from RTAACTV.

CICS PERFORMANCE MONITOR WEB USER INTERFACE VIEWSETS

So far we have concentrated on threshold event management. However, customers spend a great deal of time looking at detailed information about the current state of their CICS systems. It is not surprising, then, that CICS PM provides access to an extensive array of information regarding CICS systems and their resources. This information is displayed to the end user when he or she launches his or her Internet browser to the WUI server.

The WUI server processes requests, formats responses, and helps you navigate the data via supplied WUI menus and viewsets. These views have been tailored for performance management in both data layout and navigation. In addition, you can take actions directly from the views via command buttons (eg OPEN/CLOSE a file). You can navigate directly to these views from the CICS PM workstation client. In addition, you can also use the WUI editor to create your own menus and viewsets.

Figures 5 to 8 show examples of typical WUI views. Hyperlink fields are shown as underlines and can be conditional on content. Arrows denote sort ascending and descending buttons, and combined arrows identify summarization buttons. Hover help is also provided.

Figure 5 shows the Home menu of the CICS PM browser interface. A three-part frameset is used. There’s an assistance frame (top). A navigation frame (left) provides a familiar Windows Explorer-like navigation tree with expandable/collapsible nodes. Leaves in the tree drive queries for data, which is displayed in the

Figure 5: Home menu

main work area (centre).

Figure 6 shows the regions tab opened to provide various queries on CICS region data.

Figure 7 is an example of a tabuler layout showing a row for each task instance in the current context and scope. Filter criteria may be entered in the filter area at the top of the frame.

Figure 8 is an example of a new WUI form layout (matrix form). Hyperlinks can be taken from any of the underlined items on display.

EXAMPLE SCENARIO

As an example of the use of CICS PM, consider the following scenario.

Figure 6: Regions tabs

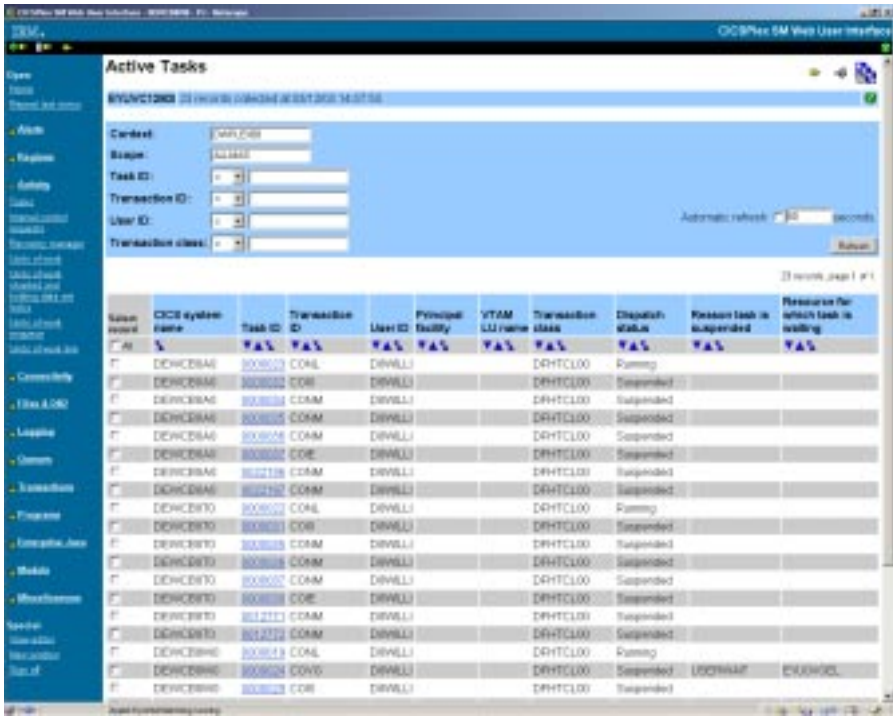

Figure 7: Tabular layout

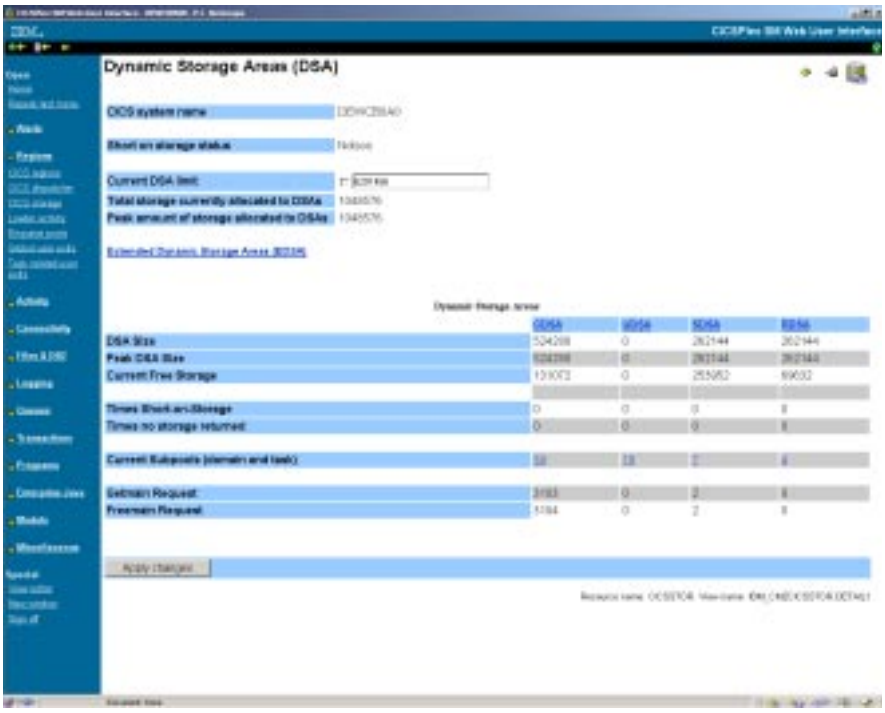

Figure 8: WUI form

In Figure 9 we have the workstation client with some low-level events active in the system.

In Figure 10 a CICS region goes short on storage, followed, in Figure 11, by a potential stall situation.

In Figure 12, we launch to the TASK display for that region and see a task suspended on UDSA.

In Figure 13 we decide to purge the transaction to see whether this clears the problem.

Of course, as is common with graphical interfaces, we get a confirmation panel – see Figures 14 and 15.

Figure 16 shows the message area confirming that the purge worked. And the events have gone from the workstation client in Figure 17.

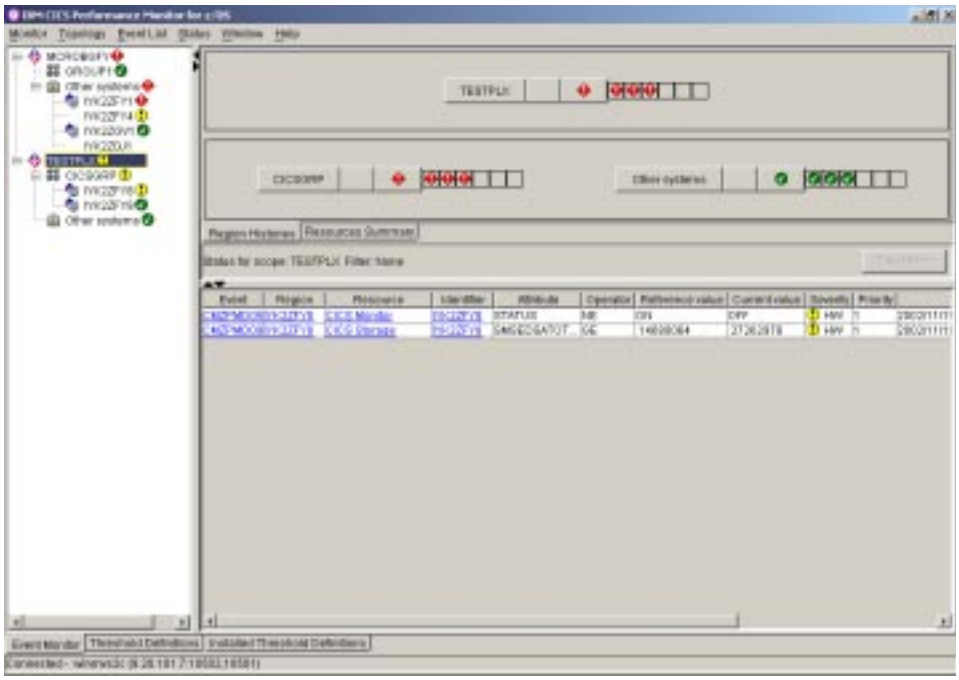

Figure 9: Low-level activity

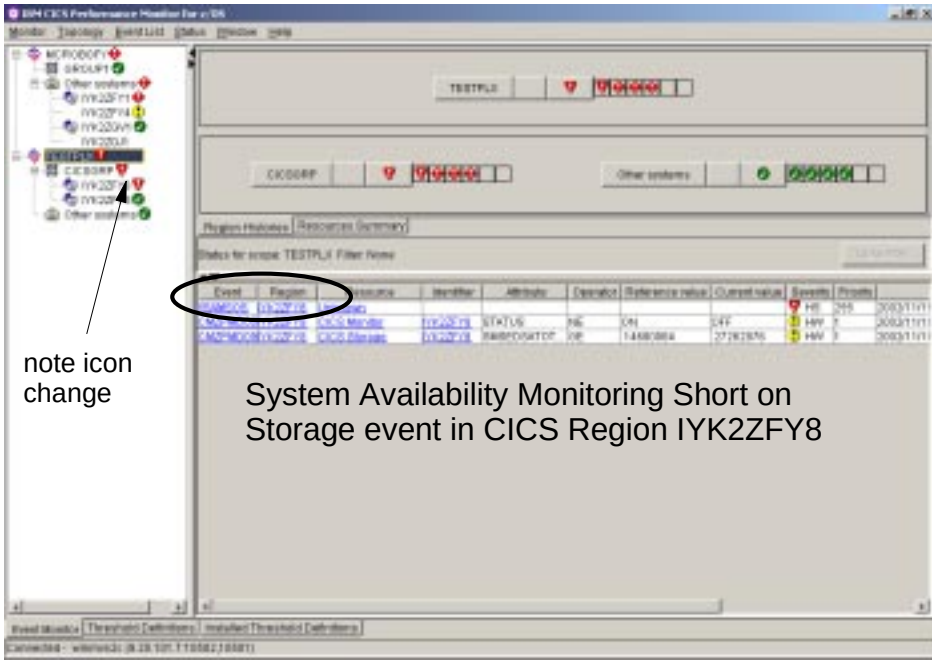

Figure 10: Short on storage

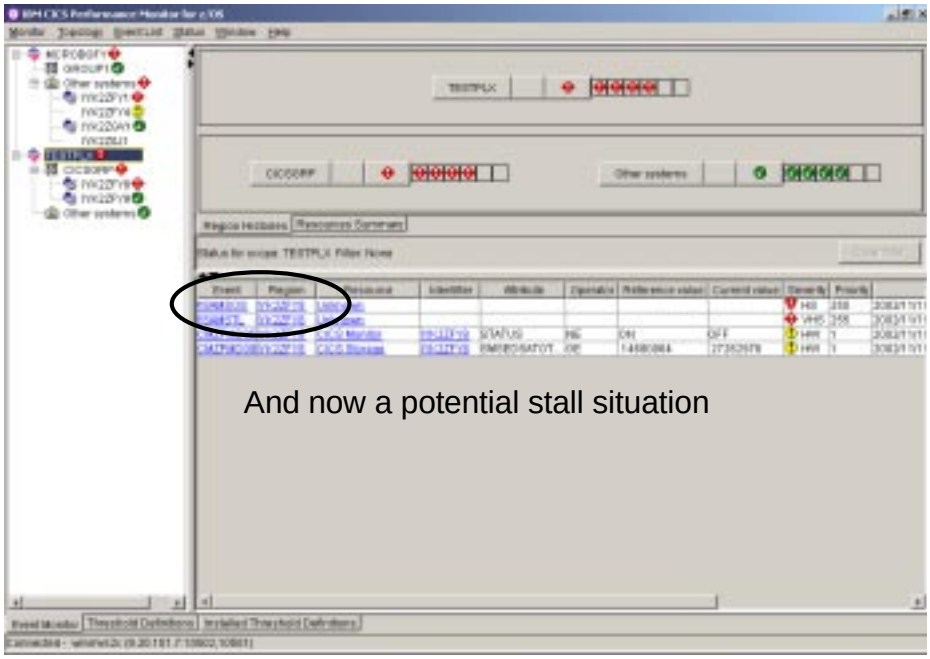

Figure 11: Potential stall

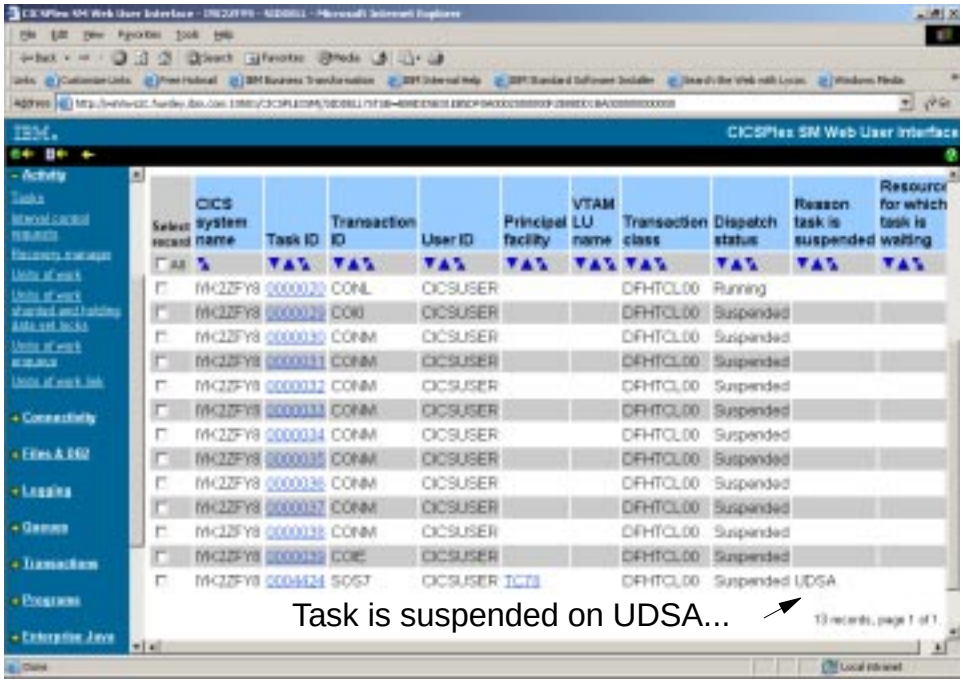

Figure 12: Suspended task

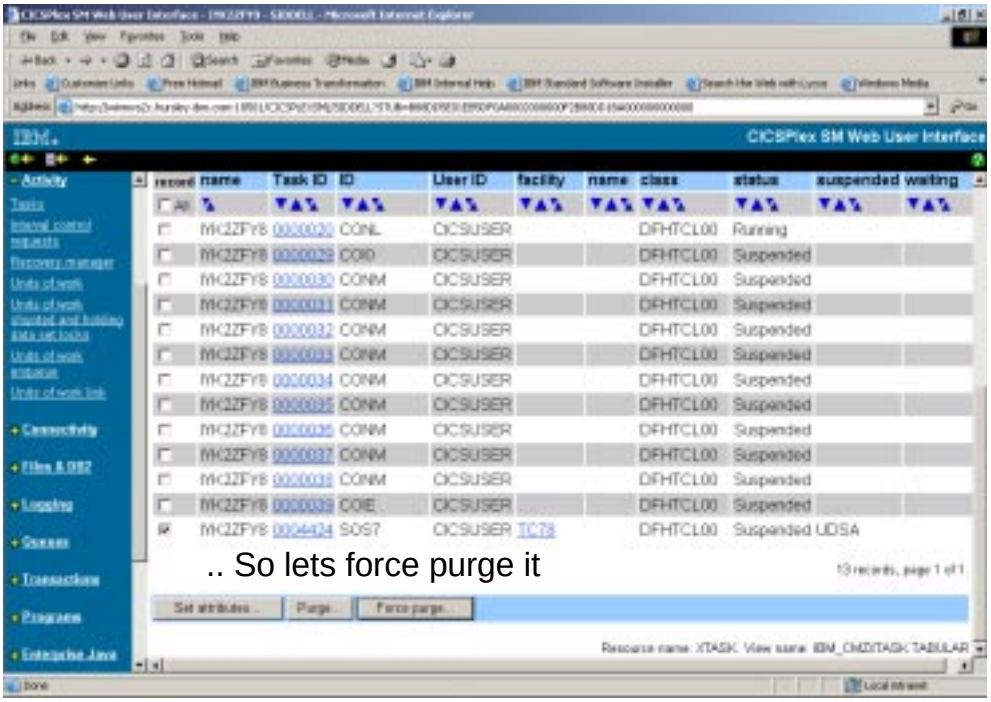

Figure 13: Purge

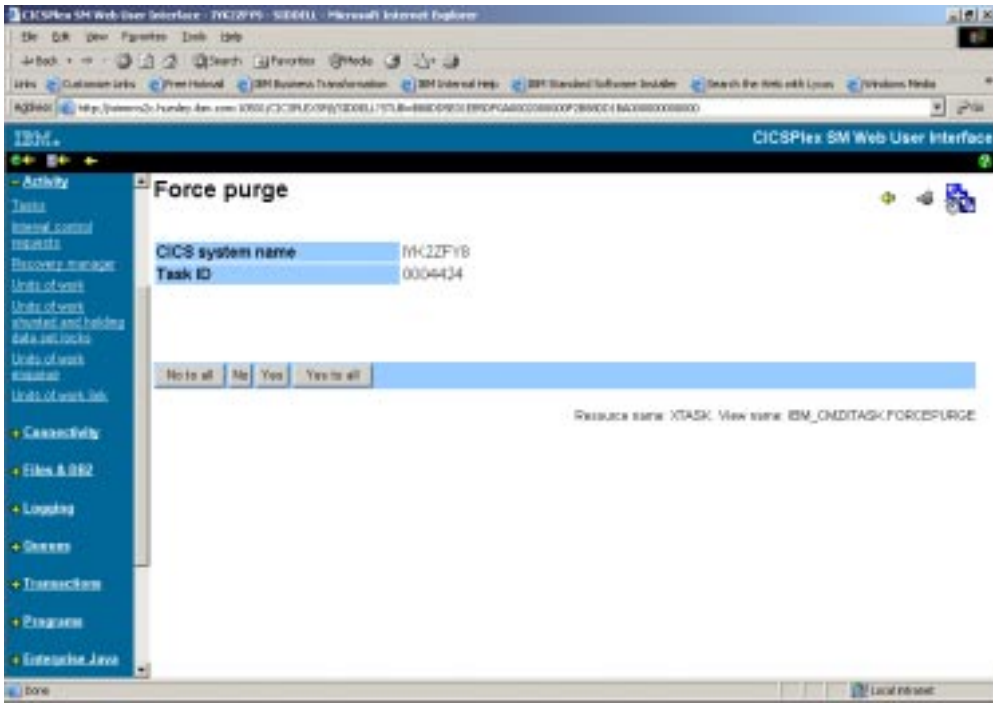

Figure 14: Confirmation panel

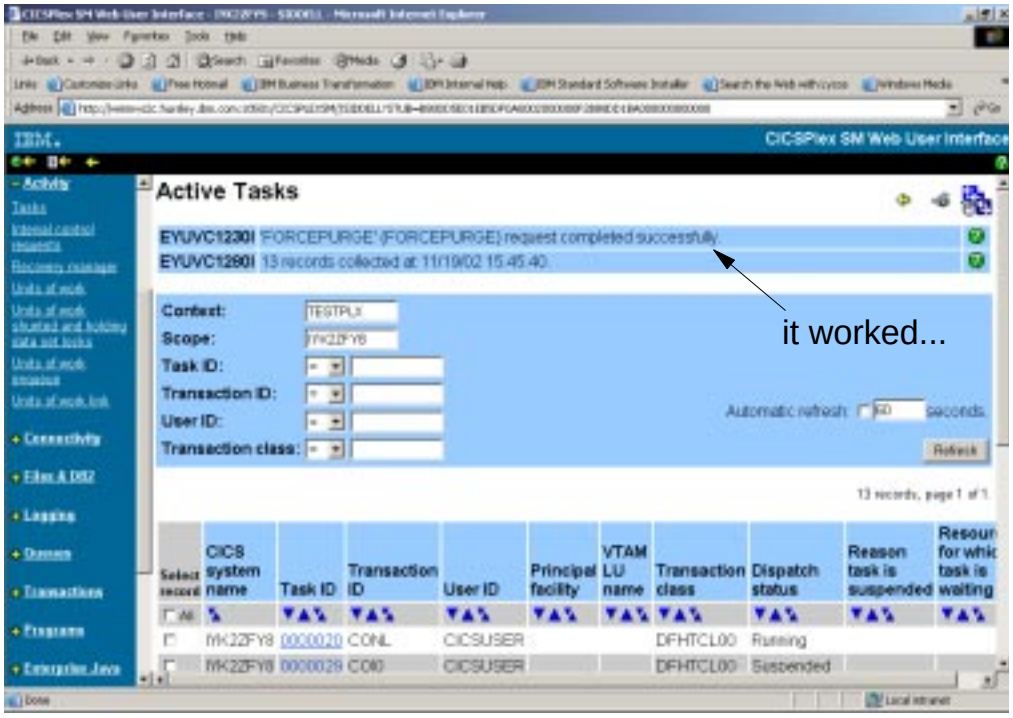

Figure 15: Message area confirmation

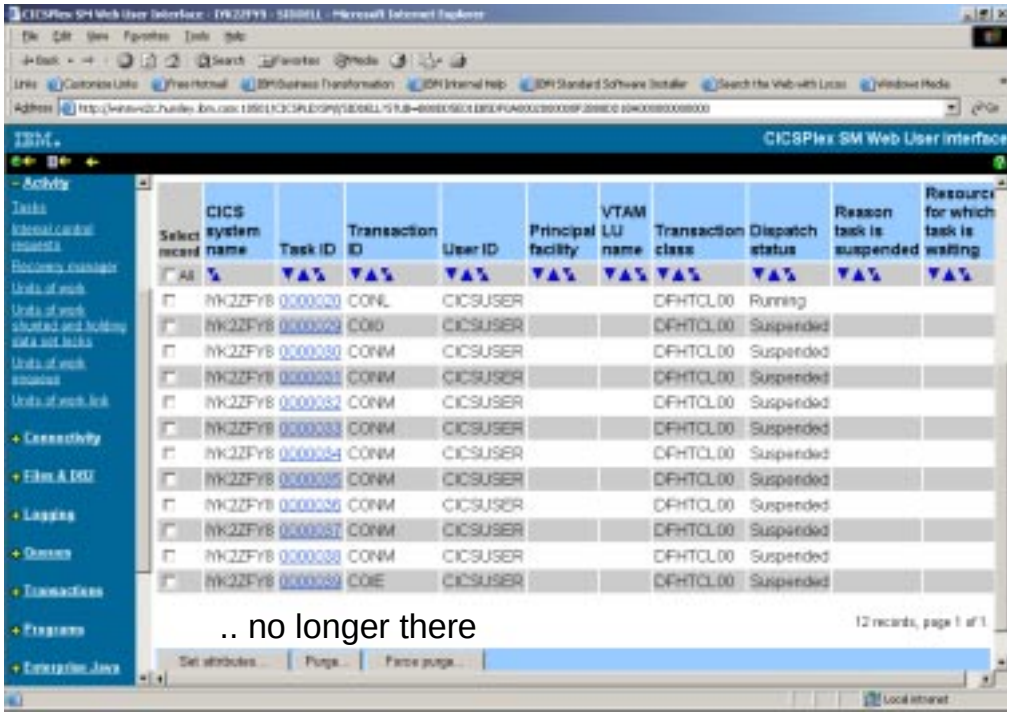

Figure 16: Events are no longer shown

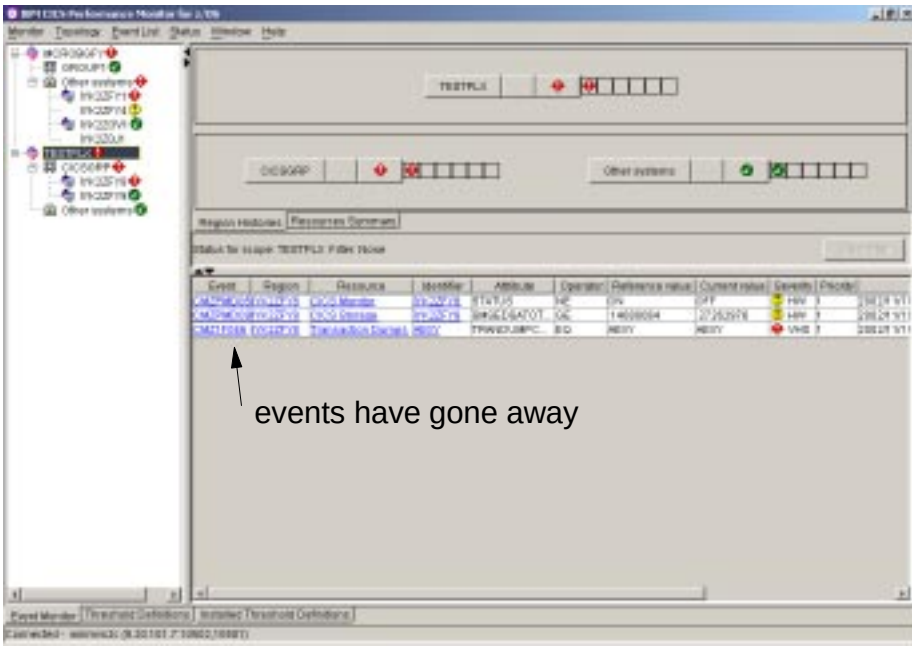

Figure 17: Events have gone

SUMMARY

As we have seen, the workstation client allows you to detect early performance problems and provides easy-to-use, intuitive, cross-system navigation for fast problem determination.

The browser interface enables you to access all CICS systems and resource information and make changes to key system and resource parameters.

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

CICS file browse – part 2

This month we conclude the code for an application that allows users to browse the application error log file.

```
     ELSE
* no column argument was entered
         PERFORM Ø58Ø-SEARCH-LOGREC THRU
                 Ø58Ø-SEARCH-LOGREC-EXIT
     END-IF.
 Ø57Ø-FIND-STRING-EXIT.
     EXIT.
 Ø58Ø-SEARCH-LOGREC.
* bump through the record one byte at a time trying to match on
* the string
     IF  WS-DISP < 1
         MOVE 3ØØ29                TO  WS-DISP
         GO TO Ø58Ø-SEARCH-LOGREC-EXIT
     END-IF.
     IF  WS-LOG-RECORD(WS-DISP:WS-FIND-LENGTH) =
         WS-FIND-STRING
         SET WS-88-STRING-FOUND TO TRUE
         GO TO Ø58Ø-SEARCH-LOGREC-EXIT
     ELSE
         SUBTRACT 1 FROM WS-DISP
         GO TO Ø58Ø-SEARCH-LOGREC
     END-IF.
 Ø58Ø-SEARCH-LOGREC-EXIT.
     EXIT.
 Ø6ØØ-PARSE-COMMAND.
     MOVE WS-MAP-CMMND             TO  WS-COMMAND-LINE.
     MOVE 5Ø                       TO  WS-CMMND-SUB.
* search backward through the command looking for the
* first non-blank
     PERFORM UNTIL
         WS-COMMAND-CHAR(WS-CMMND-SUB) > ' '
         SUBTRACT 1 FROM WS-CMMND-SUB
     END-PERFORM.
* save the end of the command entered
     MOVE WS-CMMND-SUB             TO  WS-CMMND-END.
     MOVE 1                        TO  WS-CMMND-SUB.
* search the command again from the beginning looking for the
* first non-blank
     PERFORM UNTIL
         WS-COMMAND-CHAR(WS-CMMND-SUB) > ' '
         ADD 1 TO WS-CMMND-SUB
     END-PERFORM.
```

```
     MOVE WS-CMMND-SUB             TO  WS-CMMND-START.
    COMPUTE WS-CMMND-LENGTH = (WS-CMMND-END + 1) - WS-CMMND-START.
* move the command so it is left justified
     MOVE WS-COMMAND-LINE(WS-CMMND-START:WS-CMMND-LENGTH)
                                   TO
                          WS-HOLD-COMMAND-LINE(1:WS-CMMND-LENGTH).
* parse the command entered
     EVALUATE TRUE
         WHEN WS-CMMND-LENGTH = 1 AND
             WS-HOLD-COMMAND-LINE(1:1) = 'M'
         WHEN WS-CMMND-LENGTH = 3 AND
             WS-HOLD-COMMAND-LINE(1:3) = 'MAX'
                 SET WS-88-SCROLL-MAX TO TRUE
         WHEN WS-CMMND-LENGTH = 3 AND
             WS-HOLD-COMMAND-LINE(1:3) = 'TOP'
                 SET WS-88-SCROLL-TOP TO TRUE
         WHEN WS-CMMND-LENGTH = 3 AND
             WS-HOLD-COMMAND-LINE(1:3) = 'BOT'
         WHEN WS-CMMND-LENGTH = 6 AND
             WS-HOLD-COMMAND-LINE(1:6) = 'BOTTOM'
                 SET WS-88-SCROLL-BOT TO TRUE
         WHEN WS-HOLD-COMMAND-LINE(1:3) = 'HEX'
             PERFORM Ø61Ø-SET-HEX-MODE THRU
                     Ø61Ø-SET-HEX-MODE-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:2) = 'T '
             MOVE 3                TO  WS-SEARCH-TIME-START
             PERFORM Ø62Ø-PARSE-TIME THRU
                     Ø62Ø-PARSE-TIME-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:5) = 'TIME '
             MOVE 6                TO  WS-SEARCH-TIME-START
             PERFORM Ø62Ø-PARSE-TIME THRU
                     Ø62Ø-PARSE-TIME-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:5) = 'TASK'
             PERFORM Ø63Ø-PARSE-TASK THRU
                     Ø63Ø-PARSE-TASK-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:2) = 'F '
             MOVE 3                TO  WS-FIND-START
             PERFORM Ø64Ø-PARSE-FIND THRU
                     Ø64Ø-PARSE-FIND-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:5) = 'FIND '
             MOVE 6                TO  WS-FIND-START
             PERFORM Ø64Ø-PARSE-FIND THRU
                     Ø64Ø-PARSE-FIND-EXIT
         WHEN WS-HOLD-COMMAND-LINE(1:WS-CMMND-LENGTH) NUMERIC AND
             WS-88-SCROLL-KEY
                 PERFORM Ø65Ø-PARSE-AMT THRU
                         Ø65Ø-PARSE-AMT-EXIT
         WHEN OTHER
             MOVE 'INVALID COMMAND ENTERED'
                                   TO  WS-ERROR-DISPLAY
```

```
            SET WS-88-DISPLAY-ERROR TO TRUE
    END-EVALUATE.

    MOVE SPACES                   TO  WS-MAP-CMMND.
* if the find direction is set, set the scroll
* direction accordingly
    EVALUATE TRUE
        WHEN WS-88-FIND-NEXT
            SET WS-88-SCROLL-FORWARD TO TRUE
        WHEN WS-88-FIND-PREV
            SET WS-88-SCROLL-BACK TO TRUE
    END-EVALUATE.
 0600-PARSE-COMMAND-EXIT.
    EXIT.
 0610-SET-HEX-MODE.
* HEX command entered
    EVALUATE TRUE
* no argument was entered, so just toggle
        WHEN WS-CMMND-LENGTH = 3
            IF  WS-88-HEX-ON
                SET WS-88-HEX-OFF TO TRUE
            ELSE
                SET WS-88-HEX-ON TO TRUE
            END-IF
        WHEN WS-CMMND-LENGTH = 6 AND
            WS-HOLD-COMMAND-LINE(5:2) = 'ON'
            SET WS-88-HEX-ON TO TRUE
        WHEN WS-CMMND-LENGTH = 7 AND
            WS-HOLD-COMMAND-LINE(5:3) = 'OFF'
            SET WS-88-HEX-OFF TO TRUE
        WHEN OTHER
            MOVE 'INVALID HEX MODE SPECIFIED'
                              TO  WS-ERROR-DISPLAY
            SET WS-88-DISPLAY-ERROR TO TRUE
    END-EVALUATE.
 0610-SET-HEX-MODE-EXIT.
    EXIT.
 0620-PARSE-TIME.
* TIME command entered
    COMPUTE WS-SEARCH-TIME-LENGTH = (WS-CMMND-LENGTH + 1) -
                                    WS-SEARCH-TIME-START.
    MOVE WS-SEARCH-TIME-START     TO  WS-CMMND-SUB.
    MOVE 1                        TO  WS-TIME-SUB.
* move as much of the time that was entered to the save area
    PERFORM WS-SEARCH-TIME-LENGTH TIMES
        IF  WS-HOLD-COMMAND-LINE(WS-CMMND-SUB:1) = ':'
            ADD 1 TO WS-CMMND-SUB
        ELSE
            IF  WS-TIME-SUB < 7
                MOVE WS-HOLD-COMMAND-LINE(WS-CMMND-SUB:1)
```

```
                                TO
                            WS-SEARCH-TIME-NUMERIC(WS-TIME-SUB:1)
              END-IF
              ADD 1 TO WS-TIME-SUB
                       WS-CMMND-SUB
          END-IF
      END-PERFORM.
* time can be any part HH, HHMM or HHMMSS but must be an
* even number
      SUBTRACT 1 FROM WS-TIME-SUB.
      DIVIDE WS-TIME-SUB BY 2 GIVING WS-QUOTIENT
          REMAINDER WS-REMAINDER.
      IF  WS-REMAINDER = 1
          MOVE 'TIME MUST BE AN EVEN NUMBER OF DIGITS'
                                TO  WS-ERROR-DISPLAY
          SET WS-88-DISPLAY-ERROR TO TRUE
          GO TO Ø62Ø-PARSE-TIME-EXIT
      END-IF.
* make sure the time (not including colons) is numeric
      IF  WS-SEARCH-TIME-NUMERIC IS NUMERIC
          NEXT SENTENCE
      ELSE
          MOVE 'TIME ENTERED IS NOT NUMERIC'
                                TO  WS-ERROR-DISPLAY
          SET WS-88-DISPLAY-ERROR TO TRUE
          GO TO Ø62Ø-PARSE-TIME-EXIT
      END-IF.

      IF  WS-TIME-SUB > 6
          MOVE 'TIME MUST BE 6 DIGITS OR LESS'
                                TO  WS-ERROR-DISPLAY
          SET WS-88-DISPLAY-ERROR TO TRUE
          GO TO Ø62Ø-PARSE-TIME-EXIT
      END-IF.
* move time so it takes the form HH:MM:SS to match the log record
      MOVE WS-SEARCH-TIME-NUMERIC(1:2)
                                TO  WS-SEARCH-TIME(1:2).
      MOVE WS-SEARCH-TIME-NUMERIC(3:2)
                                TO  WS-SEARCH-TIME(4:2).
      MOVE WS-SEARCH-TIME-NUMERIC(5:2)
                                TO  WS-SEARCH-TIME(7:2).
      EVALUATE WS-TIME-SUB
          WHEN 2
              ADD 1             TO  WS-TIME-SUB
          WHEN 4
          WHEN 6
              ADD 2             TO  WS-TIME-SUB
      END-EVALUATE.
      SET WS-88-SCROLL-TIME TO TRUE.
 Ø62Ø-PARSE-TIME-EXIT.
```

```
     EXIT.
 Ø63Ø-PARSE-TASK.
* TASK was entered
     COMPUTE WS-SEARCH-TASK-LENGTH = WS-CMMND-LENGTH - 5.
     COMPUTE WS-TASK-OFFSET = 15 - WS-SEARCH-TASK-LENGTH.
* right justify task number in a zero fill field
     MOVE WS-HOLD-COMMAND-LINE(6:WS-SEARCH-TASK-LENGTH)
                                TO
             WS-SEARCH-TASK(WS-TASK-OFFSET:WS-SEARCH-TASK-LENGTH).
     SET WS-88-SCROLL-TASK TO TRUE.
 Ø63Ø-PARSE-TASK-EXIT.
     EXIT.
 Ø64Ø-PARSE-FIND.
* F or FIND was entered
     INITIALIZE  WS-FIND-STRING
                 WS-FIND-LENGTH
                 WS-FIND-COLUMN.
* set find next as default
     SET WS-88-FIND-NEXT TO TRUE.
     COMPUTE WS-FIND-LENGTH = (WS-CMMND-LENGTH + 1) -
                                WS-FIND-START.
     MOVE WS-FIND-START            TO  WS-CMMND-SUB.
     MOVE 1                        TO  WS-FIND-SUB.
* if the string is quoted, parse out everything between the quotes
     IF  WS-HOLD-COMMAND-LINE(WS-FIND-START:1) = QUOTE
         COMPUTE WS-POINTER = WS-FIND-START + 1
         UNSTRING WS-HOLD-COMMAND-LINE
             DELIMITED BY QUOTE
             INTO WS-UNSTRING-AREA
             DELIMITER IN WS-DELIMITER
             COUNT IN WS-COUNT
             POINTER WS-POINTER
         END-UNSTRING
         IF  WS-DELIMITER = QUOTE
             SET WS-88-QUOTED-STRING TO TRUE
             MOVE WS-UNSTRING-AREA TO  WS-FIND-STRING
             MOVE WS-COUNT         TO  WS-FIND-LENGTH
         ELSE
             MOVE 'UNBALANCED QUOTES'
                                   TO  WS-ERROR-DISPLAY
             SET WS-88-DISPLAY-ERROR TO TRUE
             INITIALIZE  WS-FIND-STRING
                         WS-FIND-LENGTH
                         WS-FIND-COLUMN
                         WS-FIND-DIRECTION
             GO TO Ø64Ø-PARSE-FIND-EXIT
         END-IF
     END-IF.
* no quote was found to begin the string
* if there are any spaces, either the string contains spaces or
```

```
* arguments have been entered
* in either case, the string must have quotes
     IF  WS-88-UNQUOTED-STRING
         INSPECT
             WS-HOLD-COMMAND-LINE(WS-FIND-START:WS-FIND-LENGTH)
             TALLYING WS-TALLY
             FOR ALL ' '
         IF  WS-TALLY > Ø
             MOVE 'PUT STRING IN QUOTES'
                                  TO  WS-ERROR-DISPLAY
             SET WS-88-DISPLAY-ERROR TO TRUE
             INITIALIZE  WS-FIND-STRING
                         WS-FIND-LENGTH
                         WS-FIND-COLUMN
                         WS-FIND-DIRECTION
             GO TO Ø64Ø-PARSE-FIND-EXIT
         END-IF
* the string has no spaces or arguments
         MOVE WS-HOLD-COMMAND-LINE(WS-FIND-START:WS-FIND-LENGTH)
                                  TO  WS-FIND-STRING
         INITIALIZE WS-FIND-COLUMN
         SET WS-88-FIND-NEXT TO TRUE
         SET WS-88-SCROLL-FIND TO TRUE
         GO TO Ø64Ø-PARSE-FIND-EXIT
     END-IF.
* the string was in quotes and no arguments were entered
     IF  WS-POINTER = WS-CMMND-LENGTH + 1
         INITIALIZE WS-FIND-COLUMN
         SET WS-88-FIND-NEXT TO TRUE
         SET WS-88-SCROLL-FIND TO TRUE
         GO TO Ø64Ø-PARSE-FIND-EXIT
     END-IF.
* there must be a space between the closing quote and the argument
     IF  WS-HOLD-COMMAND-LINE(WS-POINTER:1) = ' '
         CONTINUE
     ELSE
         MOVE 'INVALID PARAMETER'  TO  WS-ERROR-DISPLAY
         SET WS-88-DISPLAY-ERROR TO TRUE
         INITIALIZE  WS-FIND-STRING
                     WS-FIND-LENGTH
                     WS-FIND-COLUMN
                     WS-FIND-DIRECTION
         GO TO Ø64Ø-PARSE-FIND-EXIT
     END-IF.
     INITIALIZE WS-UNSTRING-AREA
                WS-COUNT.
     ADD 1 TO WS-POINTER.
* get the first argument
     UNSTRING WS-HOLD-COMMAND-LINE
         DELIMITED BY ' ' OR WS-HEX-ZERO
```

```
            INTO WS-UNSTRING-AREA
            COUNT IN WS-COUNT
            POINTER WS-POINTER
        END-UNSTRING.
        EVALUATE TRUE
    * PREV argument entered
            WHEN WS-COUNT = 4 AND
                WS-UNSTRING-AREA(1:4) = 'PREV'
                SET WS-88-FIND-PREV TO TRUE
                SET WS-88-SCROLL-FIND TO TRUE
            WHEN WS-COUNT < 6 AND
    * column argument entered
                WS-UNSTRING-AREA(1:WS-COUNT) IS NUMERIC
                COMPUTE WS-OFFSET = 6 - WS-COUNT
    * right justify the column
                MOVE WS-UNSTRING-AREA(1:WS-COUNT)
                                      TO
                           WS-FIND-COLUMN(WS-OFFSET:WS-COUNT)
            WHEN OTHER
                MOVE 'INVALID PARAMETER'
                                      TO  WS-ERROR-DISPLAY
                SET WS-88-DISPLAY-ERROR TO TRUE
                INITIALIZE  WS-FIND-STRING
                            WS-FIND-LENGTH
                            WS-FIND-COLUMN
                            WS-FIND-DIRECTION
                GO TO Ø64Ø-PARSE-FIND-EXIT
        END-EVALUATE.
    * only one argument
        IF  WS-POINTER = WS-CMMND-LENGTH + 2
            SET WS-88-SCROLL-FIND TO TRUE
            GO TO Ø64Ø-PARSE-FIND-EXIT
        END-IF.
        IF  WS-HOLD-COMMAND-LINE(WS-POINTER:1) > ' '
            CONTINUE
        ELSE
            MOVE 'INVALID PARAMETER'  TO  WS-ERROR-DISPLAY
            SET WS-88-DISPLAY-ERROR TO TRUE
            INITIALIZE  WS-FIND-STRING
                        WS-FIND-LENGTH
                        WS-FIND-COLUMN
                        WS-FIND-DIRECTION
            GO TO Ø64Ø-PARSE-FIND-EXIT
        END-IF.
        INITIALIZE WS-UNSTRING-AREA
                   WS-COUNT.
    * get the second argument
        UNSTRING WS-HOLD-COMMAND-LINE
            DELIMITED BY ' ' OR WS-HEX-ZERO
            INTO WS-UNSTRING-AREA
            COUNT IN WS-COUNT
```

```
            POINTER WS-POINTER
        END-UNSTRING.
        EVALUATE TRUE
    * PREV argument entered
            WHEN WS-COUNT = 4 AND
                WS-UNSTRING-AREA(1:4) = 'PREV'
                SET WS-88-FIND-PREV TO TRUE
                SET WS-88-SCROLL-FIND TO TRUE
            WHEN WS-COUNT < 6 AND
    * column argument entered
                WS-UNSTRING-AREA(1:WS-COUNT) IS NUMERIC
                COMPUTE WS-OFFSET = 6 - WS-COUNT
                MOVE WS-UNSTRING-AREA(1:WS-COUNT)
                                        TO
                           WS-FIND-COLUMN(WS-OFFSET:WS-COUNT)
            WHEN OTHER
                MOVE 'INVALID PARAMETER'
                                        TO  WS-ERROR-DISPLAY
                SET WS-88-DISPLAY-ERROR TO TRUE
                INITIALIZE  WS-FIND-STRING
                            WS-FIND-LENGTH
                            WS-FIND-COLUMN
                            WS-FIND-DIRECTION
                GO TO Ø64Ø-PARSE-FIND-EXIT
        END-EVALUATE.
    * we can only have a max of two arguments for the FIND command
        IF  WS-POINTER = WS-CMMND-LENGTH + 2
            SET WS-88-SCROLL-FIND TO TRUE
            GO TO Ø64Ø-PARSE-FIND-EXIT
        ELSE
            MOVE 'TOO MANY PARAMETERS ENTERED'
                                        TO  WS-ERROR-DISPLAY
            SET WS-88-DISPLAY-ERROR TO TRUE
            INITIALIZE  WS-FIND-STRING
                        WS-FIND-LENGTH
                        WS-FIND-COLUMN
                        WS-FIND-DIRECTION
            GO TO Ø64Ø-PARSE-FIND-EXIT
        END-IF.
     Ø64Ø-PARSE-FIND-EXIT.
        EXIT.
     Ø65Ø-PARSE-AMT.
    * right justify the amount argument entered
        COMPUTE WS-AMT-OFFSET = 7 - WS-CMMND-LENGTH.
        MOVE WS-HOLD-COMMAND-LINE(1:WS-CMMND-LENGTH)
                                        TO
                WS-SCROLL-AMT(WS-AMT-OFFSET:WS-CMMND-LENGTH).
        SET WS-88-SCROLL-AMT TO TRUE.
     Ø65Ø-PARSE-AMT-EXIT.
        EXIT.
```

```
Ø7ØØ-START-BROWSE.
    EXEC CICS
        STARTBR
            DATASET('DSERRLOG')
            RIDFLD(WS-ERRLOG-KEY)
            RBA
            EQUAL
            RESP(WS-RESP-CODE)
    END-EXEC.
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
            CONTINUE
        WHEN DFHRESP(NOTFND)
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'START BROWSE - RECORD NOT FOUND'
                                    TO  WS-MAP-MESSAGE
            MOVE 'Y'                TO  WS-PROCESS-FLAG
            GO TO Ø7ØØ-START-BROWSE-EXIT
        WHEN OTHER
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'START BROWSE - ERR, RESP: '
                                    TO  WS-MAP-MESSAGE
            MOVE WS-RESP-CODE       TO  WS-RESP-CODE-DISPLAY
            MOVE WS-RESP-CODE-DISPLAY
                                    TO  WS-MAP-MESSAGE(27:9)
            MOVE 'Y'                TO  WS-PROCESS-FLAG
            GO TO Ø7ØØ-START-BROWSE-EXIT
    END-EVALUATE.
Ø7ØØ-START-BROWSE-EXIT.
    EXIT.
Ø71Ø-READNEXT.
    MOVE LENGTH OF WS-LOG-RECORD  TO  WS-LOG-RECORD-LENGTH.
    EXEC CICS
        READNEXT
            DATASET('DSERRLOG')
            INTO(WS-LOG-RECORD)
            RIDFLD(WS-ERRLOG-KEY)
            RBA
            LENGTH(WS-LOG-RECORD-LENGTH)
            RESP(WS-RESP-CODE)
    END-EXEC.
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
            CONTINUE
        WHEN DFHRESP(NOTFND)
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'READNEXT - RECORD NOT FOUND'
```

```
                                TO  WS-MAP-MESSAGE
            MOVE 'Y'            TO  WS-PROCESS-FLAG
            GO TO Ø71Ø-READNEXT-EXIT
        WHEN DFHRESP(ENDFILE)
            IF  WS-READ-COUNT = Ø
                MOVE WS-ERRLOG-KEY
                              TO  WS-START-KEY
            END-IF
            MOVE 'END OF ERROR LOG' TO  WS-MAP-MESSAGE
            MOVE 'Y'            TO  WS-ERRLOG-FLAG
            MOVE 'N'            TO  WS-SCROLL-FLAG
            GO TO Ø71Ø-READNEXT-EXIT
        WHEN OTHER
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'READNEXT - ERR, RESP: '
                                TO  WS-MAP-MESSAGE
            MOVE WS-RESP-CODE   TO  WS-RESP-CODE-DISPLAY
            MOVE WS-RESP-CODE-DISPLAY
                                TO  WS-MAP-MESSAGE(23:9)
            MOVE 'Y'            TO  WS-PROCESS-FLAG
            GO TO Ø71Ø-READNEXT-EXIT
    END-EVALUATE.
Ø71Ø-READNEXT-EXIT.
    EXIT.
Ø72Ø-READPREV.
    MOVE LENGTH OF WS-LOG-RECORD  TO  WS-LOG-RECORD-LENGTH.
    EXEC CICS
        READPREV
            DATASET('DSERRLOG')
            INTO(WS-LOG-RECORD)
            RIDFLD(WS-ERRLOG-KEY)
            RBA
            LENGTH(WS-LOG-RECORD-LENGTH)
            RESP(WS-RESP-CODE)
    END-EXEC.
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
            CONTINUE
        WHEN DFHRESP(NOTFND)
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'READPREV - RECORD NOT FOUND'
                                TO  WS-MAP-MESSAGE
            MOVE 'Y'            TO  WS-PROCESS-FLAG
            GO TO Ø72Ø-READPREV-EXIT
        WHEN DFHRESP(ENDFILE)
            MOVE 'TOP OF ERROR LOG' TO  WS-MAP-MESSAGE
            MOVE 'Y'            TO  WS-ERRLOG-FLAG
            MOVE 'N'            TO  WS-SCROLL-FLAG
```

```
            GO TO Ø72Ø-READPREV-EXIT
        WHEN OTHER
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'READPREV - ERR, RESP: '
                                    TO  WS-MAP-MESSAGE
            MOVE WS-RESP-CODE       TO  WS-RESP-CODE-DISPLAY
            MOVE WS-RESP-CODE-DISPLAY
                                    TO  WS-MAP-MESSAGE(23:9)
            MOVE 'Y'                TO  WS-PROCESS-FLAG
            GO TO Ø72Ø-READPREV-EXIT
    END-EVALUATE.
Ø72Ø-READPREV-EXIT.
    EXIT.
Ø73Ø-END-BROWSE.
    EXEC CICS
        ENDBR
            DATASET('DSERRLOG')
            RESP(WS-RESP-CODE)
    END-EXEC.
    EVALUATE WS-RESP-CODE
        WHEN DFHRESP(NORMAL)
            CONTINUE
        WHEN OTHER
            PERFORM Ø82Ø-INITIALIZE-DISPLAY THRU
                    Ø82Ø-INITIALIZE-DISPLAY-EXIT
            MOVE 'END BROWSE - ERR, RESP: '
                                    TO  WS-MAP-MESSAGE
            MOVE WS-RESP-CODE       TO  WS-RESP-CODE-DISPLAY
            MOVE WS-RESP-CODE-DISPLAY
                                    TO  WS-MAP-MESSAGE(25:9)
            MOVE 'Y'                TO  WS-PROCESS-FLAG
            GO TO Ø73Ø-END-BROWSE-EXIT
    END-EVALUATE.
Ø73Ø-END-BROWSE-EXIT.
    EXIT.
Ø8ØØ-SEND-MAP.
* calculate the display position for the ruler on the display
    IF  WS-DISP < 1Ø1
        MOVE WS-DISP              TO  WS-REMAINDER
    ELSE
        DIVIDE WS-DISP BY 1ØØ GIVING WS-QUOTIENT
            REMAINDER WS-REMAINDER
    END-IF.
    COMPUTE WS-RULER-START = WS-REMAINDER.
    COMPUTE WS-RULER-START-LENGTH = 1Ø1 - WS-REMAINDER.
    IF  WS-RULER-START-LENGTH > 79
        MOVE 79                   TO  WS-RULER-START-LENGTH
    END-IF.
    COMPUTE WS-RULER-END-LENGTH = 79 - WS-RULER-START-LENGTH.
```

```
     MOVE WS-RULER-DISPLAY(WS-RULER-START:WS-RULER-START-LENGTH)
                                  TO
                          WS-MAP-RULER1(1:WS-RULER-START-LENGTH)
                          WS-MAP-RULER2(1:WS-RULER-START-LENGTH).
     COMPUTE WS-RULER-OFFSET = WS-RULER-START-LENGTH + 1.
     IF  WS-RULER-END-LENGTH > Ø
         MOVE WS-RULER-DISPLAY(1:WS-RULER-END-LENGTH)
                                  TO
               WS-MAP-RULER1(WS-RULER-OFFSET:WS-RULER-END-LENGTH)
               WS-MAP-RULER2(WS-RULER-OFFSET:WS-RULER-END-LENGTH)
     END-IF.
* if there was an error to display, put it in the lower ruler line
* and change it to reverse video
     IF  WS-88-DISPLAY-ERROR
         MOVE WS-ERROR-DISPLAY     TO  WS-MAP-RULER2
         MOVE '2'                  TO  WS-MAP-RULER2-H
     END-IF.
* convert the RBA from hex to display
     MOVE WS-START-KEY-X           TO  WS-HOLD-RBA.
     PERFORM Ø83Ø-CONVERT-RBA THRU
             Ø83Ø-CONVERT-RBA-EXIT.
     MOVE WS-DISPLAY-RBA           TO  WS-MAP-STRBA.
     MOVE WS-END-KEY-X             TO  WS-HOLD-RBA.
     PERFORM Ø83Ø-CONVERT-RBA THRU
             Ø83Ø-CONVERT-RBA-EXIT.
     MOVE WS-DISPLAY-RBA           TO  WS-MAP-ENDRBA.
     MOVE WS-DISP                  TO  WS-MAP-DISP.
     EXEC CICS
         SEND
             MAP('DSMØ9Ø2')
             ERASE
             FRSET
             FREEKB
             FROM(WS-DSMØ9Ø2-MAP)
     END-EXEC.
 Ø8ØØ-SEND-MAP-EXIT.
     EXIT.
 Ø81Ø-RECEIVE-MAP.
* we do a receive map in order to get anything entered on the
* command line
     EXEC CICS
         RECEIVE
             MAP('DSMØ9Ø2')
             INTO(WS-DSMØ9Ø2-MAP)
     END-EXEC.
 Ø81Ø-RECEIVE-MAP-EXIT.
     EXIT.
 Ø82Ø-INITIALIZE-DISPLAY.
     PERFORM VARYING WS-SUB
         FROM 1 BY 1
```

```
        UNTIL WS-SUB > 15
            MOVE SPACES TO WS-MAP-LINE(WS-SUB)
    END-PERFORM.
Ø82Ø-INITIALIZE-DISPLAY-EXIT.
    EXIT.
Ø83Ø-CONVERT-RBA.
    PERFORM VARYING WS-CSUB1 FROM 1 BY 1 UNTIL WS-CSUB1 > 4
        MOVE WS-HOLD-RBA(WS-CSUB1:1)
                                TO  WS-LEFT
        MOVE X'ØF'              TO  WS-RIGHT
        MOVE WS-CHAR2           TO  WS-CHAR3
        PERFORM VARYING WS-CSUB2 FROM 1 BY 1 UNTIL WS-CSUB2 > 2
            IF  WS-CHAR3(WS-CSUB2:1) > X'F9'
                MOVE X'ØØ'      TO  WS-LEFT
                MOVE WS-CHAR3(WS-CSUB2:1)
                                TO  WS-RIGHT
                SUBTRACT 57 FROM WS-BIN
                MOVE WS-RIGHT   TO  WS-CHAR3(WS-CSUB2:1)
            END-IF
        END-PERFORM
        MULTIPLY WS-CSUB1 BY 2
            GIVING WS-CSUB2
        SUBTRACT 1 FROM WS-CSUB2
        MOVE WS-CHAR3(1:2)      TO WS-DISPLAY-RBA(WS-CSUB2:2)
    END-PERFORM.
Ø83Ø-CONVERT-RBA-EXIT.
    EXIT.
Ø84Ø-CONVERT-LINE.
    MOVE WS-HOLD-BYTE           TO  WS-LEFT.
    MOVE X'ØF'                  TO  WS-RIGHT.
    MOVE WS-CHAR2               TO  WS-CHAR3.
    PERFORM VARYING WS-CSUB2 FROM 1 BY 1 UNTIL WS-CSUB2 > 2
        IF  WS-CHAR3(WS-CSUB2:1) > X'F9'
            MOVE X'ØØ'          TO  WS-LEFT
            MOVE WS-CHAR3(WS-CSUB2:1)
                                TO  WS-RIGHT
            SUBTRACT 57 FROM WS-BIN
            MOVE WS-RIGHT       TO  WS-CHAR3(WS-CSUB2:1)
        END-IF
    END-PERFORM.

    MULTIPLY WS-CSUB1 BY 2
        GIVING WS-CSUB2.
    SUBTRACT 1 FROM WS-CSUB2.
    MOVE WS-CHAR3(1:2)          TO WS-DISPLAY-BYTE.
Ø84Ø-CONVERT-LINE-EXIT.
    EXIT.
Ø9ØØ-RETURN.
    EXEC CICS
        RETURN
```

```
                TRANSID('DSBR')
                COMMAREA(WS-COMMAREA)
                LENGTH(81)
           END-EXEC.
       Ø9ØØ-RETURN-EXIT.
           EXIT.
```

Here is the MAP source:

```
          TITLE 'DSMØ9Ø2 '
          PRINT NOGEN
DSMØ9Ø2   DFHMSD TYPE=&SYSPARM,                                        X
                 TIOAPFX=YES,                                          X
                 CTRL=(FREEKB,FRSET),                                  X
                 EXTATT=YES,                                           X
                 LANG=COBOL,                                           X
                 MODE=INOUT,                                           X
                 TERM=327Ø
DSMØ9Ø2   DFHMDI SIZE=(24,8Ø),                                         X
                 LINE=Ø1,                                              X
                 COLUMN=Ø1
          DFHMDF POS=(Ø1,3Ø),                                          X
                 COLOR=RED,                                            X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=Ø21,                                           X
                 INITIAL='Aces Error Log Browse'
          DFHMDF POS=(Ø2,Ø1),                                          X
                 COLOR=RED,                                            X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=Ø1Ø,                                           X
                 INITIAL='Start Key:'
STRBA     DFHMDF POS=(Ø2,12),                                          X
                 COLOR=TURQUOISE,                                      X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=ØØ8
          DFHMDF POS=(Ø2,68),                                          X
                 COLOR=RED,                                            X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=ØØ5,                                           X
                 INITIAL='Disp:'
DISP      DFHMDF POS=(Ø2,74),                                          X
                 COLOR=TURQUOISE,                                      X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=ØØ5
          DFHMDF POS=(Ø3,Ø3),                                          X
                 COLOR=RED,                                            X
                 ATTRB=(PROT,BRT),                                     X
                 LENGTH=ØØ8,                                           X
                 INITIAL='End Key:'
ENDRBA    DFHMDF POS=(Ø3,12),                                          X
```

```
                COLOR=TURQUOISE,                                       X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=ØØ8
          DFHMDF POS=(Ø5,Ø1),                                          X
                COLOR=RED,                                             X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø12,                                            X
                INITIAL='Command ===>'
CMMND     DFHMDF POS=(Ø5,14),                                          X
                COLOR=TURQUOISE,                                       X
                ATTRB=(UNPROT,FSET,IC),                                X
                HILIGHT=UNDERLINE,                                     X
                LENGTH=Ø5Ø
          DFHMDF POS=(Ø5,65),                                          X
                ATTRB=(ASKIP,DRK),                                     X
                LENGTH=ØØ1
RULER1    DFHMDF POS=(Ø6,Ø1),                                          X
                COLOR=DEFAULT,                                         X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ1    DFHMDF POS=(Ø7,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ2    DFHMDF POS=(Ø8,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ3    DFHMDF POS=(Ø9,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ4    DFHMDF POS=(1Ø,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ5    DFHMDF POS=(11,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ6    DFHMDF POS=(12,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ7    DFHMDF POS=(13,Ø1),                                          X
                COLOR=BLUE,                                            X
                ATTRB=(PROT,BRT),                                      X
                LENGTH=Ø79
LINEØ8    DFHMDF POS=(14,Ø1),                                          X
                COLOR=BLUE,                                            X
```

```
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINEØ9    DFHMDF POS=(15,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE1Ø    DFHMDF POS=(16,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE11    DFHMDF POS=(17,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE12    DFHMDF POS=(18,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE13    DFHMDF POS=(19,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE14    DFHMDF POS=(2Ø,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
LINE15    DFHMDF POS=(21,Ø1),                                          X
               COLOR=BLUE,                                             X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
RULER2    DFHMDF POS=(22,Ø1),                                          X
               COLOR=DEFAULT,                                          X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø79
          DFHMDF POS=(23,Ø1),                                          X
               COLOR=DEFAULT,                                          X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø71,                                             X
               INITIAL='PF2 = HEX, PF5 = RFIND, PF7 = UP, PF8 = DOWN, PX
               F1Ø = LEFT, PF11 = RIGHT'
MESSAGE   DFHMDF POS=(24,Ø1),                                          X
               COLOR=YELLOW,                                           X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø5Ø
          DFHMDF POS=(24,53),                                          X
               COLOR=DEFAULT,                                          X
               ATTRB=(PROT,BRT),                                       X
               LENGTH=Ø19,                                             X
               INITIAL='Press CLEAR to Exit'
```

```
DFHMSD TYPE=FINAL
END
```

Henry Cable
Senior Developer
Ahold Information Services (USA)

CICS USER EXIT LIST with the transaction EXIT

With the transaction EXIT you are able to list all exits (GLUEs and TRUEs) in a CICS address space – see Figure 1. Unfortunately, the EXIT NAME(entry point) cannot be listed using regular API (SPI) functions. If you don't use the CWA, you must customize the application. Also, your own clean-up routine (instead of CSTCTUC) must be registered, but you can terminate the transaction normally. After defining in the PPT and PCT, you can

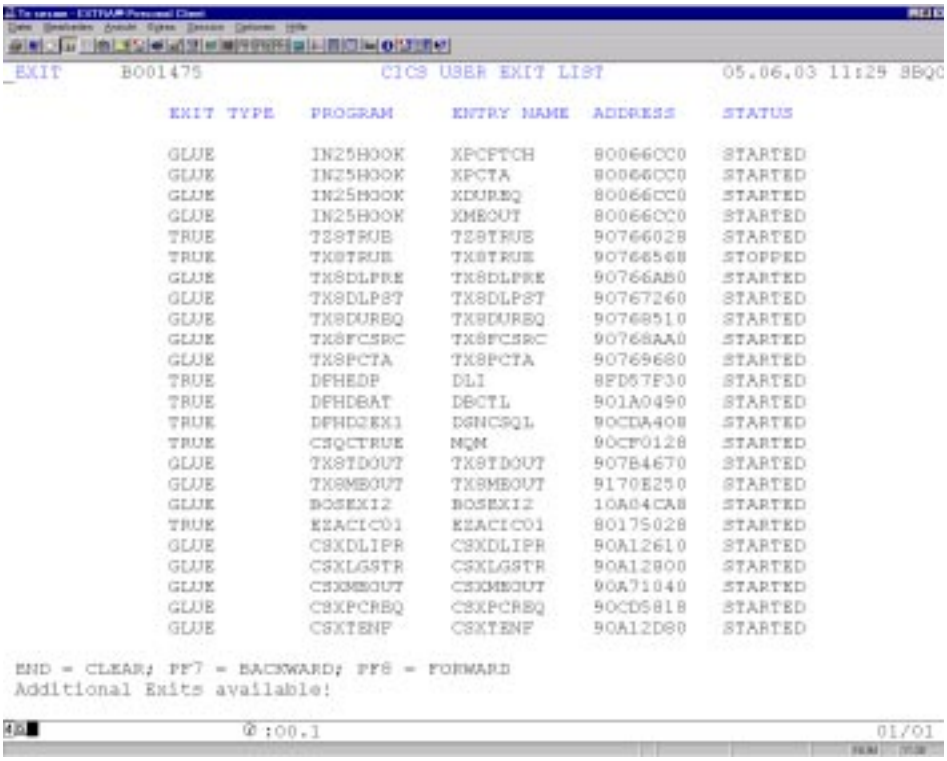

Figure 1: Exits list

run the EXIT transaction, provided that the program CSEXIT was compiled with the SP option.

CSEXIT

```
*ASM XOPTS(CICS,SP)
CSEXIT   TITLE '*************** CICS USER EXIT LIST ****************'
         SPACE
*-------------------------------------------------------------------*
*               C S E X I T                                         *
*-------------------------------------------------------------------*
* Note:      Program should be linked with AMODE=ANY / RMODE=ANY     *
* Comment:   This program is designed to list all active exits       *
*            GLUEs or TRUEs in a CICS on the screen and on           *
*            Temporary Storage.                                      *
*            Also the status "started" or "stopped" is listed.       *
*-------------------------------------------------------------------*
         SPACE
* ***************************************************************** *
* ***************************************************************** *
         EJECT
* ***************************************************************** *
*        D e f i n i t i o n s                                      *
* ***************************************************************** *
         SPACE
*      INCLUDE++ CSCWAA            INCLUDE CWA ASSEMBLER STRUCTURE
         EJECT
*-------------------------------------------------------------------
*        C I C S       C W A - B E R E I C H                      *
*        INCLUDE-ELEMENT  FUER ASM  PROGRAMME  CSCWAA             *
*-------------------------------------------------------------------
*        ADDRESSIERUNG:   EXEC CICS ADDRESS CWA(CWAPTR)           *
*        ACHTUNG      :   KEINE AENDERUNG ERLAUBT - NUR LESEN     *
*                         CWAPTR MUSS NOCH DEFINIERT WERDEN       *
*-------------------------------------------------------------------
               SPACE 3
               USING CWADSECT,CWAPTR
               SPACE 3
CWADSECT       DSECT
               SPACE 3
CWAAREA        DS   ØCL1536         CWA-BEREICH
               SPACE 1
CWAEYECA       DC    CL4' '         EYECATCHER
CWAVSYS        DC    CL4' '         ZUGEORDNETE VSAM-SYSID (CWACICID)
CWATSYS        DC    CL4' '         ZUGEORDNETE TERM-SYSID (CWACICID)
CWASYSID       DC    CL4' '         ORIGINAL SYSTEM-ID
CWAPPLID       DC    CL8' '         ORIGINAL APPLICATION-ID
CWANCVT        DC    AL4(Ø)         POINTER NLV-CVT
CWA_PTR_CUATR  DC    AL4(Ø)         ADRESSE D.CUA-TRANSAKTIONSTABELLE
```

```
CWA_CICSLEVEL   DC   ØCL4' '        CICS-LEVEL 'Ø311' OR 'Ø33Ø'
CWA_CICSLEV     DC    CL1' '        CICS-LEVEL
CWA_CICSVER     DC    CL1' '        CICS-VERSION
CWA_CICSREL     DC    CL1' '        CICS-RELEASE
CWA_CICSMOD     DC    CL1' '        CICS-MODIFICATION
CWA_CMFSTOP     DC    PL4'Ø'        STOP-TIME FOR CMF-EVENTS HHMMSSTC
                DS    XL56Ø         .........  FREI  .........
                DS    XL12Ø         .........  FREI  .........
                DS    XL22          .........  FREI  .........
CWACICTX        DC    CL4' '        CICS-ID-BESCHREIBUNG
CWACICID        DC    CL1' '        CICS-ID
CWA$PROD        EQU   C'P'           .. PROD
CWA$TEST        EQU   C'T'           .. TEST
CWA$VPRD        EQU   C'V'           .. VORPROD
CWA$SYST        EQU   C'S'           .. SYSTEM-CICS
CWACICNR        DC    CL1' '        CICS-NR
CWA$TERM        EQU   C'T'           .. TERMINAL
CWA$VSAM        EQU   C'V'           .. DATASET VSAM
CWA$PAIS        EQU   C'P'           .. PAISY
CWA$ODM         EQU   C'O'           .. ODM
CWA$INFO        EQU   C'I'           .. INFO
CWA$APPL        EQU   C'Ø'           .. APPLICATION ØØ-Ø9
*  $APPL        EQU   ????           .. APPLICATION A-C
*  $APPL        EQU   ????           .. APPLICATION E-O
*  $APPL        EQU   ????           .. APPLICATION Q-S
*  $APPL        EQU   ????           .. APPLICATION U-Z
CWADATUM        DC    CL8' '       DATUM FORMAT TT.MM.JJ
CWACTMJ         DC    CL6' '       DATUM TTMMJJ
CWAPTMJ         DC    PL4'Ø'       DATUM ØTTMMJJC
CWACJMT         DC    CL6' '       DATUM JJMMTT
CWAPJMT         DC    PL4'Ø'       DATUM ØJJMMTTC
CWACTMJ4        DC    CL8' '       DATUM TTMMJJJJ
CWAPTMJ4        DC    PL5'Ø'       DATUM ØTTMMJJJJC
CWACJ4MT        DC    CL8' '       DATUM JJJJMMTT
CWAPJ4MT        DC    PL5'Ø'       DATUM ØJJJJMMTTC
CWACMJ          DC    CL4' '       DATUM MMJJ
CWAPMJ          DC    PL3'Ø'       DATUM ØMMJJC
CWACJM          DC    CL4' '       DATUM JJMM
CWAPJM          DC    PL3'Ø'       DATUM ØJJMMC
CWACMJ4         DC    CL6' '       DATUM MMJJJJ
CWAPMJ4         DC    PL4'Ø'       DATUM ØMMJJJJC
CWACJ4M         DC    CL6' '       DATUM JJJJMM
CWAPJ4M         DC    PL4'Ø'       DATUM ØJJJJMMC
CWACT3J         DC    CL5' '       DATUM TTTJJ
CWAPT3J         DC    PL3'Ø'       DATUM TTTJJC
CWACJT3         DC    CL5' '       DATUM JJTTT
CWAPJT3         DC    PL3'Ø'       DATUM JJTTTC
CWACT3J4        DC    CL7' '       DATUM TTTJJJJ
CWAPT3J4        DC    PL4'Ø'       DATUM TTTJJJJC
CWACJ4T3        DC    CL7' '       DATUM JJJJTTT
```

```
CWAPJ4T3        DC    PL4'Ø'          DATUM JJJJTTTC
CWAZEIT         DC    CL5' '          UHRZEIT  SS:MM
*                                     ---------------------------
CWATABLE        DS    ØCL24           ,+Ø123456789-,Ø123456789
*                                     ---------------------------
CWATAB1         DS    ØCL13           TABELLE 1 /,/+/Ø123456789/-/
CWATAB2         DS    ØCL12           TABELLE 2 /,/+/Ø123456789/
CWACHKO1        DC     C','
CWATAB3         DS    ØCL12           TABELLE 3 /+/Ø123456789/-/
CWATAB4         DS    ØCL11           TABELLE 4 /+/Ø123456789/
CWACHARP        DC     C'+'
CWATAB5         DS    ØCL12           TABELLE 5 /Ø123456789/-/,/
CWATAB6         DS    ØCL11           TABELLE 6 /Ø123456789/-/
CWATAB7         DS    ØCL1Ø           TABELLE 7 /Ø123456789/
CWACHØ9         DC     C'Ø123456789'
CWACHARM        DC     C'-'
CWATAB8         DS    ØCL11           TABELLE 8 /,/Ø123456789/
CWACHKO2        DC     C','
CWACHØ92        DC     C'Ø123456789'
CWAZEITP        DC    PL4'Ø'          UHRZEIT  HHMMSSTC
CWADAY          DC    CL1Ø' '         WOCHENTAG
CWAMONTH        DC    CL9' '          MONAT
CWA_PTR_FTT     DC    AL4(Ø)          ADRESSE D. FUNKTIONSTASTENTABELLE
CWA_PTR_ANT     DC    AL4(Ø)          ADRESSE DER AKTIONSNAMENTABELLE
CWA_INFOCICS    DC    C' '            INFO-CICS IDENTIFIER
CWA_INFOCICS_Y  EQU   C'Y'            INFO-CICS IDENTIFIER -JA-
CWA_INFOCICS_N  EQU   C' '            INFO-CICS IDENTIFIER -NEIN-
CWA_DATUM_JJJJ  DC    CL1Ø' '         DATUM FORMAT TT.MM.JJJJ
                SPACE 1
CWAAREAE        EQU   *               ENDE CWA DEFINITIONEN
                SPACE 5
*-------------------------------------------------------------------
*         ENDE  DES  CICS   CWA_BEREICHES                          *
*-------------------------------------------------------------------
*         BEGINN DER DSECT FUER FUNKTIONSTASTENTABELLE              *
*         ADRESSIERUNG UEBER  "CWA_PTR_FTT"                        *
*-------------------------------------------------------------------
CWAFTTDSECT     DSECT
CWA_FTT_TASTE   DC    XL1'Ø'           TASTENIDENTIFIKATION
CWA_FTT_AKTION  DC    CL16' '          KURZBEZEICHNUNG DER TASTE
*                                      BSP. : HILFE
CWA_FTT_ANZEIGE DC    CL2Ø' '                 TEXT FUER DEN FUNKTIONS-
*                                             TASTENBLOCK EINES BILDES
*                                      BSP. : F1=HILFE
CWA_FTT_PFKEY   DC    CL4' '           PF-TASTE Z.B. "PF1 "
CWA_FTT_KURZTEXT DC   CL8' '           TASTENKUERZEL FUER POP-UP-MENUS
*                                      BSP. : F12=ABBR
CWA_FTT_TEXT    DS    CL2Ø7            BESCHREIBUNG DER AKTION
CWAFTTDSECTE    EQU   *
CWAFTTANZAHL    EQU   3Ø               ANZAHL TABELLENEINTRAEGE FTT
```

```
                SPACE 2
*-------------------------------------------------------------------
*         ENDE DER DSECT FUER FUNKTIONSTASTENTABELLE                 *
*-------------------------------------------------------------------
*         BEGINN DER DSECT FUER AKTIONSNAMENTABELLE                  *
*         ADRESSIERUNG UEBER  "CWA_PTR_ANT"                          *
*-------------------------------------------------------------------
```

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

Claus Reis
CICS Systems Programmer
Nuernberger Lebensversicherung AG (Germany)

Iona Technologies has announced a new family of products for Enterprise Application Integration (EAI). Iona's new Artix line allows large enterprise customers to renovate legacy applications and consolidate legacy middleware for a services-oriented architecture (SOA).

The Artix family enable companies to revamp any application, including those built on CICS and IMS, for a Services-Oriented Architecture (SOA).

The Artix family comprises four components. Artix Mainframe is a specialized version of Artix EnCompass that transforms CICS and IMS mainframe transactions into Web services. Artix EnCompass is software that can transform existing applications into Web services, which can then be distributed over an SOA. Artix Relay provides Web services links between CORBA, MQSeries, Tuxedo, Tibco, and other middleware products. Artix Migrate enables applications to bypass existing middleware platforms they've been hard-coded to interact with. This gives customers a relatively painless means of consolidating disparate EAI middleware.

For further information contact:
IONA Technologies, The IONA Building, Shelbourne Road, Ballsbridge, Dublin 4, Ireland.
Tel: (353) 1 637 2000.
URL: http://www.iona.com/products/artix/welcome.htm.

* * *

ObjectStar International has announced Release 4.0 of ObjectStar. the product allows for the creation, integration, and adaptation of composite applications in both complex and simple, heterogeneous, enterprise environments.

ObjectStar allows developers and system integrators to focus on the business need, rather than the technical problem. The new graphical workbench allows front-end and back-end developers to build, test, and deploy integrated applications.

At the heart of the Release 4.0 product set is the ObjectStar Integration Foundation, containing a table-driven MetaStor and a business rules engine, as well as administrative and security functions. It runs on z/OS, Solaris, Windows, and Linux.

ObjectStar Integration Gateways provide read and write access to both front-end and back-end environments. Front-end Integration Gateways provide mechanisms for Web-enabling legacy systems with little or no coding. ObjectStar Integration Gateway supports Java Connector Architecture (JCA), Enterprise Java Beans (EJB), Java Servlet 2.0 interface, and .NET architecture, as well as providing HTML and XML capabilities.

ObjectStar Database Gateways use native services to deliver access to a wide array of back-end systems, including DB2, IMS, CICS/DLI, VSAM, Model 204, and CA-IDMS, as well as most SQL databases.

For further information contact:
ObjectStar, Grove House, Lutyens Close, Chineham Court, Basingstoke, Hamps RG24 8AG, UK.
Tel: (08701) 624 930.
URL: http://www.objectstar.com/news/release4.htm.