195 CICS

February 2002

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 1998 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.

An introduction to CICS programming

The following text is aimed at people with little or no knowledge of CICS programming techniques. It is based on my own experience of the subject, and also in teaching programmers who had never heard of CICS before. However, some of the techniques I will explain below might also be useful for people already used to CICS.

The Customer Information Control Service – a big name that says too little – was born in the late sixties, when computer resources were scarce. Many of the characteristics of the product derive from the need to deal with hardware limitations, and to try to do the most with as little as possible. Although nowadays those limitations are not so important, CICS has kept many characteristics that somehow may contribute to a certain aura of being 'difficult' to program. Is this true? Yes and no. CICS programming certainly requires some practice and a few rules need to be obeyed, but the same goes for many other programming environments and programming languages. CICS is not more difficult to learn than other environments. In my opinion, a lot depends on the way things are explained. That is why I decided to write this text. In this article, I do not intend to teach CICS, but simply to share some ideas on the subject.

PSEUDO-CONVERSATIONAL PROGRAMMING

Pseudo-conversational programming is probably one of the first big words that a person has to hear when talking about CICS. My first CICS program was written as conversational, until someone told me that it had to be rewritten as pseudo-conversational. So what is it and why is it so important?

A CICS program is one that maintains a conversation with the user.

It sends a screen, the user inputs some information, the program receives that information, processes it, sends the user back some kind of answer, and this send-to-the-user and receive-from-the-user may go on indefinitely until the user orders the process to terminate, generally by pressing PF3.

A conversational program would simply be written as a loop between a send and a receive, something like:

Send – Receive – Process information – Go to Send…

But conversational programs, although possible, are not welcome in CICS.

Why? Let us consider that this program will be used by one hundred people at the same time. And let us not forget that the end user can be executing this program for a long time. The execution of the receive-process information-send, on the side of the program, is fast, but when the user gets a screen on his terminal he may be looking at it for minutes before pressing any key to answer back, or he may even get out of his chair and leave the program hanging there indefinitely. In practice, this means that the program will reside in CICS memory, in a state of inactivity, for as long as the end user decides. And if there are one hundred people using it, then there must be an equal number of copies of the program being executed.

So, what does a pseudo-conversational program do? It simply exits the code after sending a screen to the user, leaving CICS with an indication that when the user does some action at his terminal the same program (or another) should be called again. This 'leaving' with a planned return is called EXEC CICS RETURN TRANSID.

What are the advantages? One advantage is that a single copy of the executable code resides in memory, and that code can be executed by all the users accessing that program. Since the processing time of the program, between a receive and a send, is almost nil, all users can execute the same copy of the code. However, each user's data must be unique and preserved separately. This means that a CICS program has to be separated into two parts: the code and the data (in COBOL terms, the procedure division and the working storage).

So when a new user arrives on the system and starts executing that program, CICS simply creates a fresh copy of the working storage and assigns it to that user. When a program exits, CICS keeps track of the user's copy of the working storage, and when the user presses a key and starts re-executing the program, the program 'receives' a pointer

to the previous location of that working storage, in the program area known as LINKAGE SECTION, in a special item that has the mandatory name of DFHCOMMAREA.

Let us look now at a skeleton of a pseudo-conversational program:

```
001  ID DVISION.
002  WORKING-STORAGE SECTION.
003  77 ITEM_A…
004  01 ITEM_B…
005  01 WS-COMMAREA.
…
006  LINKAGE SECTION.
007  01  DFHCOMMAREA  PIC X(4000).
008  PROCEDURE DIVISION.
009      IF EIBCALEN = 0
010         PERFORM SEND-MAP
011         GO TO RETURN-TRANSID.
012      MOVE DFHCOMMAREA TO COMMAREA
013      PERFORM RECEIVE-MAP
014      IF EIBAID = DFHPF3
015         EXEC CICS RETURN.

    (process data)

016      PERFORM SEND-MAP
017      GO TO RETURN-TRANSID.
* * *
018  SEND-MAP.
019      EXEC CICS SEND MAP(MAP000).
020  RETURN-TRANSID.
021      EXEC CICS RETURN TRANSID(EIBTRNID)
022                       COMMAREA(WS-COMMAREA)
023                       LENGTH(500).
```

On line five, there is the working-storage COMMAREA. This is just the area that we need to be available between program executions. This is not all of the working-storage of the program, but just a part of it that we decided we need for the next program's execution. Typically, a COMMAREA should contain the map (MAP000) layout, that is, the copybook generated by BMS, possibly some record buffers if the program reads a file or a database, and a few other flags or variables. Let's suppose this area adds up to 500 bytes. Line seven is the counterpart of the COMMAREA. It is where the storage that was passed from the previous execution 'appears'. Note that I don't bother

to define it in any way, and I even don't care about the length: I defined a single global field with the mandatory name of DFHCOMMAREA, and with a length that must be greater than (or equal to) my original 500 bytes. (I could have mapped all the items here in the same way I defined them in the WS-COMMAREA, and with the same length, but that is not necessary at all, as I will explain below.)

Once I enter the procedure division, the first thing I do is to check whether the program is being executed for the first time or not. The CICS-provided EIBCALEN field is a convenient way to do this. If this field is zero, then this is a first-time execution, and, in that case, the only thing the program needs to do (in this small example, because a real program might do a little more) is to send a screen to the user for the first time and exit, returning to itself. This 'return', which is made in line 21, is a return to whatever transaction was associated with this program (once again, the CICS variable EIBTRNID, or 'transaction ID', holds that information for us), and specifies the area of the working storage that I need to be available for the next run, and how many bytes of it.

When the user performs some action on his newly-presented screen, the program starts executing for the second time, but now he receives a pointer to the previous WS-COMMAREA, and the information of how many bytes were passed. That length information (500) appears now in variable EIBCALEN. So we know this is not the first program's execution, and the course of action inside the procedure division will be different.

In theory, I could access all the COMMAREA received directly in the linkage section, by mapping the variables there, but that would pose two problems. I would need to use different names for the WS-COMMAREA variables and for the DFHCOMMAREA equivalents, because the program would work with the WS-COMMAREA variables in its first execution, and would work with the DFHCOMMAREA variables the remaining times. Furthermore, the RETURN TRANSID instruction, as it appears in the example, would be valid only for the first time, because all the other times the COMMAREA to return would be the DFHCOMMAREA itself, and so I would need to code a second return instruction.

All these problems disappear if I simply copy the data received in the linkage section to the new WS-COMMAREA. That is the first thing I do after checking that this is not the first run (or that EIBCALEN is not zero). With this solution, I don't need to map variables in DFHCOMMAREA, I just treat it as a block. And I don't care about the length, because in a MOVE operation the smaller field (the WS-COMMAREA) dictates the rule. And what is the purpose of having a DFHCOMMAREA greater than the actual COMMAREA used? None. But when you are developing a program, you don't know yet exactly what you need in the COMMAREA. You may well be adding variables and items to it until things are OK. So, instead of worrying about keeping an exact length every time you make a change, it is easier to leave it large enough to accommodate any size. This way, you need to ensure only that the 'length' in the return transid instruction corresponds to your work COMMAREA.

So now we are ready to receive the screen, or map, from the user, and check whether he decided to terminate the program. If not, then we do some processing based on the user's input (read a record, perhaps), send the result back to him, and return to CICS once more.

With this pseudo-conversational approach, each user in the system requires space for his copy of the working storage, so 100 users will need about 100 times that space, plus a copy of the executable code shared by all. If the program was conversational, each user would require a full copy of the program, which can easily be several times larger, if the executable code is large. And with a thousand users in the system and a lot of programs running, the savings in memory that the pseudo-conversational approach allows become even more important.

But memory-saving is not the only advantage. Another decisive factor is the overhead of loading a program. A conversational approach means that for each user a new copy of the program must be loaded and put into execution. This is a 'heavy' operation, in terms of load on both on disk and CPU. With the pseudo approach, only one load operation is necessary.

MULTI-PROGRAMMING APPLICATIONS

A CICS application written with a single program is feasible for simple situations, but for more complex ones it can be preferable to use several programs that return or that pass control to each other.

There are several ways to set up an application with several programs, but I will mention just one in detail, which I think is the simplest to start with.

With this method, each program has its own associated transaction, and, when a program wants to pass on control to another, it simply returns to that program's transaction, instead of returning to its own. In this form, the program that is leaving sends the user a screen before doing so, and the program that starts execution receives that screen. This means that between the two programs' execution there is a wait for the user action.

Practical applications? There are many. I will illustrate things with a small example. Suppose you have an application where PROGRAM0 sends a SCREEN0 with a menu of choices. According to the choice, this program 'calls' PROGRAM1, PROGRAM2, etc, each one with its own screen(s). For example, the choices can be to create, view, modify, print, or delete a record, and each program performs one of these functions. How do I set things up? There are many ways to do it, but the way I would do it is as follows.

First, I would define a COMMAREA that would work for all the programs, containing all the necessary record buffers and screen copybooks. This COMMAREA would be created in PROGRAM0 only; all the others would access it. I would set up things as illustrated in Figure 1. I show only two programs there, the base program (Program0, with screen Scr0 and transaction Trn0) and one of the option programs (Program1, Scr1, Trn1). All the other options would be similar to Program1. The Figure is intended to illustrate the relationships between these programs.

Program0 is very similar to the single program example I gave earlier, except for the part where it passes control to the other programs. Observe that the physical COMMAREA is always created in Program0,

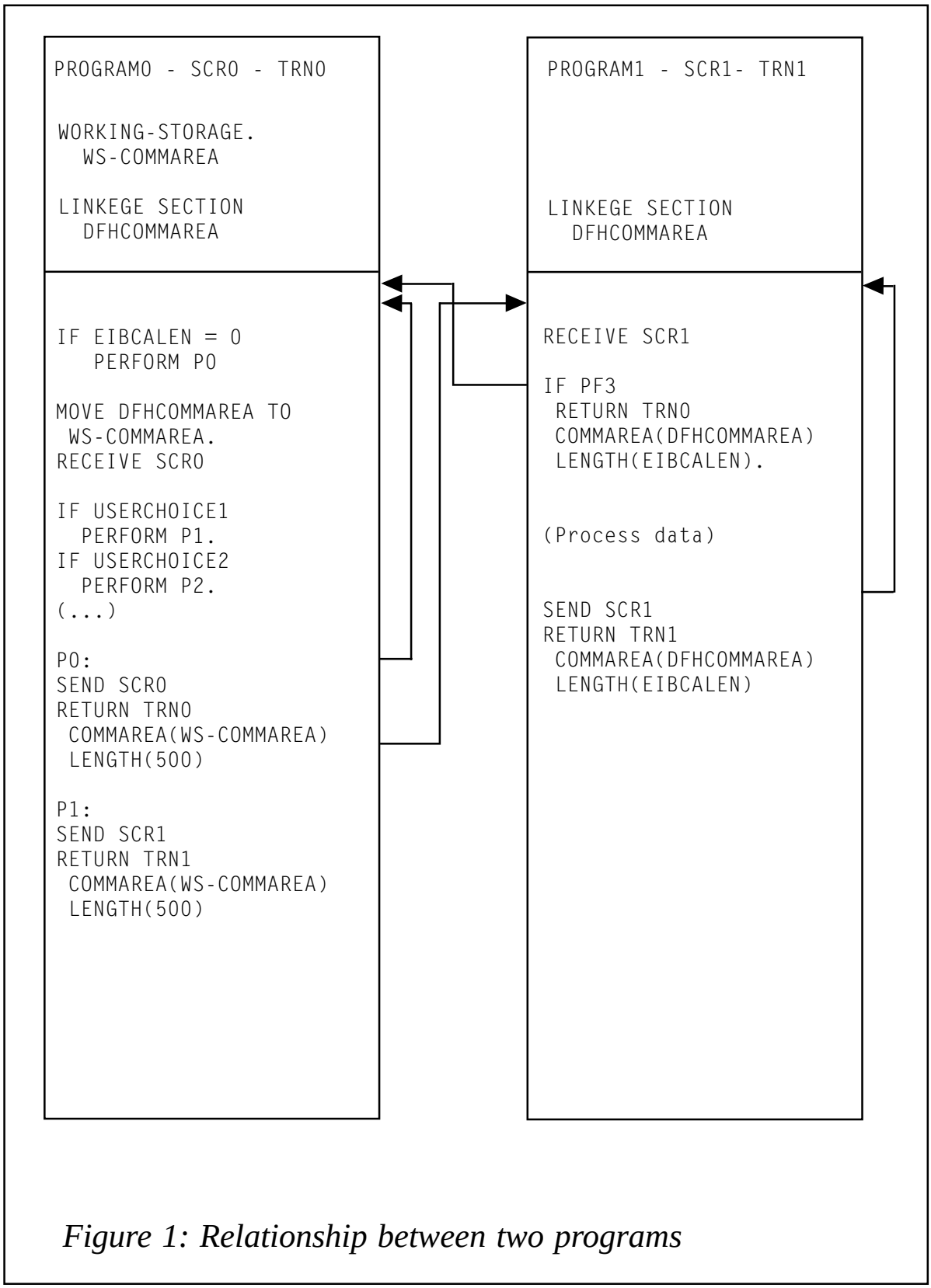

Figure 1: Relationship between two programs

and that Program1 always accesses it. For that reason, all the pseudo-conversational loops of Program1, when it returns to its own transaction ID (Trn1), refer to DFHCOMMAREA, and not to any working storage of that program. So, there is no move operation from that area to a working storage area, as in the base program. The EIBCALEN inquiry is also not necessary, and even if it was, it would be useless, because Program1 is always called with a COMMAREA passed to it, and so EIBCALEN is never zero.

Note that I send Scr1 before leaving Program0, and that Program1 always receives Scr1. When this program wants to exit, by means of PF3, it does so to Program0, after sending Scr0.

This method has the inconvenience that it needs a transaction associated with each program, something that might not be needed with other methods, for example by means of EXEC CICS XCTL PROGRAM. An easy way to avoid multiple transactions is to use a 'driver' program. A driver program is a generic program that has a generic transaction associated with it, and a large enough linkage section that does nothing but XCTL to a program, whose name is indicated in a fixed position inside a COMMAREA (preferably at the beginning of the COMMAREA). This method has the advantage that the program and its transaction can be used by any application, not just by one. So, when a program wants to call another, it simply puts that program's name in the pre-defined position in the COMMAREA and returns to the generic transaction with that COMMAREA and the correct length.

When the user starts some action at his terminal, the driver program begins execution and immediately passes control (XCTL) to the intended program, handing him the COMMAREA with the length declared as EIBCALEN.

There are, of course, many other methods for CICS programming, and with time and experience each person can develop his own way of doing things. As I said before, I just wanted to share a few ideas on the subject, not cover all the possibilities

But for those new to CICS programming, I hope I have explained in this article enough for it to be a useful starting point.

Systems Programmer
(Portugal)

Contributing to *CICS Update*

In addition to *CICS Update*, the Xephon family of *Update* publications now includes *MVS Update*, *VSAM Update*, *DB2 Update*, *RACF Update*, *AIX Update*, *MQ Update*, *NT Update*, *TCP/SNA Update*, *Oracle Update*, and *TSO/ISPF Update*. Although the articles published are of a very high standard, the vast majority are not written by professional writers, and we rely heavily on our readers themselves taking the time and trouble to share their experiences with others. Many have discovered that writing an article is not the daunting task that it might appear to be at first glance.

They have found that the effort needed to pass on valuable information to others is more than offset by our generous terms and conditions and the recognition they gain from their fellow professionals. Often, just a few hundred words are sufficient to describe a problem and the steps taken to solve it.

If you have ever experienced any difficulties, or made an interesting discovery, you could receive a cash payment, a free subscription to any of our *Updates*, or a credit against any of Xephon's wide range of products and services, simply by telling us all about it.

More information about contributing an article to a Xephon Update, and an explanation of the terms and conditions under which we publish articles, can be found at www.xephon.com/nfc. Articles can be sent to the editor, Trevor Eddolls, at any of the addresses shown on page 2, or e-mail him at trevore@xephon.com.

Ensuring absolutely trouble-free CICS operation

To ensure trouble-free CICS operation in the future, it is very important to demand certain standards from all the active CICS applications.

The following criteria should cover everything. They are intended to make it possible for the developers to use any technique (for example MRO, dynamic routing, etc) when using someone else's products and, above all, produce a trouble-free CICS environment.

If these criteria are always fulfilled, a great deal of time and money will be saved in the development and maintenance of the software, and technical problems will be almost unheard of.

The criteria that apply to software products, and which should find usage in the CICS arena, are:

1 MRO/SYSPLEX ability – all transactions of a product must be able to use dynamic routing.

2 31-bit ability – all modules must be linked with AMODE (31) and RMODE (ANY).

3 The operation in a pure LE/370 environment must be supported.

4 Techniques like Subsystem Storage Protection and Transaction Isolation must be supported.

5 The products must be InterTest-compatible. This means that all programs should work under constant InterTest control. Note: C programs aren't (yet?) supported.

6 All programs must work exclusively with standard APIs.

7 All programs must work in the USER-Key.

8 The naming conventions for all resources (programs, transactions, files, etc) must be consistent and, where necessary, they must be able to adjust to the demands of the enterprise.

9 No modifications to the CICS code must be necessary for an application to run.

10 Must not use Exits (eg GLUEs/TRUEs).

11 The names for Temporary Storage Queues must be documented and correspond to the Nlv standard (for Termid and Transaction name).

12 No working with URMs (eg ZNEP, PEP, Autoinstall).

13 Terminal addresses must be independent and fixed (eg Termid, Netname).

14 Must use transaction-oriented programming.

15 Must link modules with RENT and REUSE.

Claus Reis
CICS Systems Programmer
Nuernberger Lebensversicherung AG (Germany)

Named Counter Server

Especially with an online environment (CICS), some applications need to get a sequence number which is unique over a Parallel Sysplex environment. Most solutions to this problem need to create an affinity for a CICSPlex or they result in bad performance. But with the Named Counter Server enhancement, which is a new facility that came with CICS TS V1.3, you can get a sequence number that is unique over a Parallel Sysplex environment without an affinity and with good performance.

To use a named counter, you should start a Named Counter Server address space, and define a pool for the Coupling Facility. To use a named counter from a CICS region you don't need to do anything special to access the Named Counter Server except use the appropriate CICS commands. But if you want to use a named counter from a batch program, you should add the SDFHLOAD dataset and the dataset that contains the Named Counter Server Options Table (DFHNCOPT) to the STEPLIB of the JCL.

JCL to define a Named Counter Pool (POOL1):

```
//N3560DX JOB (IBMUSER,),'PGMRNAME',
// CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),NOTIFY=&SYSUID
//*
//IBMUSER  EXEC PGM=IXCMIAPU
//STEPLIB  DD DSN=SYS1.MIGLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
  DATA TYPE(CFRM) REPORT(YES)
  DELETE POLICY NAME(NAMEDPL1)
  /* DEFINE POLICY NAME(POLICY1) REPLACE(YES) */
  DEFINE POLICY NAME(NAMEDPL1)
     CF NAME(ICMFR65)
            TYPE(009672)
            MFG(IBM)
            PLANT(51)
            SEQUENCE(000000067809)
            PARTITION(2)
            CPCID(00)
            DUMPSPACE(2400)
```

```
     STRUCTURE NAME(DFHNCLS_POOL1)
            SIZE(512)
            INITSIZE(256)
            PREFLIST(ICMFR65)
/*
```

Named Counter Server address space procedure:

```
//NCSERVER PROC
//NCSERVER EXEC PGM=DFHNCMN,REGION=32M,TIME=144Ø
//STEPLIB  DD   DSN=CICSTS13.CICS.SDFHAUTH,DISP=SHR
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   DSN=CICS.PARMLIB.TS13(NCPARM),DISP=SHR
```

Parameter member for Named Counter Server (CICS.PARMLIB.TS13(NCPARM)):

```
POOLNAME=POOL1
```

SAMPLE CICS PROGRAM THAT USES A NAMED COUNTER

```
      ***************************************************************
      *                                                             *
      * BU PROGRAM, ONLINE OLARAK, COUPLING FACILITY'               *
      *             OLAN POOL(POOL1) ZERINDE COUNTER(COUNTER1)    *
      *                                                             *
      ***************************************************************
       IDENTIFICATION DIVISION.
       PROGRAM-ID. NCSERVON.
       AUTHOR.          ERHAN PASA
       DATE-WRITTEN.    31/Ø7/2ØØ1.
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  IBM-Z67.
       OBJECT-COMPUTER.  IBM-Z67.
       DATA DIVISION.
       FILE SECTION.
       WORKING-STORAGE SECTION.
      ************* DEFINE SAPI INPUT/OUTPUT PARAMETERS ************
       Ø1  MSG-TEXT                    PIC X(7Ø).
       Ø1  MSG-RETURN.
           Ø5 RESP-TXTS.
              1Ø RESP-TXT                 PIC X(15)
                 VALUE '*** EIBRESP  = '.
              1Ø EIBRESP-CODE             PIC 9(Ø8).
           Ø5 RESP2-TXTS.
              1Ø RESP2-TXT                PIC X(15)
```

```
           VALUE '*** EIBRESP2 = '.
        1Ø EIBRESP2-CODE            PIC 9(Ø8).
 Ø1  NCTR-REC.
     Ø5  NC-COUNTER-NAME         PIC X(16).
     Ø5  NC-POOL-SELECTOR        PIC X(8).
     Ø5  NC-CURRENT-VALUE        PIC S9(8) COMP.
     Ø5  NC-MIN-VALUE            PIC S9(8) COMP.
     Ø5  NC-MAX-VALUE            PIC S9(8) COMP.
 Ø1  NCTR-DISP-REC.
     Ø5 COUNTER-NAME.
        1Ø COUNTER-NAME-TXT        PIC X(2Ø)
           VALUE '*** COUNTER NAME  = '.
        1Ø NC-CNTR-NAME-DISP       PIC X(16).
     Ø5 POOL-SELECTOR.
        1Ø POOL-SELECTOR-TXT       PIC X(2Ø)
           VALUE '*** POOL NAME     = '.
        1Ø NC-POOL-SEL-DISP        PIC X(8).
     Ø5 CURRENT-VALUE.
        1Ø CURRENT-VALUE-TXT       PIC X(2Ø)
           VALUE '*** CURRENT VALUE = '.
        1Ø NC-CURR-VAL-DISP        PIC 9(8).
     Ø5 MIN-VALUE.
        1Ø MIN-VALUE-TXT           PIC X(2Ø)
           VALUE '*** MINIMUM VALUE = '.
        1Ø NC-MIN-VAL-DISP         PIC 9(8).
     Ø5 MAX-VALUE.
        1Ø MAX-VALUE-TXT           PIC X(2Ø)
           VALUE '*** MAXIMUM VALUE = '.
        1Ø NC-MAX-VAL-DISP         PIC 9(8).
*
*
 PROCEDURE DIVISION.
 MAIN-RTN.
*
*************  BEGIN - DEFINE ***********************
*
* POOL(POOL1)'DE COUNTER(COUNTER1)'IN .
*
*
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE +3           TO  NC-CURRENT-VALUE.
        MOVE +Ø           TO  NC-MIN-VALUE.
        MOVE +1Ø          TO  NC-MAX-VALUE.
*
        EXEC CICS
             DEFINE COUNTER(NC-COUNTER-NAME)
                    POOL(NC-POOL-SELECTOR)
```

```
                    VALUE(NC-CURRENT-VALUE)
                    MINIMUM(NC-MIN-VALUE)
                    MAXIMUM(NC-MAX-VALUE)
           END-EXEC.
*
           IF EIBRESP NOT = Ø THEN
              MOVE '*** NCTR-DEFINE BASARISIZ ***' TO MSG-TEXT
              EXEC CICS
                  SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
                  JUSTIFY(1)
              END-EXEC
              MOVE EIBRESP   TO EIBRESP-CODE
              MOVE EIBRESP2  TO EIBRESP2-CODE
              EXEC CICS
                  SEND TEXT FROM(MSG-RETURN) LENGTH(46)
                  JUSTIFY(2)
              END-EXEC
           ELSE
              MOVE '*** NCTR-DEFINE BASARILI  ***' TO MSG-TEXT
              EXEC CICS
                  SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
                  JUSTIFY(1)
              END-EXEC.
*
*************  END - DEFINE *************************
*************  BEGIN - GET **********************
*
* POOL(POOL1)'DEKI COUNTER(COUNTER1)'IN .
*
*
           MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
           MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
*
           EXEC CICS
                GET COUNTER(NC-COUNTER-NAME)
                    POOL(NC-POOL-SELECTOR)
                    VALUE(NC-CURRENT-VALUE)
                    WRAP
           END-EXEC.
*
           IF EIBRESP NOT = Ø THEN
              MOVE '*** NCTR-GET BASARISIZ ***' TO MSG-TEXT
              EXEC CICS
                  SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
                  JUSTIFY(5)
              END-EXEC
              MOVE EIBRESP   TO EIBRESP-CODE
              MOVE EIBRESP2  TO EIBRESP2-CODE
```

```
           EXEC CICS
               SEND TEXT FROM(MSG-RETURN) LENGTH(46)
               JUSTIFY(6)
           END-EXEC
        ELSE
           MOVE '*** NCTR-GET BASARILI  ***' TO MSG-TEXT
           EXEC CICS
               SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
               JUSTIFY(5)
           END-EXEC
           MOVE  NC-CURRENT-VALUE    TO  NC-CURR-VAL-DISP
           EXEC CICS
               SEND TEXT FROM(CURRENT-VALUE) LENGTH(28)
               JUSTIFY(6)
           END-EXEC.
*
*************  END - GET ************************
*************  BEGIN - QUERY ***********************
*
* POOL(POOL1)'DEKI COUNTER(COUNTER1)'E
*
*
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
*
        EXEC CICS
             QUERY  COUNTER(NC-COUNTER-NAME)
                    POOL(NC-POOL-SELECTOR)
                    VALUE(NC-CURRENT-VALUE)
                    MINIMUM(NC-MIN-VALUE)
                    MAXIMUM(NC-MAX-VALUE)
        END-EXEC.
*
        IF EIBRESP NOT = Ø THEN
           MOVE '*** NCTR-QUERY  BASARISIZ ***' TO MSG-TEXT
           EXEC CICS
               SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
               JUSTIFY(11)
           END-EXEC
           MOVE EIBRESP   TO EIBRESP-CODE
           MOVE EIBRESP2  TO EIBRESP2-CODE
           EXEC CICS
               SEND TEXT FROM(MSG-RETURN) LENGTH(46)
               JUSTIFY(12)
           END-EXEC
        ELSE
           MOVE '*** NCTR-QUERY  BASARILI  ***' TO MSG-TEXT
           EXEC CICS
```

```
               SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
               JUSTIFY(11)
            END-EXEC
            MOVE  NC-COUNTER-NAME     TO  NC-CNTR-NAME-DISP
            MOVE  NC-POOL-SELECTOR    TO  NC-POOL-SEL-DISP
            MOVE  NC-CURRENT-VALUE    TO  NC-CURR-VAL-DISP
            MOVE  NC-MIN-VALUE        TO  NC-MIN-VAL-DISP
            MOVE  NC-MAX-VALUE        TO  NC-MAX-VAL-DISP
            EXEC CICS
               SEND TEXT FROM(COUNTER-NAME) LENGTH(36)
               JUSTIFY(12)
            END-EXEC.
            EXEC CICS
               SEND TEXT FROM(POOL-SELECTOR) LENGTH(28)
               JUSTIFY(13)
            END-EXEC.
            EXEC CICS
               SEND TEXT FROM(CURRENT-VALUE) LENGTH(28)
               JUSTIFY(14)
            END-EXEC.
            EXEC CICS
               SEND TEXT FROM(MIN-VALUE) LENGTH(28)
               JUSTIFY(15)
            END-EXEC.
            EXEC CICS
               SEND TEXT FROM(MAX-VALUE) LENGTH(28)
               JUSTIFY(16)
            END-EXEC.
*
*************  END - QUERY ************************
*************  BEGIN - DELETE ***********************
*
* POOL(POOL1)'DEKI COUNTER(COUNTER1)'IN SILINMESI
*
*
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
*
        EXEC CICS
             DELETE COUNTER(NC-COUNTER-NAME)
                    POOL(NC-POOL-SELECTOR)
        END-EXEC.
*
        IF EIBRESP NOT = Ø THEN
           MOVE '*** NCTR-DELETE BASARISIZ ***' TO MSG-TEXT
           EXEC CICS
               SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
               JUSTIFY(21)
```

```
            END-EXEC
            MOVE EIBRESP   TO EIBRESP-CODE
            MOVE EIBRESP2  TO EIBRESP2-CODE
            EXEC CICS
                SEND TEXT FROM(MSG-RETURN) LENGTH(46)
                JUSTIFY(22)
            END-EXEC
         ELSE
            MOVE '*** NCTR-DELETE BASARILI  ***' TO MSG-TEXT
            EXEC CICS
                SEND TEXT FROM(MSG-TEXT) LENGTH(7Ø)
                JUSTIFY(21)
            END-EXEC.
*
*************  END - DELETE ************************
     EXEC CICS
          SEND PAGE
     END-EXEC.
     STOP RUN.
```

SAMPLE BATCH PROGRAM THAT USES A NAMED COUNTER

```
*****************************************************************
* THIS PROGRAM RUNS IN BATCH. IT DEFINES A COUNTER NAMED     *
* AS "COUNTER1" IN THE POOL(POOL1) WHICH IS DEFINED ON THE   *
* COUPLING FACILITY. THEN IT QUERIES THE COUNTER, GETS THE   *
* CURRENT VALUE OF THE COUNTER, AGAIN QUERIES THE COUNTER    *
* AND IN THE END IT DELETES THE COUNTER. AFTER EACH PROCESS, *
* SOME MESSAGES ARE DISPLAYED GIVING INFORMATION ABOUT THE   *
* COUNTER.                                                   *
*                                                            *
*****************************************************************
 IDENTIFICATION DIVISION.
 PROGRAM-ID. NCSERVBA.
 AUTHOR.          ERHAN PASA
 DATE-WRITTEN.    26/Ø7/2ØØ1.
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SOURCE-COMPUTER.  IBM-Z67.
 OBJECT-COMPUTER.  IBM-Z67.
 DATA DIVISION.
 FILE SECTION.
 WORKING-STORAGE SECTION.
************* DEFINE SAPI INPUT/OUTPUT PARAMETERS ************
 Ø1  NCTR-REC.
     Ø5  NC-FUNCTION             PIC S9(8)   COMP.
     Ø5  NC-RETURN-CODE          PIC S9(8)   COMP.
```

```
     Ø5  NC-POOL-SELECTOR        PIC X(8).
     Ø5  NC-COUNTER-NAME         PIC X(16).
     Ø5  NC-VALUE-LENGTH         PIC S9(8)   COMP VALUE +4.
     Ø5  NC-CURRENT-VALUE        PIC S9(8)   COMP.
     Ø5  NC-MIN-VALUE            PIC S9(8)   COMP.
     Ø5  NC-MAX-VALUE            PIC S9(8)   COMP.
     Ø5  NC-OPTIONS              PIC S9(8)   COMP VALUE +Ø.
     Ø5  NC-UPDATE-VALUE         PIC S9(8)   COMP VALUE +1.
     Ø5  NC-COMP-MIN             PIC S9(8)   COMP VALUE +Ø.
     Ø5  NC-COMP-MAX             PIC S9(8)   COMP VALUE +Ø.
 Ø1  NCTR-DISP-REC.
     Ø5  NC-CURR-VAL-DISP        PIC 9(8).
     Ø5  NC-MIN-VAL-DISP         PIC 9(8).
     Ø5  NC-MAX-VAL-DISP         PIC 9(8).
     Ø5  NC-OPTIONS-DISP         PIC 9(8).
*
* THE COPYBOOK WHICH CONTAINS THE INTERFACE DEFINITIONS.
* IT EXISTS IN THE DATASET "CICSTS13.CICS.SDFHCOB(DFHNCCOB)"
 COPY DFHNCCOB.
*
 LINKAGE SECTION.
 Ø1  NULL-PTR         USAGE IS POINTER.
*
 PROCEDURE DIVISION.
 MAIN-RTN.
*
        SET ADDRESS OF NULL-PTR TO NULLS.
*
* IN THE USAGE OF NAMED COUNTER SERVER CALLABLE INTERFACE,
* FOR THE UNUSED PARAMETERS, "NULL-PTR" IS GIVEN.
*
*************  BEGIN - CREATE ***********************
*
* DEFINING COUNTER(COUNTER1) IN POOL(POOL1)
*
*
        MOVE NC-CREATE    TO  NC-FUNCTION.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE +4           TO  NC-VALUE-LENGTH.
        MOVE +3           TO  NC-CURRENT-VALUE.
        MOVE +Ø           TO  NC-MIN-VALUE.
        MOVE +1Ø          TO  NC-MAX-VALUE.
        MOVE NC-WRAP      TO  NC-OPTIONS.
        MOVE +1           TO  NC-UPDATE-VALUE.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
```

```
                              NC-POOL-SELECTOR
                              NC-COUNTER-NAME
                              NC-VALUE-LENGTH
                              NC-CURRENT-VALUE
                              NC-MIN-VALUE
                              NC-MAX-VALUE
                              NC-OPTIONS
                              NC-UPDATE-VALUE
                              NULL-PTR
                              NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-CREATE FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
           DISPLAY '*** NCTR-CREATE SUCCESSFUL. ***'.
*
*************  END - CREATE ************************
*************  BEGIN - INQUIRE **********************
*
* QUERYING COUNTER(COUNTER1) WHICH IS IN POOL(POOL1)
* THEN, THE INFORMATION ABOUT THE COUNTER IS DISPLAYED.
*
*
        MOVE NC-INQUIRE   TO  NC-FUNCTION.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE +4           TO  NC-VALUE-LENGTH.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
                             NC-POOL-SELECTOR
                             NC-COUNTER-NAME
                             NC-VALUE-LENGTH
                             NC-CURRENT-VALUE
                             NC-MIN-VALUE
                             NC-MAX-VALUE
                             NC-OPTIONS
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-INQUIRE FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
           MOVE  NC-CURRENT-VALUE    TO  NC-CURR-VAL-DISP
           MOVE  NC-MIN-VALUE        TO  NC-MIN-VAL-DISP
```

```
           MOVE  NC-MAX-VALUE        TO  NC-MAX-VAL-DISP
           MOVE  NC-OPTIONS          TO  NC-OPTIONS-DISP
           DISPLAY '*** NCTR-INQUIRE SUCCESSFUL. ***'
           DISPLAY '*** NC-CURRENT-VALUE ==> ' NC-CURR-VAL-DISP
           DISPLAY '*** NC-MIN-VALUE     ==> ' NC-MIN-VAL-DISP
           DISPLAY '*** NC-MAX-VALUE     ==> ' NC-MAX-VAL-DISP
           DISPLAY '*** NC-OPTIONS       ==> ' NC-OPTIONS-DISP.
*
*************  END - INQUIRE ************************
*************  BEGIN - ASSIGN ***********************
*
* GETTING THE CURRENT VALUE OF COUNTER(COUNTER1) WHICH IS IN
* POOL(POOL1). THEN, THE VALUE IS DISPLAYED.
* AFTER THIS OPERATION, THE COUNTER AUTOMATICALLY GETS
* ITS NEW VALUE.
*
*
        MOVE NC-ASSIGN    TO  NC-FUNCTION.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE +4           TO  NC-VALUE-LENGTH.
        MOVE +2           TO  NC-UPDATE-VALUE.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
                             NC-POOL-SELECTOR
                             NC-COUNTER-NAME
                             NC-VALUE-LENGTH
                             NC-CURRENT-VALUE
                             NC-MIN-VALUE
                             NC-MAX-VALUE
                             NC-OPTIONS
                             NC-UPDATE-VALUE
                             NULL-PTR
                             NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-ASSIGN FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
           DISPLAY '*** NCTR-ASSIGN SUCCESSFUL. ***'
           MOVE  NC-CURRENT-VALUE    TO  NC-CURR-VAL-DISP
           MOVE  NC-MIN-VALUE        TO  NC-MIN-VAL-DISP
           MOVE  NC-MAX-VALUE        TO  NC-MAX-VAL-DISP
           MOVE  NC-OPTIONS          TO  NC-OPTIONS-DISP
           DISPLAY '*** NC-CURRENT-VALUE ==> ' NC-CURR-VAL-DISP
           DISPLAY '*** NC-MIN-VALUE     ==> ' NC-MIN-VAL-DISP
           DISPLAY '*** NC-MAX-VALUE     ==> ' NC-MAX-VAL-DISP
```

```
           DISPLAY '*** NC-OPTIONS       ==> ' NC-OPTIONS-DISP.
*
*************  END - ASSIGN ************************
*************  BEGIN - INQUIRE **********************
*
* QUERYING COUNTER(COUNTER1) WHICH IS IN POOL(POOL1)
* THEN, THE INFORMATION ABOUT THE COUNTER IS DISPLAYED.
*
*
        MOVE NC-INQUIRE   TO  NC-FUNCTION.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
        MOVE +4           TO  NC-VALUE-LENGTH.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
                             NC-POOL-SELECTOR
                             NC-COUNTER-NAME
                             NC-VALUE-LENGTH
                             NC-CURRENT-VALUE
                             NC-MIN-VALUE
                             NC-MAX-VALUE
                             NC-OPTIONS
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-INQUIRE FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
           MOVE  NC-CURRENT-VALUE    TO  NC-CURR-VAL-DISP
           MOVE  NC-MIN-VALUE        TO  NC-MIN-VAL-DISP
           MOVE  NC-MAX-VALUE        TO  NC-MAX-VAL-DISP
           MOVE  NC-OPTIONS          TO  NC-OPTIONS-DISP
           DISPLAY '*** NCTR-INQUIRE SUCCESSFUL. ***'
           DISPLAY '*** NC-CURRENT-VALUE ==> ' NC-CURR-VAL-DISP
           DISPLAY '*** NC-MIN-VALUE     ==> ' NC-MIN-VAL-DISP
           DISPLAY '*** NC-MAX-VALUE     ==> ' NC-MAX-VAL-DISP
           DISPLAY '*** NC-OPTIONS       ==> ' NC-OPTIONS-DISP.
*
*************  END - INQUIRE ************************
*************  BEGIN - DELETE ***********************
*
* DELETING COUNTER(COUNTER1) WHICH IS DEFINED IN POOL(POOL1)
*
*
        MOVE NC-DELETE    TO  NC-FUNCTION.
```

```
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
        MOVE 'COUNTER1'   TO  NC-COUNTER-NAME.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
                             NC-POOL-SELECTOR
                             NC-COUNTER-NAME
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-DELETE FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
           DISPLAY '*** NCTR-DELETE SUCCESSFUL. ***'.
*
*************  END - DELETE ************************
*************  BEGIN - FINISH ************************
*
* DISCONNECTING FROM THE NAMED COUNTER SERVER.
*
*
        MOVE NC-FINISH    TO  NC-FUNCTION.
        MOVE 'POOL1'      TO  NC-POOL-SELECTOR.
*
        CALL 'DFHNCTR' USING NC-FUNCTION
                             NC-RETURN-CODE
                             NC-POOL-SELECTOR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR
                             NULL-PTR.
*
        IF NC-RETURN-CODE NOT = NC-OK THEN
           DISPLAY '*** NCTR-FINISH FAILED. ***'
           DISPLAY '*** RETURN-CODE = ' NC-RETURN-CODE
        ELSE
```

```
              DISPLAY '*** NCTR-FINISH SUCCESSFUL. ***'.
      *
      *************  END - FINISH ************************
           DISPLAY '*** TEST PROGRAM ENDED ***'
           STOP RUN.
```

Erhan PASA
Senior System Programmer
AKNET AS (Turkey)

Printing TSO files under CICS – part 2

This month we conclude the code for using CICS to print TSO files.

PRINTC3 SOURCE CODE

```
*=====================================================================*
*                                                                     *
*  PRINTC3 - Relate this program to transaction "trans1", as defined *
*            in PRINTC REXX EXEC. Trans1 is started under CICS from  *
*            TSO via APPC by program PRINTC2. This program in turn   *
*            will start "trans2", directed to the printer terminal.  *
*                                                                     *
*=====================================================================*
*
PRINTC3  AMODE 31
PRINTC3  RMODE ANY
PRINTC3  DFHEIENT DATAREG=R9,EIBREG=R8
         DFHREGS
*
         MVC  COMAREA,=CL150' '            Clear communication area
         MVC  TCRE,=F'67'                  Create Status
         MVC  TSERV,=F'73'                 Inservice Status
         MVC  TACQ,=F'69'                  Acquire Status
         MVC  RECVALEN,=H'125'
         MVC  WCONVID,EIBTRMID
         EXEC CICS EXTRACT PROCESS                                  *
                           PIPLIST(R5)                              *
                           PIPLENGTH(PIPLEN)
         USING PIPLIST1,R5
         EXEC CICS IGNORE CONDITION INVREQ
```

```
         EXEC CICS RECEIVE CONVID  (WCONVID)                          *
                           INTO    (RECVAREA)                         *
                           LENGTH  (RECVALEN)
         SR   R4,R4                          Acquire loop counter
*
INQUIRE  EQU  *
         EXEC CICS INQUIRE TERMINAL  (PRINTER)                        *
                           ACQSTATUS (TSTATUS)
         CLC  TSTATUS,TACQ                   Is printer acquired?
         BE   OPENFILE                       Yes, leave this loop
         EXEC CICS SET TERMINAL   (PRINTER)                           *
                       CREATESESS (TCRE)                              *
                       SERVSTATUS (TSERV)                             *
                       ACQSTATUS  (TACQ)
*                                            No, try to acquire it
         EXEC CICS DELAY INTERVAL(10)
         AH   R4,=H'1'                       Increase loop counter
         CH   R4,=H'3'                       Limit reached, exit
         BE   ERRO1                          with error
         B    INQUIRE                        Otherwise, try again
*
OPENFILE EQU  *
         EXEC CICS SET                                                *
                   FILE (FICH)                                        *
                   ENABLED                                            *
                   OPEN
         EXEC CICS INQUIRE                                            *
                   FILE (FICH)                                        *
                   OPENSTATUS(FSTATUS)
         CLC  FSTATUS,=F'18'                 If file open OK,
         BNE  ERRO2                          start next transaction.
         EXEC CICS START                                              *
                   TRANSID (TRANS2)                                   *
                   TERMID  (PRINTER)                                  *
                   FROM    (PROGID )                                  *
                   LENGTH  (100)
         MVC  COMAREA,=CL150' '              Clear communication area
*
SENDBACK EQU  *
         EXEC CICS SEND CONVID (WCONVID)                              *
                        FROM   (COMAREA)                              *
                        LENGTH (48)                                   *
                        INVITE                                        *
                        WAIT
         EXEC CICS RETURN
*
*======= Error messages ============================================
*
ERRO1    EQU  *
```

```
         MVC  COMAREA,=CL48'Printer not Acquired'
         B    SENDBACK
ERRO2    EQU  *
         MVC  COMAREA,=CL48'CICS File not available'
         B    SENDBACK
*
*======= Area addressed by R9 ====================================
*
         LTORG
         DFHEISTG
         DS   ØF
WCONVID  DS   CL4        Conversation id
RECVAREA DS   CL121
RECVALEN DS   H          (FM5PIPLN)
PIPLEN   DS   H          (FM5PIPGD)
COMAREA  DS   CL15Ø
         DS   ØF
TCRE     DS   F          Create
TSERV    DS   F          Inservice
TACQ     DS   F          Acquired
TSTATUS  DS   F          Printer status
FSTATUS  DS   F          Inquire printer status
*
*======= Area addressed by R5 ====================================
*
PIPLIST1 DSECT
LLØØ     DS   CL4        (FM5PIPSL = LENGTH +  FM5PIPSG = X'12E2')
TRANS2   DS   CL4        Second transaction (started by this prog)
PROGID   DS   CL44       Tso name of file to print
         DS   CL2        CC parameter
         DS   CL3        LRECL parameter
FICH     DS   CL8        DDNAME of RRDS file to open
PRINTER  DS   CL4        Printer name under CICS
         DS   CL8Ø       Filler
         END
```

PRINTC4 SOURCE CODE

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. PRINTC4.
       ENVIRONMENT DIVISION.
       DATA DIVISION.
      *=============================================================*
      *
      * This program relates to "trans2", as defined in PRINTC REXX.
      * Trans2 is started by PRINTC3 against the printer terminal.
      *
      *=============================================================*
```

```
 WORKING-STORAGE SECTION.
*========================*
 77   K                 PIC S9(4)   COMP    VALUE +1.
 77   KLIMIT           PIC S9(4)   COMP    VALUE +1800.
 77   X                 PIC S9(8)   COMP    VALUE +0.
 77   N                 PIC S9(8)   COMP    VALUE +0.
 77   PARAMETROSL      PIC S9(4)   COMP    VALUE +100.
 77   NEW-PAGE         PIC 9               VALUE 1.
 77   PAGECOUNTER      PIC 9(3)            VALUE 1.
 77   PAGELINES        PIC 99              VALUE 0.
 77   LINCOUNTER       PIC 99              VALUE 0.
 77   FORMFEED         PIC X               VALUE X'0C'.
 77   FORMFEEDL        PIC S9(4)   COMP    VALUE +1.
 77   LINEFEED         PIC XX              VALUE X'0D15'.
 77   LINEFEEDL        PIC S9(4)   COMP    VALUE +2.
 77   EOM               PIC X               VALUE X'19'.
 77   EOML              PIC S9(4)   COMP    VALUE +1.
 77   PGHEAD1L         PIC S9(4)   COMP    VALUE +125.
 77   PGHEAD2-80L      PIC S9(4)   COMP    VALUE +61.
 77   SOURCELINE       PIC X(140)          VALUE SPACES.
 77   ABSTIM           PIC S9(15)  COMP-3  VALUE 0.
 77   CONTROLCHAR      PIC X               VALUE 'H'.
 77   NO-LINEFEED-132  PIC 9               VALUE  1.
*
 01  PARAMETROS.
     02  P-NOMEPROG    PIC X(44)   VALUE SPACES.
     02  CC            PIC XX      VALUE SPACES.
     02  LRECL         PIC 999     VALUE ZEROS.
     02  FICH          PIC X(8)    VALUE SPACES.
     02  FILLER        PIC X(43)   VALUE SPACES.
*
*================= Title: LetterGothic Bold Pitch11.
 01  PGHEAD1.
     02  FILLER        PIC X(11)   VALUE '&&??%%P%1B('.
     02  FILLER        PIC X(3)    VALUE 's0p'.
     02  FILLER        PIC X(5)    VALUE '12.70'.
     02  FILLER        PIC X(17)   VALUE 'h0s3b4102T%1B&l6D'.
     02  FILLER        PIC X(07)   VALUE '&&??000'.
     02  FILLER        PIC X(05)   VALUE SPACES.
     02  NOMEPROG      PIC X(45)   VALUE SPACES.
     02  DATA1         PIC X(10)   VALUE SPACES.
     02  FILLER        PIC X(03)   VALUE SPACES.
     02  HORA1         PIC X(08)   VALUE SPACES.
     02  FILLER        PIC X(08)   VALUE '    PAG.'.
     02  NUMPAG        PIC ZZ9.
*
*================= For 80 cols: Courier Pitch11.
 01  PGHEAD2-80.
     02  FILLER        PIC X(11)   VALUE '&&??%%P%1B('.
```

```
     02  FILLER        PIC X(3)    VALUE 's0p'.
     02  FILLER        PIC X(5)    VALUE '11.00'.
     02  FILLER        PIC X(10)   VALUE 'h0s0b4099T'.
     02  MARG80        PIC X(9)    VALUE '%1B&l300U'.
     02  TOPL80        PIC X(9)    VALUE '%1B&l100Z'.
     02  LPINCH        PIC X(7)    VALUE '%1B&l6D'.
     02  FILLER        PIC X(7)    VALUE '&&??000'.
*
*================= For 132 cols: LetterGothic Bold Pitch18.
 01  PGHEAD2-132.
     02  FILLER        PIC X(11)   VALUE '&&??%%P%1B('.
     02  FILLER        PIC X(3)    VALUE 's0p'.
     02  PIT132        PIC X(5)    VALUE '18.00'.
     02  FILLER        PIC X(10)   VALUE 'h0s3b4102T'.
     02  MAR132        PIC X(9)    VALUE '%1B&l005U'.
     02  TOP132        PIC X(9)    VALUE '%1B&l001Z'.
     02  LPINCH        PIC X(7)    VALUE '%1B&l6D'.
     02  FILLER        PIC X(7)    VALUE '&&??000'.
*
 01  PRINTBUFFER.
     02  PRINTBUF      PIC X(2000) VALUE SPACES.
*
 01  FILEBUFFER.
     02  LINHA  OCCURS 35000.
       04  CONTROLE    PIC X.
       04  LINHA2      PIC X(139).
*
*================================================================*
 PROCEDURE DIVISION.
*================================================================*
*
     EXEC CICS HANDLE ABEND LABEL (CLOSE-FILE)
     END-EXEC
     EXEC CICS HANDLE CONDITION ERROR  (CLOSE-FILE)
                                INVREQ (CLOSE-FILE)
                                NOTFND (READFILE-END)
                                ENDFILE(READFILE-END)
     END-EXEC
     EXEC CICS IGNORE CONDITION LENGERR
     END-EXEC
     EXEC CICS RETRIEVE INTO   (PARAMETROS)
                        LENGTH (PARAMETROSL)
     END-EXEC.
*
 READFILE.
*=========*
     ADD 1 TO N.
     IF N > 35000
        EXEC CICS ABEND ABCODE('9876')
```

```
           END-EXEC
        END-IF.
        MOVE SPACES TO LINHA(N)
        EXEC CICS READ FILE   (FICH)
                       INTO   (LINHA(N))
                       RIDFLD (N)
                       RBA
        END-EXEC
        GO TO READFILE.
*
 READFILE-END.
*=============*
        PERFORM CLOSE-FILE.
*
 FORMAT-PAGEHEADER.
*==================*
        EXEC CICS ASKTIME ABSTIME (ABSTIM)
        END-EXEC
        EXEC CICS FORMATTIME ABSTIME(ABSTIM)
                             YYYYMMDD (DATA1)
                             DATESEP('/')
                             TIME   (HORA1)
                             TIMESEP(':')
        END-EXEC
        MOVE  P-NOMEPROG  TO  NOMEPROG
        MOVE  SPACES      TO  SOURCELINE
        IF LRECL =  11Ø
           MOVE '14.5Ø'      TO  PIT132
           MOVE '%1B&l4ØØU'  TO  MAR132
           MOVE  PGHEAD2-132 TO  PGHEAD2-8Ø
        END-IF.
        IF LRECL =  12Ø
           MOVE  '16.2Ø'     TO  PIT132
           MOVE  PGHEAD2-132 TO  PGHEAD2-8Ø
        END-IF.
        IF LRECL =  132
           MOVE  '17.9Ø'     TO  PIT132
           MOVE  PGHEAD2-132 TO  PGHEAD2-8Ø
        END-IF.
        IF CC = SPACES
           MOVE  61          TO  PAGELINES
        ELSE
           MOVE  72          TO  PAGELINES
           MOVE  PGHEAD2-8Ø  TO  PRINTBUF(K:PGHEAD2-8ØL)
           ADD   PGHEAD2-8ØL TO  K
        END-IF.
*
 NEXT-LINE.
*==========*
```

```
      ADD 1 TO X
      IF X > N
         PERFORM FORM-FEED
         PERFORM SEND-BUFFER
         GO TO RETORNAR
      END-IF.
      MOVE SPACES TO SOURCELINE
      IF NEW-PAGE = 1 AND CC = SPACES
         MOVE Ø TO NEW-PAGE
         PERFORM PAGEHEADER
      END-IF.
      IF CC = SPACES
         MOVE  LINHA(X)  TO  SOURCELINE
         PERFORM SOURCELINE-OUT
         GO TO NEXT-LINE
      END-IF.
      IF CONTROLE(X) = '1'
         PERFORM FORM-FEED
      END-IF.
      IF CONTROLE(X) = 'Ø'
         PERFORM LINE-FEED
      END-IF.
      IF CONTROLE(X) = '-'
         PERFORM LINE-FEED 2 TIMES
      END-IF.
      MOVE LINHA2(X)  TO  SOURCELINE
      PERFORM SOURCELINE-OUT
      GO TO NEXT-LINE.
*
 PAGEHEADER.
*===========*
      MOVE PAGECOUNTER TO  NUMPAG
      MOVE PGHEAD1     TO  PRINTBUF(K:PGHEAD1L)
      ADD  PGHEAD1L    TO  K
      PERFORM LINE-FEED.
      MOVE PGHEAD2-8Ø  TO  PRINTBUF(K:PGHEAD2-8ØL)
      ADD  PGHEAD2-8ØL TO  K
      PERFORM LINE-FEED 2 TIMES.
*
 FORM-FEED.
*==========*
      MOVE FORMFEED  TO PRINTBUF(K:FORMFEEDL)
      ADD  FORMFEEDL TO K
      MOVE LINEFEED  TO PRINTBUF(K:LINEFEEDL)
      ADD  LINEFEEDL TO K
      ADD  1 TO PAGECOUNTER
      MOVE Ø TO LINCOUNTER
      MOVE 1 TO NEW-PAGE.
*
```

```
 LINE-FEED.
*==========*
     ADD 1 TO LINCOUNTER
     MOVE LINEFEED  TO PRINTBUF(K:LINEFEEDL)
     ADD  LINEFEEDL TO K
     IF LINCOUNTER > PAGELINES
        PERFORM FORM-FEED
     END-IF.
*
 SOURCELINE-OUT.
*==============*
     MOVE SOURCELINE  TO PRINTBUF(K:LRECL)
     ADD  LRECL       TO K
     IF NOT (LRECL EQUAL 132 AND NO-LINEFEED-132 = 1)
        PERFORM LINE-FEED
     END-IF.
     IF K > KLIMIT
        PERFORM SEND-BUFFER
     END-IF.
*
 SEND-BUFFER.
*===========*
     MOVE EOM  TO PRINTBUF(K:EOML)
     ADD  EOML TO K
     EXEC CICS SEND FROM(PRINTBUF)
                    LENGTH(K)
                    CTLCHAR(CONTROLCHAR)
     END-EXEC
     MOVE SPACES TO PRINTBUF
     MOVE 1 TO K.
*
 CLOSE-FILE.
*==========*
     EXEC CICS HANDLE CONDITION ERROR NOHANDLE
     END-EXEC
     EXEC CICS HANDLE ABEND NOHANDLE
     END-EXEC
     EXEC CICS SET FILE(FICH) CLOSED
     END-EXEC.
*
 RETORNAR.
*========*
     EXEC CICS RETURN
     END-EXEC.
```

Information point – reviews

Given that most of the names for the computing industry have the word 'Information' in them, such as Information Technology (IT), we are often expected to be information experts. With that in mind, a search has been made for the best places for information on CICS.

Internet Web sites have grown to the point where they now provide most of the best information sources. We begin with the best example of the Internet providing free highly-usable access to what previously would have cost you tens of thousands of dollars/pounds/euros for paper-based manuals. Admittedly, IBM has offered manuals on CD-ROM and tape for the last 15 years, but these always required a significant effort to make them work effectively for those who use them. The biggest long-term headache was keeping everything up-to-date, which required a detailed knowledge of exactly which release(s) (test and production) of every piece of installed software were in use.

IBM MANUALS ON THE INTERNET

IBM offers thousands of its manuals on the Internet at no cost for Web viewing and/or downloading, including many CICS-related publications. Originally only in a browser-based simulation of BookManager, many are now (also or exclusively) being offered in Adobe Acrobat (.pdf) format.

CICS Library – http://www.ibm.com/software/ts/cics/library/#books

CICS Library provides links to the manuals for the last or most recent release of all supported versions of CICS for z/OS and VSE/ESA.

Manuals for some unsupported releases and versions, as well as related CICS products, can be found by browsing these pages:

- z/OS (A-C) – http://www.ibm.com/servers/eserver/zseries/zos/bkserv/zappls1.html.
- OS/390 (A-C) – http://www.ibm.com/servers/s390/os390/bkserv/shelves_d2.html.

- MVS/ESA (A-C) - http://www.ibm.com/servers/s390/os390/bkserv/mvs_shelves3.html
  - CICS/MVS 2.1.2 - http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/DFHHSH21.
- VSE/ESA - http://www.ibm.com/servers/s390/os390/bkserv/vse
  - CICS/VSE 2.3 - http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/Shelves/DFHVSH17.

All mainframe manuals – http://www.ibm.com/servers/eserver/zseries/zos/bkserv

This page provides a z/OS-centric entry point to all on-line IBM mainframe manuals. You can search directly from the *Find It* section near the top of the page, or browse by operating system and version/release (or hardware), then look for links labelled *List - all bookshelves*, then *List books*. There is even information on accessing licensed manuals.

REDBOOKS

http://www.redbooks.ibm.com

Redbooks have always provided an outlet for less formal material than manuals, but much of it of extreme value. Many of them are essentially detailed diaries of IBM projects, at both customer and IBM sites, with some effort made to remove installation-specific information in the sample code and text. Others are based on technical presentations at conferences and other venues. A more recent trend, exemplified by the five volume *ABCs of System Programming*, is to provide the type of textbooks you would expect to find in a computer bookstore.

You can do a direct search by keyword and find over 104 matches for CICS. Or you can click on Redbooks Online on the left sidebar. There you will find a more comprehensive search capability. The right sidebar provides access to a dozen Redbook Portals by subject area (*zSeries* for CICS), from where you can list all the Redbooks on one

page, most recent first, or just the 15 most popular. These portals also provide access to Redpieces, Redpapers, and Residencies. Redpieces are redbooks in progress. Redpapers do not qualify as redbooks for one reason or another, and are only available on-line. Residencies list the future openings to be a member of the team that creates a specific Redbook.

IBM TRANSACTION SYSTEMS HOME PAGE

http://www.ibm.com/software/ts

The *Products* drop-down list in the centre of the screen provides access to product groups. Selecting CICS provides a general CICS page, from where you can select exactly which CICS product you are interested in. The *Products* drop-down list on the CICS page is also where you will find REXX for CICS.

All of these pages have a left sidebar with additional choices of interest. Although some of the sidebar entries remain the same, the results vary. Selecting *Case studies* on the general CICS page lists only the CICS case studies. From the Transaction Systems page, IMS and WebSphere are also included.

Download is unique to the CICS page and has some interesting software, support, and documentation. There are often beta test programs available that are listed here, too. Likewise, you have to go to the CICS page to find the *Library* link. There you will find the following sections:

- CICS Transaction Server
- Information Center
- Articles
- Books
- Brochures
- Flyers
- Magazines

- Presentations
- Redbooks
- Whitepapers.

Under *Magazines*, you will find recent editions of *CICS Transaction 390 Magazine* available for download in Adobe Acrobat (.pdf) format. You can also sign up for a subscription and have paper copies of each new issue mailed to you as they are published.

Also on the CICS page, in the lower right corner, is a section entitled *More resources*. *CICS and the Web*, *CICS and Java*, as well as the aforementioned *CICS Transaction 390 Magazine*, are some of the links you will find there. Beyond what you will find on the CICS and Java page, there are other sources, too:

- CICS Transaction Server for OS/390 Support for Java – http://www.ibm.com/software/ts/cics/java/ejb.
- The CICS Java interface – http://www.ibm.com/software/ts/cics/library/manuals/jcics.

BOB YELAVICH'S HOME PAGE

http://www.yelavich.com

After 40 years with IBM and 28 years with CICS, Bob retired in December 1996. By that time, he had become 'The Voice of CICS', perhaps best known for the satellite video broadcasts he produced on the IBM Field Television Network (FTN). Today, in retirement, he maintains his own Web site, the highlight of which is his twice weekly *CICS Newsletter*. In the left sidebar, as well as Bob's profile, you will find two other areas of interest:

- CICS-related documents and presentations.
- CICS Reference information and trivia.

Where else would you find summaries of IBM CICS announcement letters all the way back to 1968? But you will also find up-to-the-minute CICS technical information, such as *A Perspective on CICS*

"DPL" in *CICS-Related Documents*, where Bob takes an in-depth look at the various linkage methods (how one program calls another) and how they differ. Or there's *Web-Enabling CICS Applications – A Selection Guide*. There is a lot here.

XEPHON IS LIBRARY

http://www.xephon.com/is_library.html

Click the *IS Library* button in the left sidebar for free access to 400 articles extracted from Xephon Reports, including many on CICS and related topics. They are divided into five categories:

- Business and IT strategies
- New technologies and applications
- Data centre management
- System/390
- Other systems.

System/390 is where you will find CICS-specific articles. Click on an article title to have an Adobe Acrobat (.pdf) version e-mailed to you; free registration is required the first time. To locate titles of interest on this page, select text anywhere on the right side of the page and then use your browser's *Find* function to search the titles. Alternatively, you can use the *Search Site* function on the left sidebar to list both articles and *Update* journal pages.

MVS HELP

http://mvshelp.com

The *Help Boards* and *Interview Q&As*, both buttons on the left sidebar, are the areas of interest here. The *Help Boards* button opens into a new window where you will find that CICS has a moderated forum of its own. It does not look like much until you realize that the

default only displays posts from the last 10 days. From the drop-down list in the top right corner, select *Show all topics* and hit the *Go* button. The CICS forum averages nearly 1000 posts per year. There are over 2500 registered forum members, though not all participate in CICS.

Interview Q&As is an unlikely sounding source of useful information. And when you hit the button for it, the warning that greets you makes it sound even less appealing. But click on the CICS link and you will see a very long list of CICS questions with answers. The spelling, grammar and even the technical accuracy is occasionally in question, but what remains is a large and valuable FAQ (Frequently Asked Question) resource.

CICS TIPS AND TRICKS

http://www.intac.com/~gbergman/tips.html

There may be less than 20 tips here, but the quality is high, since they are reviewed by a CICS guru before being posted (there is also the matter of a free gift to successful submitters). Quite a few are technical in nature, and a couple include Assembler code.

CICS-L

news:bit.listserv.cics-l

This is a popular newsgroup, averaging nearly 1000 posts per month. There are two Web-accessible, though not very Web-friendly, archives. Both go back more than a decade. But each is quite different from the other, so pick the one that best suits your needs.

- http://www.marist.edu/htbin/wlvindex?CICS-L
- http://vm.akh-wien.ac.at/wa/~LISTSERV/CICS_L.

It is easy to miss the fact, but the first archive includes a search feature right at the bottom of the very long Web page. Unfortunately, searches take a very long time to complete.

Newsreaders such as Outlook Express prompt you to subscribe upon

exit from a newsgroup. Each future posting (1000 per month) will then be e-mailed to you. To subscribe directly, e-mail listserv@uga.cc.uga.edu with the contents of the message *SUBSCRIBE CICS-L* followed by your name (first name then last name, separated by a blank) on the same line.

SHARE CICS PROJECT

http://www.share.org/cics.html

There is not a lot of technical information here, but it serves as a reminder of the huge amount of information that is presented at SHARE conferences in North America and GSE in Europe. GUIDE has merged with SHARE in Europe and its North American organization has closed, leaving its members to join SHARE.

A FINAL NOTE

It is always worth mentioning, when reviewing Internet information sources on any subject, that accuracy varies widely. Even though *CICS Update* and other Xephon journals do not guarantee accuracy of everything printed, the review process catches most errors. On the other hand, Internet forums, even moderated ones, have a much higher error rate.

This is all part of a wider problem that affects almost everyone since the use of the Internet has become so widespread. The reliability of the (source of) information you have found must be evaluated before accepting the information as valid. Before the Internet, the *cost to publish* for printed material was high enough that most sources tended to be accurate.

The importance of this point became obvious as I reviewed a CICS forum on the Internet, followed a particular thread, then discovered an article on exactly the same topic in the then-current issue of *CICS Update*. The difference between educated guesses, and researched and tested answers.

Jon E Pearkins
Adiant Corporation (Canada)

CICS questions and answers

Q I need to have my CICS regions running 24 hours a day. However, the CEEMSG, MSGUSR, and other queues get pretty big after a couple of days. Someone told me that there is a way I can 'archive' the previous day and start the new day with a fresh one. Can this be done? And if so, how?

A The answer to this is different depending on the version of CICS. For CICS/ESA Version 4.1. and below the following should be added to the SYSOUT DD that needs to be segmented:

```
//MSGUSR   DD   SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
//MSGUSR   DD   SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
//MSGUSR   DD   SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
//MSGUSR   DD   SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
//MSGUSR   DD   SYSOUT=*,FREE=CLOSE,SPIN=UNALLOC
 **repeat above for each day CICS will be up
                      (or for each Open/Close) **
//MSGUSR   DD   SYSOUT=*
```

Then a Close/Open of the 'Extra' TD queue will produce the desired results. (A program can be written to automate this process – see the section below for further details). While the queue is closed (although for a short time) messages written will be lost, so care should be taken over the timing of the Close/Open.

For CICS TS Version 1.1. and above the following needs to be done:

- Ensure the DCT has been migrated to RDO and that the SYSOUT DD cards have been removed from the CICS JCL. CICS dynamically allocates these entries just like files.
- A Close/Open of the 'Extra' TD queue will produce the desired results.

A simple program can be written to issue the SPI calls to Close and Open the required 'Extra' TD queues. The queues to Close/Open could be listed in the program, or the program could browse

the list of 'Extra' TD queues and Close/Open all of them for example. The program could be started from PLTPI ready to run for the first time just after midnight and then re-schedule itself to run just after midnight again.

Q Where can I find information on how CICS does the pseudo-reentrancy for COBOL programs? Alternatively, where can I find source for DFHECI?

A IBM provide access to the source for DFHECI (and other modules) via VPL (View Program Listing). VPL replaces the old microfiche system. See http://publib.boulder.ibm.com/pubs/html/vpl/vpl_ugcust22.html for the user guide to VPL, which gives you an idea of what you'd get.

To gain access to VPL use the following contacts:

- For customers in Canada the contact is Rachele Janczyn, 905-316-3280
- For customers in Europe, Middle East, and Africa the contact is the DIAL-IBM Coordinator in each country.
- For customers in the USA the contact is Robert Maddera, 303-939-3649 or email bmaddera@us.ibm.com.

Q How do I remove an auto-installed Connection? I'm unable to discard it, I need to test changes to the model and I don't have hundreds of PCs at hand!

A Set the Connection 'Rel' and 'Out', Define (using CEDA) a new Connection with the same ID as the one you wish to remove. Make the Netname 'DUMMY' (for example), and install the new Connection. Auto-install will now be driven as the Netname is not found already connected to CICS.

Q I've heard that others code DD DUMMY for DFHAUXT/DFHBUXT, DFHDMPA/DFHDMPB – how does this work? What if I need to take an IBM format dump or run AuxTrace?

A Yes, this is possible, just DD DUMMY, for example, DFHAUXT. When you want to get a Trace use ADYN transaction (see IBM supplied RDO group: DFH$UTIL) to dynamically allocate your

Aux Trace dataset like so:

```
ADYN   (press Enter)
FREE DDNAME(DFHAUXT) UNALLOC   (Press Enter)
```

You should get the following message:

```
DYNALLOC RETURN CODE  0 , ERROR CODE  X'0000' , REASON CODE  X'0000'
```

This has dropped the DUMMY allocation.

Now overtype the command (FREE DDNAME....) with:

```
ALLOC DDNAME(DFHAUXT) DSNAME(my.auxt.file) STAT(SHR)   (Press Enter)
```

You should get the following message:

```
DYNALLOC RETURN CODE  0 , ERROR CODE  X'0000' , REASON CODE  X'0000'
```

EOF (Erase-End of Field) the command line and press *Enter* to exit ADYN. You should get:

```
TRANSACTION END
```

Now use CEMT/CETR and then capture your Trace.

To set things back the way you found them do the following:

Ensure DFHAUXT is Closed.

```
ADYN   (press Enter)
FREE DDNAME(DFHAUXT) UNALLOC   (Press Enter)
ALLOC DDNAME(DFHAUXT) DUMMY   (Press Enter)
(EOF, Press Enter)
```

Q The CICS 3270 Bridge does not support all API commands. Is there a scan utility like DFHMSCAN to help identify unsuitable modules?

A At CICS/TS Version 1.3. the utility program DFHEISUP can be used to do this. You will need to review and APPLY PTF UQ49286 (APAR PQ42109). This utility can be used to scan for API calls.

If you have any CICS-related questions, then please send them in and we will do our best to find answers. Alternatively, e-mail them directly to cicsq@xephon.net.

TIBCO Software has announced plans for integrating the mainframe into its enterprise application integration environment, combining TIBCO Substation ES for IBM and mainframe integration using TIBCO ActiveEnterprise sites to cross-access existing applications running on large mainframe environments with other systems.

TIBCO Adapter for CICS resides outside the mainframe environment and uses the CICS Client to enable non-mainframe applications to request data from and interact with CICS applications that are running on the mainframe.

TIBCO Substation ES resides on the mainframe and enables non-mainframe applications to send and receive messages on behalf of transaction processing systems such as CICS and IMS that are running on z/OS and OS/390. It's now available with interfaces to CICS and IMS.

For further information contact:
TIBCO Software, 3165 Porter Drive, Palo Alto, CA 94304, USA.
Tel: (650) 846 1000.
URL: http://www.tibco.com.

* * *

IBM has announced Version 1.1 of its CICS Interdependency Analyzer (IA) for z/OS and OS/390 for identifying resource interdependencies within CICS systems.

The run-time tool provides information for the systems programmer to split workloads and allow for more efficient workload balancing across CICS Plex and Sysplex. This information, in report form, can be interrogated online.

CICS IA is a series of software modules that identify the sets of CICS resources used by individual CICS transactions. It intercepts EXEC CICS commands and records those that act on transactions, programs, BMS maps, files, TS queues, and TD queues. It supports all current levels of CICS TS and CICS/ESA 4.1.

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

* * *

IBM has announced CICS Transaction Server for z/OS Version 2.2 with a range of major enhancements, including facilities to support applications written in Java and for reuse or incorporation of existing applications and data within these applications.

It supports Java SDK 1.3, and session beans conforming to the J2EE EJB 1.1 spec.

Other new functions aimed at application development include an enhanced 3270 bridge; an integrated CICS translator for use with COBOL, PL/I, CICS COBOL, and PL/I XML application capability; and enhancements to function shipping of remote file requests.

Connectivity is addressed with support for external call interface (ECI) over TCP/IP, improved CICS exploitation of TCP/IP services, connection optimization, and support for VTAM LU alias facility.

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