November 2004

In this issue

3 Designing efficient CICS screens
10 Creating useful IMS buffer handler statistics
17 Execute native CEMT commands from batch – part 2
28 EXCI batch client control program
48 December 2001 – November 2004 index
50 CICS news

© Xephon Inc 2004
Subscriptions and back-issues
A year's subscription to CICS Update, comprising twelve monthly issues, costs $270.00 in the USA and Canada; £175.00 in the UK; £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 2000 issue, are available separately to subscribers for $24.00 (£16.00) 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.

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

© Xephon Inc 2004. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher.

Printed in England.
Designing efficient CICS screens

A good screen design is critical to a successful on-line system. Programmers designing the screen should put themselves in the position of the terminal operator who is going to use the screen. This will help in developing a user-friendly design.

Excessive terminal response time inevitably becomes the most obvious focus of user complaints in large systems and multi-terminal networks. The three fundamental variables influencing terminal response time are overall system tuning, individual application program efficiency, and the quantity of data transmitted. This article focuses on these bottlenecks and suggests ways to improve response time and design a user-friendly screen.

The points described below are pivotal to user-friendly and efficient screen design:

1. **Self-explanatory fields** – the field labels on the screen should convey their purpose, but they should not be too descriptive. They should contain just enough information for the user to get a hint from the field name. For example, PRODUCT ID can be coded as PRODID and DEPARTMENT ID as DEPTID. The top-left corner on the screen can display the transaction code and the top-right corner the corresponding application program.

2. **Sequence of fields on screen** – the more important or mandatory fields should be coded on the top part of the screen and should flow through from the left side to the right. This will make data entry faster.

3. **Highlighting all the fields in error** – if more than one field is in error, then the screen should highlight all of them, not just the first one, and the error message on the screen should be for the first field in error. For example, if fields
ACCOUNT and GROUP are in error, the screen will highlight both the fields but the error message should say ‘ACCOUNT NO IS NOT CORRECT’. Once the user corrects ACCOUNT, the screen should highlight GROUP and the error message should be for GROUP.

4 Minimizing the number of keystrokes to fetch a page. If a multi-page screen is being developed, then code a field PAGE on the screen. This should indicate which PAGE number of the total number of pages is being displayed on the screen. If the user types \( n \) in the PAGE field, it should fetch the corresponding \( n \)th page. This field will save the keystrokes required to scroll down; for example, let’s say that a screen has 20 pages and the user wants to see the last page, s/he will have to press a program function key 19 times before s/he gets to see page 20. On the other hand, if the user enters 20 in the PAGE field and presses the Enter key, s/he can directly get page 20 in one keystroke. One alternative would be to define a user-defined key for retrieving the last page or the first page.

Similarly a field Record Number could be coded on a screen to fetch a particular record to a screen having many records.

5 Minimizing the data transmission. It is sensible to keep the length of datastreams short. It is particularly important to keep the number of terminal transmissions as small as possible because this may well be the slowest part of the path a transaction takes. The efficiency of the datastream therefore affects both response time and line usage.

The MAPONLY/DATAONLY option (in the SEND command) and MDT (Modified Data Tag), if used intelligently, can reduce the data transfer between the terminal and the application program. MDT is a one-bit attribute associated with a field that indicates whether a field has been changed on the screen. If it is off (0), it means the field has not been modified by a terminal.
operator. If it is on (1), that means the field has been modified by the terminal operator. If option FRSET is coded for the mapset/map macro, MDT will be set to off (0) for all fields of the mapset/map.

If you have a screen with many input fields, which you may have to read several times, you can reduce the length of the input datastream by specifying FRSET when you write back to the screen in preparation for the next read. FRSET turns off the MDTs, so those fields entered before that write are not present unless the user re-enters them the next time. If you are dealing with a relatively full screen and a process where there may be a number of error cycles (or repeat transmissions for some other reason), this can be a substantial saving. However, because only changed fields are sent on subsequent reads, the program must save input from each cycle and merge the new data with the old. This is not necessary if you are not using FRSET, because the MDTs remain on, and all fields are sent regardless of when they were entered.

The MAPONLY option sends only the constant data in a map, and does not merge any variable data from the program. When you send a skeleton screen to be used for data entry, you can often use MAPONLY.

Sending only changed fields is important when, for example, a message is added to the screen, or one or two fields on an input screen are highlighted to show errors. In these situations, you should use the DATAONLY option to send a map that consists of nulls except for the changed fields. For fields in which only the attribute byte has changed, you need to send only that byte, and send the remaining fields as nulls. BMS uses this input to build a datastream consisting of only the fields in question and all other fields on the screen will remain unchanged.

6 Allowing for more data entry space on the screen. The screen design should make space for more data entry on
Figure 1: Designs 1 and 2
the screen – but this should not overcrowd the screen or violate point 1 mentioned above.

In Figure 1, Design 2 saves three lines per page by coding the first header detail line only once instead of for each transaction detail line, and shifting the second transaction detail line inside a little to make the first header detail line more visible.

The result is obvious because Design 2 has space for one more transaction detail line than Design 1.

Note: the screens in Figure 1 are designed for 24x80 3270 terminals. The hyphen (-) characters on the screens indicate unprotected user-entered fields.

7  **Automatically populate the known fields.** A few screen fields can be populated automatically depending on previously-entered fields or from a combination of fields. This will save a couple of keystrokes by the user and will provide faster data entry. Let’s say in the above screen, if a particular DEPTID ‘SALES’ is located in STATE ‘CA’, then SALES should be populated automatically by a program when the user enters CA in the STATE field.

8  **Cursor positioning.** If the screen is being produced for the first time, the cursor should be on the first data entry/unprotected field on the screen. If a field is in error, the cursor should be placed on that field position using dynamic cursor positioning.

9  **Using skipper and stopper techniques.** Skipper allows the cursor to skip, using the ASKIP attribute, from one unprotected field to the next in spite of a gap between them. Stopper allows the cursor to be stopped after the unprotected field, thereby preventing field overflow using the PROT attribute. These increase user-friendliness, but stoppers should not be used excessively because they slow down the data entry operations.

10  **Use native terminal control commands for sending**
unformatted data. If your output to a terminal is entirely, or even mostly, unformatted, you can send it using native terminal control commands rather than BMS (that is, using SEND without the MAP option). This command is much more efficient in terms of processor overhead.

11 *Do not send blank fields to the screen.* Sending to the screen fields that consist entirely of blanks or that are filled out on the right by trailing blanks usually wastes line capacity. The only case in which BMS requires you to do this is when you need to erase a field on the screen that currently contains data, or replace it with a datastream shorter than that currently on the screen, without changing the rest of the screen.

This is because when BMS builds the data-stream representing your map, it includes blanks but omits nulls. This makes the output datastream shorter. BMS omits any field whose first data character is null regardless of subsequent characters in the field.

BMS requires you to initialize to nulls any area to be used to build a map. BMS uses nulls in attribute positions and in the first position of data to indicate that no change is to be made to the value in the map. If you are re-using a map area in a program, you should take special care to clear it in this way.

12 *Design data entry operations to reduce line traffic.* Often, users are required to enter data on the same screen several times. Only the data changes on each cycle; the titles, field labels, instructions, and so on remain unchanged. In this situation, when an entry is accepted and processed, you can respond with a SEND CONTROL ERASEAUP (or a map that contains only a short confirmation message and specifies ERASEAUP). This causes all the unprotected fields on the screen (that is, all the input data from the last entry) to be erased and to have their MDTs reset. The labels and other text, which are in
protected fields, are unchanged. The screen is ready for the next data-entry cycle, and only the necessary data has been sent.

13 **Use nulls instead of blanks.** Outside BMS, nulls have no special significance in an output datastream. If you need a blank area on a screen, you can send either blanks or nulls to it; they take up the same space in the output stream. However, if the blank field is likely to be changed by the user and subsequently read, use nulls, because they are not transmitted back.

14 **Compress data sent to the screen.** When you send unformatted data to the screen, or create a formatted screen outside BMS, you can compress the data further by inserting Set Buffer Address (SBA) and Repeat-to-Address (RA) orders into the datastream. SBA allows you to position data on the screen, and RA causes the character following it to be generated from the current point in the buffer until a specified ending address occurs. SBA is useful whenever there are substantial unused areas on the screen that are followed by data. RA is useful when there are long sequences of blanks on the screen. Please note that if you wish to insert an SBA into the datastream, then the values must be ASCII and not EBCDIC.

15 **Use methods that avoid the need for nulls or blanks.** If there is a requirement for any large area of a screen to be blank, you should consider methods other than transmitting blanks or nulls. For example, using BMS, putting SBA and RA orders directly into the datastream, or using the ERASE and ERASEAUP options.
Creating useful IMS buffer handler statistics

In *CICS Update* issues 221 and 222, April and May 2004 (see *Helpful exit for shutdown assistant users*), we saw that it is possible to do many important and helpful things during the CICS shutdown procedure. In a similar manner, it is also possible to call program CSMON05 using:

```
EXEC CICS LINK PROGRAM('CSMON05')
```

to create IMS/DB statistics.

This article contains:

*Figure 1: IMSDCT definitions*
The definitions for the IMSDCT to write the statistic on transient data are shown in Figures 1 and 2.

1. The definitions for the IMSDCT to write the statistic on transient data.
2. Program CSMON05 to write the statistics in the transient data queue.
3. A dummy PSB named CICSPSB, which you can keep available in each environment.
4. A sample buffer handler statistic.

The definitions for the IMSDCT to write the statistic on transient data are shown in Figures 1 and 2.
CSMON05

CSMON05 TITLE '*** CICS IMS BUFFER STATISTICS ***'
SPACE
DFHEISTG DSECT
SPACE
* ****************************** ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ************ ........
BAS R6,DUMP WRITE TRANSACTION-DUMP
B CSIMS999 RETURN TO CSSHUT

SPACE

CSIMS100 DS ØH

SPACE

EXEC DLI STAT USING PCB(1) INTO(IMSAREA) LENGTH(STATLEN)

SPACE

CLC DIBSTAT,BLANK
BE CSIMS200

CLC DIBSTAT,=C'GA'
BE CSIMS900

MVC CODE,=C'STAT' MOVE DUMP-CODE
BAS R6,DUMP WRITE TRANSACTION-DUMP
B CSIMS999 RETURN TO CSSHUT

SPACE

CSIMS200 DS ØH

CP PASSFLAG,=P'1'
BNE CSIMS300

SPACE

EXEC CICS WRITEQ TD QUEUE ('IMS') FROM (IMSAREA1) LENGTH (ARLENGTH) RESP (RESPONSE)

SPACE

CLC RESPONSE,DFHRESP(NORMAL)
BE CSIMS250

MVC CODE,=C'QUE1' MOVE DUMP-CODE
BAS R6,DUMP WRITE TRANSACTION-DUMP
B CSIMS999 RETURN TO CSSHUT

SPACE

CSIMS250 DS ØH

SPACE

EXEC CICS WRITEQ TD QUEUE ('IMS') FROM (IMSAREA2) LENGTH (ARLENGTH) RESP (RESPONSE)

SPACE

CLC RESPONSE,DFHRESP(NORMAL)
BE CSIMS270

MVC CODE,=C'QUE2' MOVE DUMP-CODE
BAS R6,DUMP WRITE TRANSACTION-DUMP
B CSIMS999 RETURN TO CSSHUT

SPACE

CSIMS270 DS ØH

SPACE

AP PASSFLAG,=P'1'

SPACE
CSIMS300 DS 0H
SPACE
EXEC CICS WRITEQ TD
  * QUEUE ("IMS")
  * FROM (IMSAREA3)
  * LENGTH (ARLENGTH)
  * RESP (RESPONSE)
SPACE
CLC  RESPONSE,DFHRESP(NORMAL)
BE CSIMS100
MVC  CODE,=C'QUE3'
    MOVE DUMP-CODE
BAS  R6,DUMP
    WRITE TRANSACTION-DUMP
B  CSIMS999
    RETURN TO CSSHUT
SPACE
CSIMS900 DS 0H
EXEC DLI TERM
CLC  DIBSTAT,BLANK
BE  CSIMS999
MVC  CODE,=C'TERM'
    MOVE DUMP-CODE
BAS  R6,DUMP
    WRITE TRANSACTION-DUMP
SPACE
CSIMS999 DS 0H
SPACE
EXEC  CICS RETURN
EJECT
* ******************************************************************* *
* SUB-ROUTINES                                                        *
* ******************************************************************* *
SPACE
* ******************************************************************* *
* WRITE TRANSACTION-DUMPS FOR EVERY ABNORMAL RESPONSE-CODES            *
* ******************************************************************* *
SPACE
DUMP     DS 0H
SPACE
EXEC  CICS DUMP TRANSACTION DUMPCODE(CODE) RESP(RESPONSE)
SPACE
CLC  RESPONSE,DFHRESP(NORMAL)
BE  DUMP999
SPACE
EXEC  CICS WRITE OPERATOR TEXT(MSG001) RESP(RESPONSE)
SPACE
DUMP999  DS 0H
BR  R6
    RETURN TO CALLER
SPACE
MSG001  DC  CL50'CSMON05-001 A TRANSACTION-DUMP CAN NOT BE WRITTEN!'
  BLANK  DC  CL20' '       WORK WITH BLANKS
  STATLEN DC  Y(L'IMSAREA)
    LENGTH OF WA STAT-CALL
  ARLENGTH DC  Y(L'IMSAREA1)
    LENGTH OF TD-RECORD
EJECT
* ******************************************************************* *
<table>
<thead>
<tr>
<th>BSIZ</th>
<th>NBUF</th>
<th>RET</th>
<th>RBA</th>
<th>KEY</th>
<th>ISRT</th>
<th>ES</th>
<th>ISRT</th>
<th>KS</th>
<th>BFR</th>
<th>ALT</th>
<th>BGWRT</th>
<th>SYN</th>
<th>PTS</th>
<th>GETS</th>
<th>SCHBFR</th>
<th>FOUND</th>
<th>READS</th>
<th>USR</th>
<th>WTS</th>
<th>ERRORS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2K</td>
<td>500</td>
<td>84162</td>
<td>132</td>
<td>0</td>
<td>322</td>
<td>445</td>
<td>0</td>
<td>486</td>
<td>51768</td>
<td>250</td>
<td>49393</td>
<td>339</td>
<td>1602</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4K</td>
<td>500</td>
<td>6173</td>
<td>3934</td>
<td>0</td>
<td>60</td>
<td>155</td>
<td>0</td>
<td>486</td>
<td>8094</td>
<td>130</td>
<td>7890</td>
<td>368</td>
<td>92</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8K</td>
<td>500</td>
<td>285032</td>
<td>18186</td>
<td>0</td>
<td>249</td>
<td>3220</td>
<td>0</td>
<td>486</td>
<td>160916</td>
<td>2263</td>
<td>160071</td>
<td>3480</td>
<td>1425</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12K</td>
<td>1843</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>1368</td>
<td>0</td>
<td>1295</td>
<td>73</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16K</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20K</td>
<td>5</td>
<td>0</td>
<td>448</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>448</td>
<td>0</td>
<td>444</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>.5K</td>
<td>20</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>445</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1K</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>8464</td>
<td>79</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2K</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>39</td>
<td>40</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4K</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>5956</td>
<td>308</td>
<td>23</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8K</td>
<td>500</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>14180</td>
<td>106</td>
<td>1</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32K</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>486</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00/00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3: Example statistics
* REGISTER EQUATES                                          *
* *----------------------------------------------------------------*
  SPACE
  EQUREG
  EJECT
* *----------------------------------------------------------------*
* LITERALS                                                    *
* *----------------------------------------------------------------*
  SPACE
  LTORG
  DC  C’ ’
  END

**DUMMYPSB**

PRINT ON,DATA,GEN
*---------------------------------------------------------------*
* D U M M Y - P S B                                           *
* -------------------------------------------------------------*
* THIS PSB IS NEEDED FOR IMS-BUFFER-STATISTICS FOR THE          *
* ONLINE-ENVIRONMENT.                                         *
* IT IS AVAILABLE ON                                           *
* 'IMS.TEST.PSBLIB'                                            *
* 'IMS.PROD.PSBLIB'                                            *
*---------------------------------------------------------------*
* PCB  TYPE=DB,DBDNAME=DØ34DB,PROCOPT=GOT,KEYLEN=5              *
  SENSEG NAME=ADØØ                                           *
*---------------------------------------------------------------*
* SPACE 5                                                      *
  PSBGEN LANG=COBOL,PSBNAME=CICSPSB                           *
*---------------------------------------------------------------*
* END                                                           *

**STATISTICS**

An example of the output is shown in Figure 3.

_Claus Reis_
_CICS Systems Programmer_
_Nuernberger Lebensversicherung AG (Germany)_  © Xephon 2004
Execute native CEMT commands from batch – part 2

This month we conclude the code for a REXX EXEC that executes native CEMT commands from batch using the CPSM API.

/* Build print lines. Default strips and prefixes date and timestamp */
/* @BLANK - Blank line, no date and timestamp */
/* @      - No stripping, retains leading blanks */
/* @@     - No stripping, No date and timestamp */
do
  select
    when message = '@BLANK@' then msgline.msgm = ' ' 
    when word(message,1) = '@' then 
      do 
        message = substr(message,2,length(message)-1)
        msgline.msgm = date() time() message 
      end 
    when substr(message,1,2) = '@@' then 
      do 
        message = substr(message,3,length(message)-2)
        msgline.msgm = message 
      end 
    otherwise msgline.msgm = date() time() strip(message) 
  end
end
/* If a number is provided, add that number of blank lines after */
/* the message */
if msglines > Ø then 
  do msgt=1 to msglines 
    msge = msgt + msgm 
    msgline.msge = ' ' 
  end 
/* Write the contents of the MSGLINE stem to the MSGDD */
call tsotrap "EXECIO * DISKW" msgdd "(STEM MSGLINE. FINIS"
drop msgline. msgb msgt msge 
pull tracelvl . module . sigl . sparms 
call modtrace 'STOP' sigl 
interpret 'trace' tracelvl 
return
/* JOBINFO - Get job related data from control blocks */
/* ITEM - Optional item number desired, default is all */
jobinfo: module = 'JOBINFO'
  if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
call modtrace 'START' sigl
arg item

/* Chase control blocks */
tcb = ptr(540)
ascb = ptr(548)
tiot = ptr(tcb+12)
jscb = ptr(tcb+18)
ssib = ptr(jscb+316)
asid = c2d(stg(ascb+36,2))
jobtype = stg(ssib+12,3)
jobnum = strip(stg(ssib+15,5),'L',Ø)
stepname = stg(tiot+8,8)
procstep = stg(tiot+16,8)
program = stg(jscb+360,8)
jobdata = jobtype jobnum stepname procstep program asid

/* Return job data */
if item <> '' & (datatype(item,'W') = 1) then do
  pull tracelvl . module . sigl . sparms
  call modtrace 'STOP' sigl
  interpret 'trace' tracelvl
  return word(jobdata,item)
end
else do
  pull tracelvl . module . sigl . sparms
  call modtrace 'STOP' sigl
  interpret 'trace' tracelvl
  return jobdata
end

/* ISITUP - Check to see whether an address space is active */
/* ASNAME - Address Space Name */
isitup: module = 'ISITUP'
if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
call modtrace 'START' sigl
arg asname
if asname = '' then call rcexit 86 'Missing Address Space Name'
asexists = 'NO'

/* Accept the address space name to look for */
psa = ptr(16)
asvt = ptr(psa+556)+512
asvtxu = ptr(asvt+4)

/* Chain from the psa to the asvt and run the ascb chain */
do i=Ø to asvtxu - 1
  ascb = stg(asvt+16+i*4,4)
/* Is this an active ascb? */
   if bitand(ascb,'8ØØØØØØØ'x) = 'ØØØØØØØØ'x then
      do
         ascb = c2d(ascb)
         job = stg(ptr(ascb+172),8)
         stc = stg(ptr(ascb+176),8)
      end
      if stc = asname | job = asname then asexists = 'YES'
   end

   pull tracelvl . module . sigl . sparms
   call modtrace 'STOP' sigl
   interpret 'trace' tracelvl
   return asexists

   /* PTR - Pointer to a storage location */
   /* ARG(1) - Storage Address */
   ptr: return c2d(storage(d2x(arg(1)),4))

   /* STG - Return the data from a storage location */
   /* ARG(1) - Location */
   /* ARG(2) - Length */
   stg: return storage(d2x(arg(1)),arg(2))

   /* CPSMERR - Format a CPSM error message for RCEXIT */
   /* CPSMRC - CPSM Return Code */
   /* CPSMMOD - CPSM subroutine issuing the error */
   /* VERB - CPSM API Verb issuing the error */
   /* REASON - CPSM Reason Code */
   /* RESOURCE - CPSM Resource */
   /* RESPONSE - CPSM Response Code */
   /* RESULT - CPSM Result Set */
   cpsmerr: module = 'CPSMERR'
      if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
      parse arg sparms
      push trace() time('L') module 'From:' sigl 'Parms:' sparms
      call modtrace 'START' sigl
      arg cpsmrc cpsmmod verb reason resource response err_result
      if err_result = '' then err_result = Ø
      /* Process obscure conditions */
      if eyuresp(response) <> 'OK' then
         do
            select
               /* Exception list for FEEBACK (add more WHEN clauses as needed) */
               /* Invalid Context or Scope */
               when eyuresp(response) = 'INVALIDPARM' &,
                  eyureas(reason) = 'CONTEXT' then
                  do
                     call msg 'The CONTEXT:' context 'is invalid'
                     call msg 'Correct the value and rerun'
                     if tsoenv = 'BACK' then say
                        cpsmrc = 16
                  end
when eyuresp(response) = 'INVALIDPARM' &,
eyureas(reason) = 'SCOPE' then
do
call msg 'The SCOPE: ' scope ' is invalid'
call msg 'Correct the value and rerun'
if tsoenv = 'BACK' then say
  cpsmrc = 16
end

/* All regions in SCOPE are down */
when eyuresp(response) = 'NOTAVAILABLE' &,
eyureas(reason) = 'SCOPE' then
do
call msg 'It appears like all CICS regions in',
scope ' are down'
call msg 'Confirm CICS regions in scope' scope,
' are up'
if tsoenv = 'BACK' then say
  cpsmrc = 20
end

/* All other INVALIDPARM conditions */
when eyuresp(response) = 'INVALIDPARM' then
do
call msg 'An invalid parm was detected in the'
call msg 'CPSM verb' verb ' used in' cpsmmod
call msg 'The' eyureas(reason) ' is the problem'
if tsoenv = 'BACK' then say
  cpsmrc = 16
end

/* If not an exception, gather the FEEDBACK */
otherwise
do
  signal off novalue
  if tsoenv = 'BACK' then say
    call msg 'Unexpected CPSM' cpsmmod verb 'API',
      'Error, see' msgdd ' output for details'
    call saydd msgdd 0 cpsmmod 'Unexpected CPSM',
    verb 'API Error, collecting CPSM feedback data'
  /* If no ERR_RESULT, then there is a single FEEDBACK record */
  if err_result = 0 then
    call cpsmfdbk err_result
  /* If there is an ERR_RESULT, then loop until NODATA */
  else
    do until eyuresp(feedback_response) = 'NODATA'
      call cpsmfdbk err_result
  /* Invalid GET PARM */
  if eyuresp(feedback_response) = 'INVALIDPARM',
    & eyureas(feedback_reason) = 'RESULT' then
    do
      cpsmrc = 16
      call msg 'Invalid GET PARM: ' getparm
leave  
end  
end  

/* Issue termination message */
msgprefix = cpsmmmod verb
msg = eyureas(reason) resource eyuresp(response)
MAXRC = cpsmrc

/* Terminate the CPSM connection and call rcexit */
call cpsmterm
call rcexit MAXRC msgprefix msg

else
do
   pull tracelvl . module . sigl . sparms
   call modtrace 'STOP' sigl
   interpret 'trace' tracelvl
   return
end  

/* CPSMFDBK - CPSM Feedback command used to collect CPSM error data */
/* FEEDBACK_RESULT - Feedback Result Set */
cpsmfdbk: module = 'CPSMFDBK'
   if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
      parse arg sparms
      push trace() time('L') module 'From:' sigl 'Parms:' sparms
      call modtrace 'START' sigl
/* Accept FEEDBACK_RESULT */
   arg feedback_result
   /* Set the length for the FEEDBACK table (with room for growth) */
   feedback_len = 300
/* If FEEDBACK_RESULT = Ø then use the simple form of FEEDBACK */
   if FEEDBACK_RESULT = Ø then
      FBRC = eyuapi("FEEDBACK",
         "INTO(FEEDBACK_DATA)",
         "LENGTH(FEEDBACK_LEN)",
         "THREAD(CPSM_THREAD)",
         "RESPONSE(FEEDBACK_RESPONSE)",
         "REASON(FEEDBACK_REASON)"
    )
else
   /* If FEEDBACK_RESULT <> Ø then use the RESULT form of FEEDBACK */
   FBRC = eyuapi("FEEDBACK",
      "INTO(FEEDBACK_DATA)",
      "LENGTH(FEEDBACK_LEN)",
      "THREAD(CPSM_THREAD)",
      "RESULT(FEEDBACK_RESULT)",
      "RESPONSE(FEEDBACK_RESPONSE)",
      "REASON(FEEDBACK_REASON)"
    )
/* TPARSE the FEEDBACK record */
   if eyuresp(feedback_response) = 'OK' then
do
TRC = eyuapi("TPARSE",
"OBJECT(Feedback)",
"PREFIX(Feedback)",
"THREAD(CPSM_THREAD)",
"STATUS(TPARSE_RESPONSE)",
"VAR(Feedback_DATA.1)"
if TRC <> Ø | tparse_response <> 'OK' then
call saydd msgdd Ø 'TPARSE error',
'TRC=' trc
'TPARSE_RESPONSE='tparse_response
/* Write a message to the MSGDD if FEEDBACK worked */
call saydd msgdd 0 'FEEDBACK OBJECT:' feedback_object,
'ACTION:' feedback_object_act,
'EIBRESP:' feedback_ceibresp,
'EIBRESP2:' feedback_ceibresp1,
'EIBFN:' feedback_ceibfn,
'RESPONSE:' eyuresp(feedback_response),
'REASON:' eyureas(feedback_reason),
'DIAGNOSTICS:' feedback_diagnostic
end
else
/* Write a message to the MSGDD if there is a FEEDBACK error */
call saydd msgdd 1 cpsmmod object 'FEEDBACK Error:',
'RESPONSE:' eyuresp(feedback_response),
'REASON:' eyureas(feedback_reason)
pull tracelvl . module . sigl . sparms
modtrace 'STOP' sigl
interpret 'trace' tracelvl
return
/* CPSMCMAS - Get CMAS name */
/* N/A - None */
cpsmcmas: module = 'CPSMCMAS'
if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
modtrace 'START' sigl
/* Check if the CMASMAP DD exists */
if listdsi('CMASMAP' 'FILE') = Ø then do
/* If so, the read it and look for a match on SYSNAME and use CMAS */
call tsotrap "EXECIO * DISKR CMASMAP (STEM CMASMAP.FINIS"
do cmm=1 to cmasmmap.Ø
if substr(cmasmmap.cmm,1,1) = '*' then iterate
parse var cmasmmap.cmm cmas1par cmasename .
cmas = 'MISSING'
if cmas1par = MVSVAR('SYSNAME') then do
  cmas = cmasename
leave
end
end
end
else
    /* If not CMASMAP DD, the use the built-in pattern */
do
        cmas = 'C'||mvsvar('SYSCLONE')||'XCMAS'
end
pull tracelvl . module . sigl . sparms
 call modtrace 'STOP' sigl
 interpret 'trace' tracelvl
return cmas

/* CPSMINIT - Initialize a CPSM session */
/* CMAS - CPSM CMAS */
cpsminit: module = 'CPSMINIT'
if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
 call modtrace 'START' sigl
arg cmas
if cmas = '' then cmas = cpsmcmas()
cpsm_ver = 'Ø22Ø'  /* Change as CPSM Version changes */
/* Set TRC=9999 for shutdown check to insure a CPSMTERM is run */
    TRC = 9999
/* Confirm the CMAS is active */
if isitup(cmas) = 'NO' then
call rcexit 86 'CMAS:' cmas 'is not active on' lpar
/* Initialize the CPSM API */
call rcexit eyuinit() 'Error initializing the CPSM REXX API'
/* Connect to a CMAS */
CRC = eyuapi("CONNECT",
               "CONTEXT("cmas")",
               "SCOPE("cmas")",
               "VERSION("cpsm_ver")",
               "THREAD(CPSM_THREAD)",
               "RESPONSE(RESPONSE)",
               "REASON(REASON)")
/* Error processing */
cmasmsg = cmas '(Version' cpsm_ver')'
call rcexit CRC 'Error connecting to' cmasmsg
call cpsmerr 1Ø 'CPSMINIT CONNECT' reason cmas response
if cpsm_thread = Ø then call rcexit 1Ø 'No valid CPSM
Thread'
/* Connected OK */
connmsg = 'Connected to' cmasmsg 'on' lpar
call saydd msgdd Ø connmsg
pull tracelvl . module . sigl . sparms
call modtrace 'STOP' sigl
 interpret 'trace' tracelvl
return cpsm_thread
/* CPSMGET - Get a CPSM Result Set */
/* THREAD - CPSM Thread */
/* CONTEXT - CPSM Context */
/* SCOPE - CPSM Scope */
/* OBJECT - CPSM Object */
/* GETPARM - CPSM GET PARM */
/* FILTER - CPSM Filter */
cpsmget: module = 'CPSMGET'
if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
call modtrace 'START' sigl
arg cpsm_thread context scope object getparm filter
if cpsm_thread = '' then call rcexit 41 'CPSM Thread missing'
if context = '' then call rcexit 42 'CPSM Context is missing'
if scope = '' then call rcexit 43 'CPSM Scope is missing'
if object = '' then call rcexit 44 'CPSM Object is missing'
/* Set up the common parts of the GET command */
getprefix = "GET",
"OBJECT("object")",
"CONTEXT("context")",
"SCOPE("scope")",
"COUNT(GET_COUNT)"
getsuffix = "RESULT(GET_RESULT)",
"THREAD(CPSM_THREAD)",
"RESPONSE(RESPONSE)",
"REASON(REASON)"
/* Determine whether FILTER is used, set length, and build syntax */
getfilter = ''
if filter <> '' then
  do
    if substr(reverse(filter),1,1) <> '.' then
      filter = strip(filter)||'.'
    call saydd msgdd Ø 'CPSMGET' object 'FILTER:' filter
    filter_len = length(filter)
    getfilter = "CRITERIA(FILTER) LENGTH("filter_len")"
  end
/* Determine whether GETPARM is used, set length and build syntax */
getparms = ''
if getparm <> '#' & getparm <> '' then
  do
    if substr(reverse(getparm),1,1) <> '.' then
      getparm = strip(getparm)||'.'
    call saydd msgdd Ø 'CPSMGET' object 'GETPARM:' getparm
    getparm_len = length(getparm)
    getparms = "PARM(GETPARM) PARMLEN("getparm_len")"
  end
/* Assemble and execute the GET command */
getcmd = getprefix getfilter getparms getsuffix
call saydd msgdd Ø 'CPSMGET command:' getcmd
GRC = eyuapi(getcmd)
/* If NODATA is found, continue */  
  if eyuresp(response) = 'NODATA' then  
    nop
/* Error processing */  
  else
    do  
    call rcexit GRC 'GET failed for' object  
    call cpsmerr 45 'CPSMGET GET' reason object response,  
      get_result
  end  
/* Exit with the RESULT ID and count */  
  if get_result = Ø then call rcexit 46 object 'count=Ø'  
  call saydd msgdd Ø object 'GET completed' get_count 'rows'  
  pull tracelvl . module . sigl . sparms  
  call modtrace 'STOP' sigl  
  interpret 'trace' tracelvl  
  return get_result get_count
/* CPSMPOBJ - Perform an action on a CPSM object */  
/* THREAD - CPSM Thread */  
/* CONTEXT - CPSM Context */  
/* SCOPE - CPSM Scope */  
/* OBJECT - CPSM Object */  
/* ACTION - CPSM Object Action */  
/* POPARM - CPSM PERFORM PARM (Set to NONE if not required) */  
/* FILTER - CPSM Filter */  
cpsmpobj: module = 'CPSMPOBJ'  
  if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'  
  parse arg sparms  
  push trace() time('L') module 'From:' sigl 'Parms:' sparms  
  call modtrace 'START' sigl  
  arg cpsm_thread context scope object action poparm filter  
  if cpsm_thread = '' then call rcexit 71 'CPSM Thread missing'  
  if context = '' then call rcexit 72 'CPSM Context is missing'  
  if scope = '' then call rcexit 73 'CPSM Scope is missing'  
  if object = '' then call rcexit 74 'CPSM Object is missing'  
  if poparm = '' then call rcexit 75 'CPSM POParm is missing'  
  if action = '' then call rcexit 76 'CPSM Action is missing'  
  if filter = '' then call rcexit 77 'CPSM Filter is missing'  
/* Build the filter */  
  if substr(reverse(filter),1,1) <> '.' then  
    filter = strip(filter)||'.'  
  call saydd msgdd Ø module object 'FILTER:' filter  
  filter_len = length(filter)
/* Build the parm */  
  if poparm = 'NONE' | poparm = '.' then  
    do  
      poparm = '.'  
      poparm_len = length(poparm)
    end
else
    do
        if substr(reverse(poparm),1,1) <> '.' then
            poparm = strip(poparm)||'. '
        poparm_len = length(poparm)
        call saydd msgdd Ø 'CPSMPOBJ' object 'POPARM:', poparm 'used, length='poparm_len
    end

/* Perform the filtered action on the CPSM object */
pobjcmd = "PERFORM",
    "OBJECT("object")",
    "ACTION("action")",
    "PARM(POPARM)",
    "PARMLEN("poparm_len")",
    "CONTEXT("context")",
    "SCOPE("scope")",
    "CRITERIA(FILTER)",
    "LENGTH("filter_len")",
    "COUNT(POBJ_COUNT)",
    "RESULT(POBJ_RESULT)",
    "THREAD(CPSM_THREAD)",
    "RESPONSE(RESPONSE)",
    "REASON(REASON)"
        call saydd msgdd Ø 'CPSMPOBJ command:' pobjcmd
        PRC = eyuapi(pobjcmd)
/* Error processing */
    if eyuresp(response) = 'NODATA' then
        nop
    else
        do
            call rcexit PRC 'PERFORM OBJECT failed for' object action
            if eyuresp(response) = 'OK' then
                call saydd msgdd Ø object 'PERFORM' action, 'completed' pobj_count 'rows'
            else
                call saydd msgdd Ø object 'PERFORM' action 'failed'
                call cpsmerr 78 'CPSMPOBJ PERFORM' reason object, response pobj_result
            call saydd msgdd Ø 'CPSMPOBJ command:' pobjcmd
            PRC = eyuapi(pobjcmd)
/* Error processing */
        else
            do
                call rcexit PRC 'PERFORM OBJECT failed for' object action
                if eyuresp(response) = 'OK' then
                    call saydd msgdd Ø object 'PERFORM' action, 'completed' pobj_count 'rows'
                else
                    call saydd msgdd Ø object 'PERFORM' action 'failed'
                    call cpsmerr 78 'CPSMPOBJ PERFORM' reason object, response pobj_result
            exit with the RESULT ID and count */
            if pobj_result = Ø then call rcexit 79 object 'count=Ø'
            pull tracelvl . module . sigl . sparms
            call modtrace 'STOP' sigl
            interpret 'trace' tracelvl
            return pobj_result pobj_count
/* CPSMTERM - Terminate a CPSM session */
/* CMAS - CPSM CMAS */
cpsmterm: module = 'CPSMTERM'
if wordpos(module,probe) <> Ø then trace 'r'; else trace 'n'
parse arg sparms
push trace() time('L') module 'From:' sigl 'Parms:' sparms
    call modtrace 'START' sigl
    arg cmas
    if cmas = '' then cmas = cpsmcmas()
    TRC = eyuapi("TERMINATE",
                "RESPONSE(RESPONSE)",
                "REASON(REASON)")
    call rcexit TRC 'CPM Terminate error'

    /* Free the CPSM function package */
    call rcexit eyuterm() 'Error terminating the CPSM REXX API'
    termmsg = 'Disconnected from' cmasmsg 'on' mvsvar('SYSNAME')
    call saydd msgdd Ø termmsg
    pull tracelvl . module . sigl . sparms
    call modtrace 'STOP' sigl
    interpret 'trace' tracelvl
    return TRC

    /* MODTRACE - Module Trace */
    /* TRACETYP - Type of trace entry */
    /* SIGLINE - The line number called from */
    modtrace: if modtrace = 'NO' then return
    arg tracetyp sigline
    tracetyp = left(tracetyp,5)
    sigline = left(sigline,5)

    /* Adjust MODSPACE for START */
    if tracetyp = 'START' then
        modspace = substr(modspace,1,length(modspace)+1)

    /* Set the trace entry */
    traceline = modspace time('L') tracetyp module sigline sparms

    /* Adjust MODSPACE for STOP */
    if tracetyp = 'STOP' then
        modspace = substr(modspace,1,length(modspace)-1)

    /* Determine where to write the traceline */
    if ispfenv = 'YES' & tsoenv = 'FORE' then
        do
            zedlmsg = traceline
            address ISPEXEC "LOG MSG(ISRZØØØ)"
        end
    else
        say traceline

    /* SAY to SYSTSPRT */
    return

Robert Zenuk
Systems Programmer (USA) © Xephon 2004
EXCI batch client control program

INTRODUCTION
The External CICS Interface (EXCI) was introduced with CICS for MVS/ESA Version 4 Release 1. EXCI was specifically written for communication between MVS and CICS. EXCI allows MVS client programs to allocate and open sessions (known as pipes) to a specific CICS region, and then to pass Distributed Program Link (DPL) requests (through the pipe) to that CICS region. The CICS Multi-Region Operation (MRO) facility of the CICS Inter-Region Communication (IRC) facility supports these requests. In CICS terms, each EXCI pipe maps onto one MRO session. EXCI can also be used within the same mainframe and in a sysplex for mainframe-to-mainframe communication within the sysplex using cross-system MRO (XCF/MRO) support. XCF/MRO is an extremely fast cross-memory access method.

TWO INTERFACES
EXCI has two interfaces – EXCI CALL and EXEC CICS.
The EXCI CALL interface consists of six commands that open and allocate sessions to CICS, issue DPL requests using those sessions, and subsequently close and deallocate the sessions to CICS. The six commands are:

- Initialise_User
- Allocate_Pipe
- Open_Pipe
- DPL_Call
- Close_Pipe
- Deallocate_Pipe.
The EXEC CICS interface provides a single composite command, which performs all six commands of the EXCI CALL interface – EXEC CICS LINK PROGRAM.

The EXEC CICS interface is obviously the easiest to program and less prone to programming errors because it is a single composite command. However, it is far less flexible and provides much poorer performance if multiple DPL requests are to be processed, because each EXEC CICS LINK PROGRAM command results in all six commands of the EXCI CALL interface being processed. The EXCI CALL interface is far more difficult to program, but, once the first three commands have been processed successfully, any number of DPL requests can be processed before closing and deallocating the session to CICS – offering far better performance.

THE BEST OF BOTH WORLDS

I effectively wanted what was, in my opinion, the ‘best of both worlds’. The flexibility and performance advantages for multiple DPL calls of the EXCI CALL interface, and the simplicity of the EXEC CICS interface – so that client programs could concentrate on their business function without having to worry too much about the EXCI environment.

Additionally, I also had to consider that CICS application programmers were not necessarily batch or MVS application programmers. Therefore, the intention was to provide a solution that was as typically CICS COMMAREA driven as possible and which would also enable the application programmers to concentrate solely on the DPL call.

THE SOLUTION

The solution was an EXCI batch client control program (CM420), which processed the Initialize_User, Allocate_Pipe, and Open_Pipe commands before linking to a specified EXCI client program that processed single or multiple DPL_Call commands. The EXCI control program (CM420) then
processed the Close_Pipe and Deallocate_Pipe commands before terminating.

An EXCI client program would therefore be able to concentrate on the DPL_Call(s), ie its own processing, while CM420 takes care of the rest. A COMMAREA is used for communication between CM420 and the EXCI client program.

THE PROGRAMS

All programs are written in Assembler. However, an EXCI client program could be written in any of the languages supported (ie C, COBOL, or PL/I).

- CM420 – EXCI batch client control program
- CM420A01 – Assembler copybook COMMAREA CM420COM
- CM419 – I/O sub-program
- CM419A01 – Assembler copybook COMMAREA CM419COM.

CM420

CM420 provides the processing for all the EXCI CALL commands except DPL_Call. It also provides a COMMAREA, CM420COM, containing all the information (variables, diagnostic area, etc) required by EXCI client programs. The PARM field of the job step must contain the name of the EXCI client program to be linked to and the VTAM APPLID of the CICS server region to be connected to:

```plaintext
//* ================================================================= *
//* EXECUTE AN EXTERNAL CICS INTERFACE (EXCI) CLIENT PROGRAM          *
//* ================================================================= *
//STEP1    EXEC PGM=CM420,                                           
//         PARM='CM412,TESTCICS'                                   
//STEPLIB  DD   DISP=SHR,DSN=CWM.CICSTS.TEST.LOAD                  
//         DD   DISP=SHR,DSN=CWM.CICSTS.TEST.SDFHEXCI              
//SYSPRINT DD   SYSOUT=*                                         
//SYSUDUMP DD   SYSOUT=*                                         
//SYSIN    DD   SYSOUT=*                                         
```

*COMMAND  +CONTINUATION  *
DELETE ALL(*) GROUP(CWMTEST)
*
DEFINE TRANCLASS(CWM1) GROUP(CWMTEST)
    MAXACTIVE(10)
*/

The following CICS resources must also be defined in the CSD and installed in the CICS server region:

DEFINE CONNECTION(M420) GROUP(CWMSYS)
 DESCRIPTION(Specific Connection - CM420)
    NETNAME(CM420) ACCESSMETHOD(IRC)  PROTOCOL(EXCI)
    CONNTYPE(SPECIFIC) SINGLESESS(NO) DATASTREAM(USER)
    RECORDFORMAT(U) QUEUELIMIT(NO) AUTOCONNECT(NO)
    INSERVICE(YES) ATTACHSEC(IDENTIFY) BINDSECURITY(NO)
    USEDFLTUSER(NO) XLNACTION(KEEP)
*
DEFINE SESSIONS(M420) GROUP(CWMSYS)
 DESCRIPTION(Specific Sessions - CM420)
    CONNECTION(M420) MAXIMUM(0,0) RECEIVEPFX(CM)
    RECEIVECOUNT(4) SENDSIZE(4096) RECEIVESIZE(4096) SESSPRIORITY(0)
    AUTOCONNECT(NO) BUILDCHAIN(YES) USERAREALEN(0)
    IOAREALEN(4096,4096) RELREQ(NO) DISCREQ(NO) NEPCLASS(NO)
    RECOVOPTION(SYSDEFAULT)
*
DEFINE TRANSACTION(M420) GROUP(CWMSYS)
 DESCRIPTION('CM420 EXCI Server Transaction')
    PROGRAM(DFHMIRS) TWASIZE(0) PROFILE(DFHCICSA) STATUS(ENABLED)
    TASKDATALOC(BELOW) TASKDATAKEY(USER) STORAGECLEAR(NO)
    RUNAWAY(SYSTEM) SHUTDOWN(DISABLED) ISOLATE(YES) DYNAMIC(NO)
    ROUTABLE(NO) TRANCLASS(DFHTCL00) DTIMOUT(10)
    RESTART(NO) SPURGE(YES) TPURGE(YES) DUMP(YES) TRACE(YES)
    CONFDATA(YES) ACTION(BACKOUT) WAIT(YES) WAITTIME(0,0,0)
    RESSEC(NO) CMDSEC(NO)

If you change any of the resource names you will have to make appropriate changes in the CM420 source code.

CM420 not only provides a COMMAREA for EXCI processing but also loads a batch I/O sub-program (CM419) and provides a COMMAREA (CM419COM) for that program. CM420 uses CM419 only for writing messages to SYSPRINT.

CM420 uses the IBM-supplied copybooks DFHXCRCD and DFHXCPLD for return codes and parameter list equates.
respectively. If any EXCI warnings or errors are detected, CM420 writes diagnostic messages to SYSPRINT containing the response, reason, subreason-1 and subreason-2 codes, all converted into displayable format as follows:

CM420401I OPEN_PIPE processing..
CM420010E Error processing EXCI call, diagnostics follow:-
CM420010E EXCI Response . . . : 00000008
CM420010E EXCI Reason . . . : 00000203
CM420010E EXCI Subreason-1 . : 0000092
CM420010E EXCI Subreason-2 . : 00000000

The above codes could be checked in the CICS External Interfaces Guide to establish that this was a retryable error during processing of the Open_Pipe command because the target CICS region was not available or, more specifically, because the region was not logged on to IRC.

CM420 writes several messages to SYSPRINT regarding its own progress and also information that could be useful if a dump is produced before linking to the EXCI client program specified.

CM420000I *CM420 -CM4200001 01/30/04 09.54*
CM420002I Program CM420 loaded at address X'80007138'.
CM420003I WORKING storage address X'00006BF8' length 00000856 bytes.
CM420003I CM420COM storage address X'00006EA8' length 0000160 bytes.
CM420002I Program CM419 loaded at address X'80008D10'.
CM420003I CM419COM storage address X'00006C80' length 0000548 bytes.
CM420101I PARM Field - Program=CM412 Region=TESTCICS.
CM420201I INITIALIZE_USER processing....
CM420201I INITIALIZE_USER successful.
CM420301I ALLOCATE_PIPE processing....
CM420302I ALLOCATE_PIPE successful.
CM420401I OPEN_PIPE processing....
CM420402I OPEN_PIPE successful.
CM420001I Loading program CM412 ....
CM420002I Program CM412 loaded at address X'800080C8'.
CM420501I LINKing to sub-program CM412 ...
CM420502I Control returned from sub-program CM412    RC=00000000

CM420601I CLOSE_PIPE processing....
CM420602I CLOSE_PIPE successful.

CM420701I DEALLOCATE_PIPE processing....
CM420702I DEALLOCATE_PIPE successful.

CM420901I Program Terminated. Highest Return Code 00000000

An EXCI client program linked to by CM420 must return control to CM420 for clean-up processing of the EXCI environment.

CM420 and all EXCI client programs must include the CICS-supplied EXCI program stub DFHXCSTB. You can use the CICS-supplied procedure DFHEXTAL to assemble and link-edit CM420 and all EXCI client programs written in Assembler. Procedures are also supplied for EXCI client programs written in other languages.

*ASM XOPTS(EXCI)
CM420    TITLE 'CM420 : EXCI BATCH CLIENT CONTROL'
***********************************************************************
*
*                  C A R L   W A D E   M C B U R N I E                  *
*                   -  I T   C O N S U L T A N T  -                   *
*                            www.cwmit.com                            *
***********************************************************************
*
* MODULE NAME = CM420                                                  *
* MODULE TYPE = CSECT (Main Program)                                   *
* DESCRIPTION = CICS EXCI Batch Client Control Program                 *
*                JCL PARM='sub-program,cics-region'                     *
*                This program is more or less a utility for             *
*                enabling EXCI DPL calls using a sub-program,           *
*                which must be specified in the JCL PARM.               *
*                Different sub-programs can, therefore, be             *
*                used to perform different functions.                   *
*                The program performs the following:                    *
*                1. Obtains and validates the JCL PARM=               *
*                2. Establishes an EXCI connection with the             *
*                CICS region specified in PARM=                        *
*                Initialize_User                                      *
*                Allocate_Pipe                                       *
*                Open_Pipe                                           *
***********************************************************************

3. Calls the sub-program specified in PARM= which then processes the DPL call.
4. Terminates the EXCI connection with the CICS region specified in PARM=
   Close_Pipe
   Deallocate_Pipe
6. Closes SYSPRINT and terminates.
This program must be loaded below 16MB RMODE(24) because of the I/O macros used. However, it
must run in 31-bit addressing mode AMODE(31) for EXCI.
***********

* CHANGE HISTORY:
* ---------------
* ***********

** REGISTER EQUATES **
** -------- -------- ----------------------------------------------- **
** REG Ø RØ Work Register **
** REG 1 R1 Work Register **
** REG 2 R2 DSECT - CM42ØCOM **
** REG 3 BASE Base Register for CSECT CM42Ø **
** REG 4 R4 Work Register **
** REG 5 R5 Work Register **
** REG 6 R6 Work Register **
** REG 7 R7 Work Register **
** REG 8 R8 DSECT - CM419COM **
** REG 9 R9 DSECT - EXCI_RETURN_CODE **
** REG 10 R10 **
** REG 11 R11 **
** REG 12 R12 **
** REG 13 DYNREG DSECT - DFHEISTG **
** REG 14 R14 Linkage **
** REG 15 R15 Linkage and some return codes **
***********

** COPYbooks **
** ----------------------------------------------- **
** COPY DFHXCRCD EXCI DSECTS AND RETURN CODES **
** COPY DFHXCLPLD EXCI PARAMETER LIST EQUATES **
** COPY CM419AØ1 DSECT - CM419COM (COMMAREA) **
** COPY CM42ØAØ1 DSECT - CM42ØCOM (COMMAREA) **
***********

* Addressability to DFHEISTG will be established by CICS EXCI. **
*--------------------------------------------------------------------*
DFHEISTG                      CICS EXCI DYNAMIC STORAGE
EJECT
*--------------------------------------------------------------------*
* CM42Ø Dynamic Storage - Start                                       *
*--------------------------------------------------------------------*
DYNSTOR DS $0H                  CM42Ø DYNAMIC STORAGE
JCLPARMA DS A                   --> JCL PARMLIST ------!
PARMLEN DS H                   LENGTH OF PARMFIELD <---!
PARMFLDA DS A                  --> PARM FIELD
EXCIPROG DS CL8                EXCI SUB-PROGRAM NAME
EXCIAPPL DS CL8                APPLID OF CICS REGION
EPLOC DS CL8                   PROGRAM NAME FOR LOAD
LOADPT DS F                    LOAD POINT ADDRESS
HIGH_RC DS F                   HIGHEST RETURN CODE
CURR_RC DS F                   CURRENT RETURN CODE
SUB_RC DS F                    SUB PROGRAM RETURN CODE
WORKØ5 DS CL5                  WORK AREA 5 BYTES
WORKØ9 DS CL9                  WORK AREA 9 BYTES
WORKDW_1 DS D                  WORK AREA DOUBLE WORD
WORKDW_2 DS D                  WORK AREA DOUBLE WORD
A1ØØSR14 DS F                  SAVE REGISTER 14
A2ØØSR14 DS F                  SAVE REGISTER 14
A3ØØSR14 DS F                  SAVE REGISTER 14
A4ØØSR14 DS F                  SAVE REGISTER 14
A5ØØSR14 DS F                  SAVE REGISTER 14
A6ØØSR14 DS F                  SAVE REGISTER 14
A7ØØSR14 DS F                  SAVE REGISTER 14
A9ØØSR14 DS F                  SAVE REGISTER 14
Z1ØØSR14 DS F                  SAVE REGISTER 14
Z2ØØSR14 DS F                  SAVE REGISTER 14
Z3ØØSR14 DS F                  SAVE REGISTER 14
Z4ØØSR14 DS F                  SAVE REGISTER 14
DS $0D                     ALIGN STORAGE
CM419ST DS CL(CM419COM_LENGTH)  STORAGE FOR CM419COM
CM42ØST DS CL(CM42ØCOM_LENGTH)  STORAGE FOR CM42ØCOM
LINK_PL CALL ,                 X
                  (CM419ST,  X
                   CM42ØST),   X
                  VL,         X
                  MF=L
DYNSTORL EQU *-DYNSTOR        LENGTH OF DYNAMIC STORAGE
*--------------------------------------------------------------------*
* CM42Ø Dynamic Storage - End                                           *
*--------------------------------------------------------------------*
EJECT
*--------------------------------------------------------------------*
* Register Equates                                                       *
*--------------------------------------------------------------------*
DFHREGS                      CICS STANDARD EQUATES
BASE    EQU   3                      BASE CODE REGISTER
DYNREG  EQU   13                     DYNAMIC STORAGE REGISTER
EJECT
***********************************************************************
*======================================================================*
*=       E  N  T  R  Y          P  O  I  N  T                        *=
*======================================================================*
***********************************************************************
CM42Ø  DFHEIENT  CODEREG=(BASE),DATAREG=(DYNREG)
CM42Ø  AMODE  31
CM42Ø  RMODE  24
***********************************************************************
*B     A000_MAINLINE  BRANCH OVER EYE-CATCHERS
ASMEYE  DC    C'*'                    ASTERISK
ASMPROG  DC    C'CM42Ø   '             PROGRAM NAME
DC    C'-'                    HYPHEN
ASMLVL  DC    C'CWM00001'             PROGRAM LEVEL
DC    C' '                    BLANK
ASMDATE  DC    C'&SYSDATE'             DATE OF ASSEMBLY
DC    C' '                    BLANK
ASMTIME  DC    C'&SYSTIME'             TIME OF ASSEMBLY
DC    C'*'                    ASTERISK
ASMEYEL  EQU   *-ASMEYE                LENGTH OF EYE-CATCHER
EJECT
***********************************************************************
*---------------------------------------------------------------------*
*- AØØØ MAINLINE : Controls the flow of the program                 -*
*---------------------------------------------------------------------*
*---------------------------------------------------------------------*
AØØØ_MAINLINE  DS  ØH
BAL   R14,A1ØØ_INITIALIZE     PERFORM INITIALIZATION
CLC   HIGH_RC,=F'4'             IF RC > 4
BH    AØØØTERM                THEN TERMINATE
BAL   R14,A2ØØ_INIT_USER       EXCI INITIALIZE_USER
CLC   HIGH_RC,=F'4'             IF RC > 4
BH    AØØØTERM                THEN TERMINATE
BAL   R14,A3ØØ_ALLOC_PIPE      EXCI ALLOCATE_PIPE
CLC   HIGH_RC,=F'4'             IF RC > 4
BH    AØØØTERM                THEN TERMINATE
BAL   R14,A4ØØ_OPEN_PIPE       EXCI OPEN_PIPE
CLC   HIGH_RC,=F'4'             IF RC > 4
BH    AØØØTERM                THEN TERMINATE
BAL R14,A500_SUB_PROG EXCI SUB PROGRAM
BAL R14,A600_CLOSE_PIPE EXCI CLOSE_PIPE
BAL R14,A700_DEALLOC_PIPE EXCI DEALLOCATE_PIPE
A000TERM BAL R14,A900_TERMINATION PERFORM TERMINATION

***********************************************************************
*--------------------------------------------------------------------*
*- A Ø Ø Ø _ R E T U R N :  Return Control                          -*
*--------------------------------------------------------------------*
***********************************************************************
AØØØ_RETURN  DS ØH
L R15,HIGH_RC LOAD HIGHEST RETURN CODE
RETURN DFHEIRET RCREG=(15)

***********************************************************************
*=====================================================================*
*=       E X I T             P O I N T                        =*
*=====================================================================*
***********************************************************************
EJECT

***********************************************************************
*--------------------------------------------------------------------*
*- A 1 Ø Ø _ I N I T I A L I Z E :  Perform Initialization          -*
*--------------------------------------------------------------------*
***********************************************************************
A1ØØ_INITIALIZE  DS ØH

*--------------------------------------------------------------------*
*- Clear DYNSTOR.                                                   -*
*--------------------------------------------------------------------*
LA R4,DYNSTOR ADDRESS DYNSTOR
LA R5,DYNSTORL LENGTH OF DYNSTOR
XR R6,R6 FROM ADDRESS NOT REQUIRED
XR R7,R7 SET LENGTH TO Ø
ICM R7,8,=C' ' SET PADDING TO BLANKS
MVCL R4,R6 CLEAR STORAGE TO BLANKS
ST R1,JCLPARMA SAVE PARMLIST PTR FOR LATER
ST R14,A1ØØSR14 SAVE REGISTER 14

*--------------------------------------------------------------------*
*- Establish addressability and map CM42ØCOM, CM419COM, and         -*
**EXCI_RETURN_CODE.**

*--------------------------------------------------------------------*
| LA R2,CM420ST  ADDRESS CM420COM |
| USING CM420COM,R2  MAP CM420COM |
| MVC CM420COM_EYECATCH,='CM420COM'  EYECATCHER |
| XC HIGH_RC,HIGH_RC  SET HIGHEST RC TO 0 |
| XC CURR_RC,CURR_RC  SET CURRENT RC TO 0 |
| LA R8,CM419ST  ADDRESS CM419COM |
| USING CM419COM,R8  MAP CM419COM |
| MVC CM419COM_EYECATCH,='CM419COM'  EYECATCHER |
| MVI CM419COM_SYSPRINT_SI,CM419COM_CLOSE  CLOSED STATUS |
| MVI CM419COM_SYSIN_SI,CM419COM_CLOSE  CLOSED STATUS |
| LA R9,CM420COM_RETURN_AREA  RETURN_AREA STORAGE |
| USING EXCI_RETURN_CODE,R9  MAP RETURN_AREA |

*--------------------------------------------------------------------*

* LOAD the I/O sub-program CM419. If the load fails the program will abend. *

*--------------------------------------------------------------------*

A100L419 DS 0H

MVC EPLOC,='CM419'  MOVE SUB PROGRAM NAME FOR LOAD

A100LOAD LOAD EPLOC=EPLOC

ST R0,LOADPT  SAVE EP ADDRESS

*--------------------------------------------------------------------*

* Write start message using program eyecatcher as text. *

*--------------------------------------------------------------------*

A100MSG DS 0H

MVC CM419COM_SYSPRINT_MSG,='CM4200001'  MOVE MSG NO.

MVC CM419COM_SYSPRINT_DATA(ASMEYEL),ASMEYE  MOVE EYECATCHER

BAL R14,Z200_WRITE_SYSPRINT  WRITE MSG TO SYSPRINT

BAL R14,Z200_WRITE_SYSPRINT  BLANK LINE

*--------------------------------------------------------------------*

* Issue CM420 loaded message. Convert the EP address to displayable characters. *

*--------------------------------------------------------------------*

A100420P DS 0H  ISSUE MESSAGE

MVC CM419COM_SYSPRINT_MSG,='CM420002I'  MOVE MSG NO.

MVC CM419COM_SYSPRINT_DATA(L'CM420002I'),CM420002I & TEXT

MVC CM419COM_SYSPRINT_DATA+8(8),='CM420'  & PGM

ST BASE,WORK05  EPA IN WORK FIELD

UNPK WORK09,WORK05  UNPACK ADDRESS

MVC WORKDW_1(L'WORKDW_1'),WORK09  MOVE REQ. BYTES (8)

TR WORKDW_1,TRANTA0-240  TRANSLATE REQ. BYTES

MVC CM419COM_SYSPRINT_DATA+37(L'WORKDW_1'),WORKDW_1  MOVE

BAL R14,Z200_WRITE_SYSPRINT  WRITE MSG TO SYSPRINT

*--------------------------------------------------------------------*

* Issue WORKING storage message. Convert the address and length to displayable characters. *

*--------------------------------------------------------------------*

A100WORK DS 0H  ISSUE MESSAGE

MVC CM419COM_SYSPRINT_MSG,='CM420003I'  MOVE MSG NO.
MVC CM419COM_SYSPRINT_DATA(L'CM420003I),CM420003I & TEXT
MVC CM419COM_SYSPRINT_DATA(8),=C'WORKING' & TEXT
LA R4,DYNSTOR --> DYNSTOR
ST R4,WORK05 SAVE IN WORK FIELD
UNPK WORK09,WORK05 UNPACK ADDRESS
MVC WORKDW_1(L'WORKDW_1),WORK09 MOVE REQ. BYTES (8)
TR WORKDW_1,TRANTAB0-240 TRANSLATE REQ. BYTES
MVC CM419COM_SYSPRINT_DATA+27(L'WORKDW_1),WORKDW_1 MOVE
LA R4,DYNSTORL LOAD LENGTH OF DYNSTOR
CVD R4,WORKDW_1 CONVERT TO DECIMAL
UNPK WORKDW_2,WORKDW_1 CONVERT TO...
OI WORKDW_2+7,X'F0' ... DISPLAYABLE DECIMAL
MVC CM419COM_SYSPRINT_DATA+44(L'WORKDW_2),WORKDW_2 MOVE
BAL R14,Z200_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

*--------------------------------------------------------------------*
*- Issue CM420COM storage message. Convert the address and length to-*
*- displayable characters.                                          -*
*--------------------------------------------------------------------*
A100420S DS 0H ISSUE MESSAGE
MVC CM419COM_SYSPRINT_MSG,=C'CM420003I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM420003I),CM420003I & TEXT
MVC CM419COM_SYSPRINT_DATA(8),=C'CM420COM' & TEXT
LA R4,CM420ST --> CM420COM STORAGE
ST R4,WORK05 SAVE IN WORK FIELD
UNPK WORK09,WORK05 UNPACK ADDRESS
MVC WORKDW_1(L'WORKDW_1),WORK09 MOVE REQ. BYTES (8)
TR WORKDW_1,TRANTAB0-240 TRANSLATE REQ. BYTES
MVC CM419COM_SYSPRINT_DATA+27(L'WORKDW_1),WORKDW_1 MOVE
LA R4,CM420COM_LENGTH LOAD LENGTH OF CM420COM
CVD R4,WORKDW_1 CONVERT TO DECIMAL
UNPK WORKDW_2,WORKDW_1 CONVERT TO...
OI WORKDW_2+7,X'F0' ... DISPLAYABLE DECIMAL
MVC CM419COM_SYSPRINT_DATA+44(L'WORKDW_2),WORKDW_2 MOVE
BAL R14,Z200_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

*--------------------------------------------------------------------*
*- Issue CM419 loaded message. Convert the LOADPT address to  -*
*- displayable characters.                                          -*
*--------------------------------------------------------------------*
A100419P DS 0H ISSUE MESSAGE
BAL R14,Z200_WRITE_SYSPRINT BLANK LINE
MVC CM419COM_SYSPRINT_MSG,=C'CM420002I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM420002I),CM420002I & TEXT
MVC CM419COM_SYSPRINT_DATA+8(L'EPLOC),EPLOC OVERLAY PGM
MVC WORK05(L'LOADPT),LOADPT ADDITIONAL BYTE AVOIDS SWAP
UNPK WORK09,WORK05 UNPACK ADDRESS
MVC WORKDW_1(L'WORKDW_1),WORK09 MOVE REQ. BYTES (8)
TR WORKDW_1,TRANTAB0-240 TRANSLATE REQ. BYTES
MVC CM419COM_SYSPRINT_DATA+37(L'WORKDW_1),WORKDW_1 MOVE
BAL R14,Z200_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

*--------------------------------------------------------------------*
* Issue CM419COM storage message. Convert the address and length to displayable characters.

A10419S DS ØH
  ISSUE MESSAGE
  MVC CM419COM_SYSPRINT_MSG,=C'CM420003I' MOVE MSG. NO.
  MVC CM419COM_SYSPRINT_DATA(L'CM420003I),CM420003I & TEXT
  MVC CM419COM_SYSPRINT_DATA(8),=C'CM419COM' & TEXT
  LA R4,CM419ST --> CM419COM STORAGE
  ST R4,WORKØ5 SAVE IN WORK FIELD
  UNPK WORKØ9,WORKØ5 UNPACK ADDRESS
  MVC WORKDW_1(L'WORKDW_1),WORKØ9 MOVE REQ. BYTES (8)
  TR WORKDW_1,TRANTABØ-24Ø TRANSLATE REQ. BYTES
  MVC CM419COM_SYSPRINT_DATA+27(L'WORKDW_1),WORKDW_1 MOVE
  LA R4,CM419COM_LENGTH LOAD LENGTH OF CM419COM
  CVD R4,WORKDW_1 CONVERT TO DECIMAL
  UNPK WORKDW_2,WORKDW_1 CONVERT TO ...
  OI WORKDW_2+7,X'FØ' ... DISPLAYABLE DECIMAL
  MVC CM419COM_SYSPRINT_DATA+44(L'WORKDW_2),WORKDW_2 MOVE
  BAL R14,Z200_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

* Obtain JCL input parameters, two positional parameters, separated by commas, are expected:

A100PARM DS ØH
  BAL R14,Z200_WRITE_SYSPRINT BLANK LINE
  L R1,JCLPARMA --> JCL PARMLIST ADDRESS
  L R1,Ø(,R1) --> JCL PARMLIST
  XR R6,R6 CLEAR R6
  ICM R6,B'ØØ11',Ø(R1) WAS A PARM FIELD PASSED ?
  BZ A100PMSG NO - PROCESS ERROR
  STH R6,PARMLEN YES - SAVE LENGTH
  LA R1,2(R1) --> PARM FIELD
  ST R1,PARMFLDA SAVE PARM FIELD ADDRESS
  AR R6,R1 FIRST BYTE AFTER PARM FIELD
  LR R5,R6 SAVE FOR FURTHER SEARCHES
  LA RØ,X'6B' SET SEARCH FOR COMMA
  A100SRØ1 SRST R6,R1 SEARCH PARMS FOR COMMA
  BC 1,A100SRØ1 CPU CONDITION CONTINUE SEARCH
  BC 2,A100PMSG NOT FOUND - BRANCH FOR ERROR
  LR R7,R6 COMMA FOUND - SAVE ADDRESS
  SR R6,R1 LENGTH OF PARM
  LR R4,R6 SAVE LENGTH
  C R6,=F'8' IF LENGTH > 8
  BH A100PMSG THEN PARM INVALID
  S R6,=F'1' SUB. 1 FOR MOVE
  EX R6,GETPROG EXECUTE MOVE
  B A100NXØ1 NEXT SEARCH
  GETPROG MVC EXCIPROG(Ø),Ø(R1) MOVE EXCI PROGRAM NAME
A1ØØNXØ1 DS ØH
LA R1,1(R7) BUMP AFTER "," FOR SEARCH
LR R6,R5 FIRST BYTE AFTER PARM FIELD
LA RØ,X'6B' SET SEARCH FOR COMMA
A1ØØSRØ2 SRST R6,R1 SEARCH PARM FOR COMMA
BC 1,A1ØØSRØ2 CPU CONDITION CONTINUE SEARCH
BC 4,A1ØØPSMG COMMA FOUND = TOO MANY PARMs!
* NO COMMA THEN LAST PARM
A1ØØLP SR R6,R1 LENGTH OF PARM
C R6,=F'8' IF LENGTH > 8
BH A1ØØPSMG THEN PARM INVALID
S R6,=F'1' SUB. 1 FOR MOVE
EX R6,GETAPPL EXECUTE MOVE
B A1ØØSRØK FINISHED SEARCHES
GETAPPL MVC EXCIAPPL(Ø),Ø(R1) MOVE EXCI APPLID
A1ØØSRØK DS ØH SEARCHES AND PARMs OK
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø1Ø1I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø1Ø1I),CM42Ø1Ø1I & TEXT
MVC CM419COM_SYSPRINT_DATA+21(L'RECIPIENT),RECIPIENT & PROG
MVC CM419COM_SYSPRINT_DATA+37(L'RECIPIENT),RECIPIENT & APPL
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
B A1ØØRET RETURN
A1ØØPSMG DS ØH INVALID OR NO INPUT PARMs.
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø1Ø2E' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø1Ø2E),CM42Ø1Ø2E & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
MVC CURR_RC,=F'8' RC=Ø8
B A1ØØRET RETURN TO CALLER

EJECT

***********************************************************************
*--------------------------------------------------------------------*
* A 2 Ø Ø _ I N I T _ U S E R : EXCI Initialize_User              *
*--------------------------------------------------------------------*
*--------------------------------------------------------------------*
*--------------------------------------------------------------------*

A2ØØ_INIT_USER DS ØH
ST R14,A2ØØSR14 SAVE REGISTER 14
XC CURR_RC,CURR_RC CLEAR CURRENT RC
BAL R14,Z2ØØ_WRITE_SYSPRINT BLANK LINE

*- Write message for Initialize_User call. -*

MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø2Ø1I'     MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø2Ø1I),CM42Ø2Ø1I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT  WRITE MSG TO SYSPRINT

*- Call Initialize_User -*

MVC CM42ØCOM_VERSION_NUM,=AL4(VERSION_1)  SET VERSION
MVC CM42ØCOM_CALL_TYPE,=AL4(INIT_USER)    SET CALL TYPE
MVC CM42ØCOM_USER_NAME,=C'CM42Ø   '       SET APPLICATION
A2ØØCALL CALL  DFHXCIS,                                                X
  (CM42ØCOM_VERSION_NUM,                                  X
   CM42ØCOM_CALL_TYPE,                                     X
   CM42ØCOM_USER_NAME),                                    X
   VL,                                                     X
   MF=(E,CM42ØCOM_PL)

*- Check the response code and produce diagnostics if required -*

A2ØØCHCK CLC EXCI_RESPONSE,=AL4(EXCI_NORMAL)  IF NORMAL RESPONSE
  BE A2ØØOK THEN OK MESSAGE
  BAL R14,Z3ØØ_EXCI_DIAGNOSTICS ELSE DIAGNOSTICS
  MVC CURR_RC,EXCI_RESPONSE                   SET RC
  B A2ØØRET AND RETURN.

A2ØØOK MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø2Ø2I'     MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø2Ø2I),CM42Ø2Ø2I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT  WRITE MSG TO SYSPRINT

*- Return to caller -*

A2ØØRET BAL R14,Z4ØØ_CHECK_RC       CHECK HIGHEST RC
  L R14,A2ØØSR14 RESTORE REGISTER 14
  BR R14 RETURN TO CALLER

***********************************************************************

*= A 3 Ø Ø _ A L L O C _ P I P E : EXCI Allocate_Pipe *=

***********************************************************************
A3ØØ_ALLOC_PIPE DS ØH
ST R14,A3ØØSR14 SAVE REGISTER 14
XC CURR_RC,CURR_RC CLEAR CURRENT RC
BAL R14,Z2ØØ_WRITE_SYSPRINT BLANK LINE
*--------------------------------------------------------------------*
* Write message for Allocate_Pipe call.                           *
*--------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø3Ø1I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø3Ø1I),CM42Ø3Ø1I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
*--------------------------------------------------------------------*
* Call Allocate_Pipe                                               *
*--------------------------------------------------------------------*
MVC CM42ØCOM_CALL_TYPE,=AL4(ALLOCATE_PIPE) CALL TYPE
MVC CM42ØCOM_OPTIONS,=AL1(SPECIFIC_PIPE) OPTIONS
MVC CM42ØCOM_CICS_APPL,EXCIAPPL APPLID
A3ØØCALL CALL DFHXCIS, X
(CM42ØCOM_VERSION_NUM, X
CM42ØCOM_RETURN_AREA, X
CM42ØCOM_USER_TOKEN, X
CM42ØCOM_CALL_TYPE, X
CM42ØCOMPIPE_TOKEN, X
CM42ØCOM_CICS_APPL, X
CM42ØCOMOPTIONS), X
VL, X
MF=(E,CM42ØCOM_PL)
*--------------------------------------------------------------------*
* Check the response code and produce diagnostics if required      *
*--------------------------------------------------------------------*
A3ØØCHK CLC EXCI_RESPONSE,=AL4(EXCI_NORMAL) IF NORMAL RESPONSE
BE A3ØØOK THEN OK MESSAGE
BAL R14,Z3ØØ_EXCI_DIAGNOSTICS ELSE DIAGNOSTICS
MVC CURR_RC,EXCI_RESPONSE SET RC
B A3ØØRET AND RETURN.
A3ØØOK MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø3Ø2I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø3Ø2I),CM42Ø3Ø2I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
*--------------------------------------------------------------------*
* Return to caller                                                 *
*--------------------------------------------------------------------*
A3ØØRET BAL R14,Z4ØØ_CHECK_RC CHECK HIGHEST RC
L R14,A3ØØSR14 RESTORE REGISTER 14
BR R14 RETURN TO CALLER
EJECT
************************************************************************
*--------------------------------------------------------------------*
* A 4 Ø Ø _ O P E N _ P I P E : EXCI Open_Pipe                        *
*--------------------------------------------------------------------*
************************************************************************
* R2 DSECT - CM42ØCOM -
* R3 BASE
* R8 DSECT - CM419COM
* R9 DSECT - EXCI_RETURN_CODE
* R13 DSECT - DFHEISTG
* R14 Linkage
******************************************************************************
A4ØØ_OPEN_PIPE DS ØH
ST R14,A4ØØSR14 SAVE REGISTER 14
XC CURR_RC,CURR_RC CLEAR CURRENT RC
BAL R14,Z2ØØ_WRITE_SYSPRINT BLANK LINE
* ------------------------------------------------------------------------*
* Write message for Open_Pipe call. *
* ------------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø4Ø1I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø4Ø1I),CM42Ø4Ø1I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
* ------------------------------------------------------------------------*
* Call Open_Pipe *
* ------------------------------------------------------------------------*
MVC CM42ØCOM_CALL_TYPE,=AL4(OPEN_PIPE) CALL TYPE
A4ØØCALL CALL DFHXCIS, X
(CM42ØCOM_VERSION_NUM, X
CM42ØCOM_RETURN_AREA, X
CM42ØCOM_USER_TOKEN, X
CM42ØCOM_CALL_TYPE, X
CM42ØCOM_PIPE_TOKEN), X
VL, X
MF=(E,CM42ØCOM_PL)
* ------------------------------------------------------------------------*
* Check the response code and produce diagnostics if required *
* ------------------------------------------------------------------------*
A4ØØCHCK CLC EXCI_RESPONSE,=AL4(EXCI_NORMAL) IF NORMAL RESPONSE
BE A4ØØOK THEN OK MESSAGE
BAL R14,Z3ØØ_EXCI_DIAGNOSTICS ELSE DIAGNOSTICS
MVC CURR_RC,EXCI_RESPONSE SET RC
B A4ØØRET AND RETURN.
A4ØØOK MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø4Ø2I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø4Ø2I),CM42Ø4Ø2I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
* ------------------------------------------------------------------------*
* Return to caller *
* ------------------------------------------------------------------------*
A4ØØRET BAL R14,Z4ØØ_CHECK_RC CHECK HIGHEST RC
L R14,A4ØØSR14 RESTORE REGISTER 14
BR R14 RETURN TO CALLER
EJECT
******************************************************************************
* A 5 Ø Ø _ S U B _ P R O G : sub-program for DPL call *
* ------------------------------------------------------------------------*
A5ØØ_SUB_PROG DS ØH
ST R14, A5ØØSR14           SAVE REGISTER 14
XC CURR_RC, CURR_RC          CLEAR CURRENT RC
BAL R14, Z2ØØ_WRITE_SYSPRINT  BLANK LINE

* Issue LOADING Message.
*---------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG, =C'CM42ØØØ1I'    MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42ØØØ1I), CM42ØØØ1I & TEXT
MVC CM419COM_SYSPRINT_DATA+16(L'EXCIPROG), EXCIPROG
BAL R14, Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

* LOAD the DPL sub-program specified in the job step PARM=*
* If the LOAD fails the program will abend.  *
*---------------------------------------------------------------------*
LA R4, EPLOC                --> PROGRAM NAME
A5ØØLOAD DS ØH
MVC EPLOC, EXCIPROG         MOVE SUB PROG. NAME FOR LOAD
BAL R14, Z1ØØ_LOAD_PROG      PROCESS LOAD

* Issue LINKing Message.
*---------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG, =C'CM42Ø5Ø1I'    MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø5Ø1I), CM42Ø5Ø1I & TEXT
MVC CM419COM_SYSPRINT_DATA+23(L'EXCIPROG), EXCIPROG
BAL R14, Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
BAL R14, Z2ØØ_WRITE_SYSPRINT BLANK LINE
MVC CM419COM_SYSPRINT_MSG(L'CM419COM_SYSPRINT_IOA-1), =C'*'  SEPERATOR LINE
BAL R14, Z2ØØ_WRITE_SYSPRINT WRITE SEPERATOR

* LINK to DPL sub-program passing CM419COM and CM42ØCOM. If the *
* LINK doesn't work the program will abend.  *
*---------------------------------------------------------------------*
LA R4, EPLOC                --> PROGRAM NAME
A5ØØLINK LINK EPLOC=(R4), PARAM=(CM419ST, CM42ØST),
MF=(E, LINK_PL)
ST R15, SUB_RC              SAVE RETURN CODE

* Convert RC to displayable decimal characters and issue message. *
*---------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG(L'CM419COM_SYSPRINT_IOA-1), =C'*'  SEPERATOR LINE
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE SEPERATOR
BAL R14,Z2ØØ_WRITE_SYSPRINT BLANK LINE
L R4,SUB_RC LOAD RC
CVD R4,WORKDW_1 CONVERT TO DECIMAL
UNPK WORKDW_2,WORKDW_1 CONVERT TO ...
OI WORKDW_2+7,X'FØ' ... DISPLAYABLE DECIMAL
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø5Ø2I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø5Ø2I),CM42Ø5Ø2I & TEXT
MVC CM419COM_SYSPRINT_DATA+34(L'EXCIPROG),EXCIPROG & PROG
MVC CM419COM_SYSPRINT_DATA+46(L'WORKDW_2),WORKDW_2 & RC
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT
CLC CURR_RC,SUB_RC IF CURRENT RC > SUBPROG RC
BH A5ØØRET THEN RETURN
MVC CURR_RC,SUB_RC ELSE RESET CURRENT RC

*--------------------------------------------------------------------*
* Return to caller                                                  *
*--------------------------------------------------------------------*
A5ØØRET BAL R14,Z4ØØ_CHECK_RC CHECK HIGHEST RC
L R14,A5ØØSR14 RESTORE REGISTER 14
BR R14 RETURN TO CALLER
EJECT

***********************************************************************
*--------------------------------------------------------------------*
*- A 6 Ø Ø _ C L O S E _ P I P E : EXCI Close_Pipe                  *
*--------------------------------------------------------------------*
***********************************************************************
A6ØØ_CLOSE_PIPE DS ØH
ST R14,A6ØØSR14 SAVE REGISTER 14
XC CURR_RC,CURR_RC CLEAR CURRENT RC
BAL R14,Z2ØØ_WRITE_SYSPRINT BLANK LINE

*--------------------------------------------------------------------*
* Write message for Close_Pipe call.  *
*--------------------------------------------------------------------*
MVC CM419COM_SYSPRINT_MSG,=C'CM42Ø6Ø1I' MOVE MSG. NO.
MVC CM419COM_SYSPRINT_DATA(L'CM42Ø6Ø1I),CM42Ø6Ø1I & TEXT
BAL R14,Z2ØØ_WRITE_SYSPRINT WRITE MSG TO SYSPRINT

*--------------------------------------------------------------------*
* Call Close_Pipe                                                  *
*--------------------------------------------------------------------*
MVC CM42ØCOM_CALL_TYPE,=AL4(CLOSE_PIPE) CALL TYPE
A6ØØCALL CALL DFHXCIS, X
(CM42ØCOM_VERSION_NUM, X
CM42ØCOM_RETURN_AREA, X
CM42ØCOM_USER_TOKEN, X
CM42ØCOM_CALL_TYPE, X
CM42ØCOM_PIPE_TOKEN), X
VL, X
MF=(E,CM42ØCOM_PL)

Editor’s note: we will conclude the code next month.

Carl Wade McBurnie
IT Consultant (Germany) © Xephon 2004
December 2001 – November 2004 index

Items below are references to articles that have appeared in *CICS Update* since issue 193, December 2001. References show the issue number followed by the page number(s). Subscribers can download copies of all issues in Acrobat PDF format from Xephon’s Web site.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Issue Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>3270 bridge</td>
<td>203.37-44, 216.28-49</td>
</tr>
<tr>
<td>722</td>
<td>209.3-5</td>
</tr>
<tr>
<td>Abend</td>
<td>209.3-5</td>
</tr>
<tr>
<td>Adapter</td>
<td>220.31-47</td>
</tr>
<tr>
<td>Affinities</td>
<td>200.7-22</td>
</tr>
<tr>
<td>APPC</td>
<td>224.9-27</td>
</tr>
<tr>
<td>Audit trail</td>
<td>222.39-47, 223.44-47</td>
</tr>
<tr>
<td>Autoinstall</td>
<td>225.14-28</td>
</tr>
<tr>
<td>Batch</td>
<td>198.27-42, 227.24-43, 228.16-27</td>
</tr>
<tr>
<td>BMS</td>
<td>206.3-9, 208.7-12</td>
</tr>
<tr>
<td>Book review</td>
<td>207.50</td>
</tr>
<tr>
<td>Browse</td>
<td>205.12-25, 206.17-30, 217.28-43</td>
</tr>
<tr>
<td>Case</td>
<td>203.44-47</td>
</tr>
<tr>
<td>Case translation</td>
<td>197.11-13</td>
</tr>
<tr>
<td>CEDA</td>
<td>220.3-7</td>
</tr>
<tr>
<td>CEKL</td>
<td>225.3-7</td>
</tr>
<tr>
<td>CEMT</td>
<td>212.3-6, 227.24-43, 228.16-27, 213.37-47</td>
</tr>
<tr>
<td>Chargeback</td>
<td>202.18-39, 203.20-36</td>
</tr>
<tr>
<td>CICSplex SM</td>
<td>194.19-28, 205.25-37, 206.10-17, 207.38-50, 208.13-17, 212.22-51, 213.37-47</td>
</tr>
<tr>
<td>CICSplex SM API</td>
<td>207.9-28, 208.22-43</td>
</tr>
<tr>
<td>CICS TS 2.2</td>
<td>197.3-11</td>
</tr>
<tr>
<td>Client/server application</td>
<td>223.10-26</td>
</tr>
<tr>
<td>Closing files</td>
<td>203.3-14</td>
</tr>
<tr>
<td>Cold start</td>
<td>217.3-6</td>
</tr>
<tr>
<td>Communication</td>
<td>207.3-8</td>
</tr>
<tr>
<td>Compiler</td>
<td>200.22-30</td>
</tr>
<tr>
<td>Conflicts</td>
<td>212.6-15</td>
</tr>
<tr>
<td>Connectors</td>
<td>218.7-17</td>
</tr>
<tr>
<td>COPY</td>
<td>181.28-31, 206.3-9</td>
</tr>
<tr>
<td>CPSMCONN</td>
<td>209.14</td>
</tr>
<tr>
<td>Cryptographic co-processor</td>
<td>197.13-17</td>
</tr>
<tr>
<td>CWA</td>
<td>197.18-30, 199.7-12</td>
</tr>
<tr>
<td>DB2</td>
<td>196.20-41, 197.31-51, 215.23-40</td>
</tr>
<tr>
<td>Deactiviating</td>
<td>201.24-40</td>
</tr>
<tr>
<td>Debugging</td>
<td>217.7-13, 220.8-23, 227.3-9</td>
</tr>
<tr>
<td>Definitions</td>
<td>223.27-43</td>
</tr>
<tr>
<td>DFHLG070</td>
<td>219.3-10</td>
</tr>
<tr>
<td>DFHRMUTL</td>
<td>209.6-13</td>
</tr>
<tr>
<td>DFHSHUNT</td>
<td>200.3-7</td>
</tr>
<tr>
<td>ECI</td>
<td>198.11-26</td>
</tr>
<tr>
<td>Edit</td>
<td>205.12-25, 206.17-30</td>
</tr>
<tr>
<td>EJB</td>
<td>225.29-47, 226.20-30</td>
</tr>
<tr>
<td>EJB Server</td>
<td>214.3-13</td>
</tr>
<tr>
<td>e-mail</td>
<td>203.14-19, 210.19-33, 213.14-37, 214.44-47</td>
</tr>
<tr>
<td>Exit enabling</td>
<td>196.42</td>
</tr>
<tr>
<td>File control</td>
<td>193.3-9, 204.8-13</td>
</tr>
<tr>
<td>Help</td>
<td>216.12-15</td>
</tr>
<tr>
<td>Hung terminal</td>
<td>198.3-10, 202.7-18</td>
</tr>
<tr>
<td>IIS</td>
<td>193.9-22</td>
</tr>
<tr>
<td>IMS statistics</td>
<td>228.9-16</td>
</tr>
<tr>
<td>Interval control</td>
<td>201.6-11</td>
</tr>
<tr>
<td>ISC connections</td>
<td>204.30-44</td>
</tr>
<tr>
<td>Java</td>
<td>198.11-26, 218.6-17</td>
</tr>
<tr>
<td>JNDI</td>
<td>221.28-46</td>
</tr>
<tr>
<td>Journal management</td>
<td>208.18-22</td>
</tr>
<tr>
<td>JPDA</td>
<td>220.8-23</td>
</tr>
<tr>
<td>JVM</td>
<td>214.3-13</td>
</tr>
<tr>
<td>Kill</td>
<td>224.3-8, 225.3-7, 226.3-9</td>
</tr>
<tr>
<td>LDAP</td>
<td>221.28-46</td>
</tr>
<tr>
<td>LE options</td>
<td>201.3-5</td>
</tr>
<tr>
<td>Topic</td>
<td>Pages</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Log</td>
<td>196.3-10, 214.15-27</td>
</tr>
<tr>
<td>Log exit</td>
<td>205.3-12</td>
</tr>
<tr>
<td>Logging</td>
<td>193.3-9</td>
</tr>
<tr>
<td>Log Manager</td>
<td>211.4-11</td>
</tr>
<tr>
<td>Loop analysis</td>
<td>222.13-26</td>
</tr>
<tr>
<td>Management</td>
<td>223.3-6, 223.3-10, 225.14-28</td>
</tr>
<tr>
<td>Maps</td>
<td>202.3-7</td>
</tr>
<tr>
<td>Maxtask</td>
<td>222.39-47, 223.44-47</td>
</tr>
<tr>
<td>MCT</td>
<td>222.3-6</td>
</tr>
<tr>
<td>Messages</td>
<td>186.9-22, 215.16-22</td>
</tr>
<tr>
<td>Mixed-case fields</td>
<td>220.3-7</td>
</tr>
<tr>
<td>Monitoring</td>
<td>199.12-40, 206.30-42, 207.29-38, 210.16-18, 213.6-14, 214.13-14</td>
</tr>
<tr>
<td>Netsat</td>
<td>219.11-18</td>
</tr>
<tr>
<td>Newcopy</td>
<td>219.31-43, 220.23-31</td>
</tr>
<tr>
<td>Opening files</td>
<td>203.3-14</td>
</tr>
<tr>
<td>OTE</td>
<td>201.40-47, 221.3-15</td>
</tr>
<tr>
<td>Output</td>
<td>226.10-19</td>
</tr>
<tr>
<td>PCL</td>
<td>196.10-20</td>
</tr>
<tr>
<td>Performance</td>
<td>195.12-13, 198.27</td>
</tr>
<tr>
<td>Performance Monitor</td>
<td>213.3-6, 216.15-28, 217.14-27, 222.6-13</td>
</tr>
<tr>
<td>PL/I</td>
<td>194.10-18</td>
</tr>
<tr>
<td>Problem determination</td>
<td>219.3-10</td>
</tr>
<tr>
<td>Programming</td>
<td>195.3-11</td>
</tr>
<tr>
<td>QMF</td>
<td>203.14-19, 205.37-43, 211.34-46, 212.15-22</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>194.40-49, 200.30-42</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>195.41-43, 196.43, 197.51, 198.43, 199.50-51, 201.47, 202.47, 203.47, 204.44-45, 205.43, 207.51, 210.43, 211.47, 212.51, 213.47, 214.47, 220.47, 221.47, 223.47, 225.47</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>195.41-43, 196.43, 197.51, 198.43, 199.50-51, 201.47, 202.47, 203.47, 204.44-45, 205.43, 207.51, 210.43, 211.47, 212.51, 213.47, 214.47, 220.47, 221.47, 223.47, 225.47</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>195.41-43, 196.43, 197.51, 198.43, 199.50-51, 201.47, 202.47, 203.47, 204.44-45, 205.43, 207.51, 210.43, 211.47, 212.51, 213.47, 214.47, 220.47, 221.47, 223.47, 225.47</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>195.41-43, 196.43, 197.51, 198.43, 199.50-51, 201.47, 202.47, 203.47, 204.44-45, 205.43, 207.51, 210.43, 211.47, 212.51, 213.47, 214.47, 220.47, 221.47, 223.47, 225.47</td>
</tr>
<tr>
<td>Questions and answers</td>
<td>195.41-43, 196.43, 197.51, 198.43, 199.50-51, 201.47, 202.47, 203.47, 204.44-45, 205.43, 207.51, 210.43, 211.47, 212.51, 213.47, 214.47, 220.47, 221.47, 223.47, 225.47</td>
</tr>
<tr>
<td>RCT</td>
<td>211.3-4</td>
</tr>
<tr>
<td>Reactivating</td>
<td>201.24-40</td>
</tr>
<tr>
<td>Refresh programs</td>
<td>200.42-51</td>
</tr>
<tr>
<td>Resource management</td>
<td>210.3-4</td>
</tr>
</tbody>
</table>
Advanced Computer Technology has announced C\TREK, its CICS optimization tool. The product provides users with online access to real-time CICS structure information, helping system programmers to identify potential operational and performance problems. C\TREK captures data with each request, allowing the user to view the information while the system continues to process. Data from each domain can be viewed. C\TREK also presents important domain information from each anchor block and associated control blocks. The product recommends (via the Help displays) possible solutions.


***

IBM has announced Version 3.3 of CICS VSAM Recovery for z/OS. The product lets users recover CICS and batch VSAM data after physical or logical corruption. The new version includes changes that increase the overall recovery capability, and improve management and usability.

A new batch backout feature can remove updates made to VSAM datasets by failed batch job steps (batch backout). It allows CICS VR to be notified of logical back-ups created for VSAM datasets made by almost any back-up product (including non-IBM ones). There have been various usability enhancements within grouping and back-up functions.

Features have been added to keep the CICS VR server address space available. Disaster recovery utilities have also been enhanced so they allow further control over the items that are sent to the remote recovery site.

For further information contact your local IBM representative. URL: www.ibm.com/software/htp/cics/vr. ***

NEON Systems has announced Shadow z/services, which converts CICS, IMS, and CA-IDMS applications into Web services.

Shadow z/Services’ introspection technology enables developers to parse application logic and screen definitions and generate a Web Services Description Language (WSDL). For screen-based applications, developers can use the product to control a logical flow between several screens to support a business process. That microflow is published as a Web service.

Shadow z/Services’ Web services client component enables mainframe application developers to invoke external Web services from CICS/TS, CA-IDMS, and batch.

For further information contact: NEON Systems, 14100 Southwest Freeway, Suite 500, Sugar Land, TX 77478, USA. Tel: (281) 491 4200. URL: www.neonsys.com/Shadow/shadow_zservices.asp.

***

Aton International has updated its TN3270 terminal emulation package for Pocket PCs running Windows Mobile and Pocket PC Phone Edition.

The product uses the TN3270 terminal protocol to connect to CICS, VM/CMS, or MVS. The new version adds full-screen landscape mode, soft keyboard, and session switching, allowing access to mainframe and Web services data.

URL: www.aton.com/Products/products.htm.