July 1999

In this issue

3  The CICS Web Interface – serving objects
18  Simplifying CICS to JES2 spool functions
25  Dealing with program abends
39  Displaying CPU usage by TCB – part 2
47  High-values for CSP transactions
48  CICS news

© Xephon plc 1999
The CICS Web Interface – serving objects

After installing the CICS Web Interface (CWI) feature on a CICS 4.1 (or higher) region, the need for a general object server utility soon becomes clear. (The term ‘object’ is meant to denote, but is not limited to, entities such as HTML pages, JPEG/GIF images, or plain text files.) The object server should exploit APAR PQ08889, which removes the 32KB limit on data outbound from the CWI. It is also often desirable to support the passing of a token amongst HTML pages, provided they contain a set keyword name for the token, delimited by an ampersand (&) and a semicolon (;). This article explains the steps that can be taken to achieve this and provides the source code for such an object server.

Note: the phrase ‘object server’ or ‘object server program’ should be taken to mean the object server program provided by this article. You should also be aware that there are significant benefits to serving binary objects like JPEG/GIF images from the MVS Web server available in an OS/390 Open Edition environment versus doing so over the CWI.

If/when this is done, the object server mechanism provided would not necessarily need modification, and hopefully this article is written to make any adjustments fairly easy and straightforward.

ASSUMPTIONS
To keep this article to a reasonable length, certain assumptions must be made regarding the CWI. These assumptions are:

• The reader has knowledge of the CWI – by having installed it and/or started writing CWI applications. Familiarity with the CICS Web Interface Guide is expected.
• If the reader is a systems programmer, the function and purpose of a CWI analyser program is understood.
• If the reader is writing CWI applications, the use of the template manager utility and symbol lists is understood.
• The reader understands what is meant by a converter program,
alias tranid, server program, and token (query_string), as applied to a CWI URL.

- The reader is familiar with the Template Manager PDS defined to DDname DFHHTML in a CWI region’s start-up JCL and knows how to FTP a binary object into DFHHTML.

URL CONSIDERATIONS
A CWI URL, as described by the *CICS Web Interface Guide*, is typically formatted as follows:

```
http://Host:CICS_Port/Converter_Pgm/Alias_Tranid/Server_Pgm{?token}
```

The object server program, named JCHWBOS, is invoked as what might be called a ‘stand-alone’ converter program in a CWI URL using the above format. The URL for the object server is:

```
http://Host:CICS_Port/JCHWBOS/Obj_Tranid/Obj_Filename{?token}
```

where:

- ‘Obj_Tranid’ (specified in the Alias_Tranid field of the URL) is used to indicate the type and size of the object to be served. Obj_Tranid may be one of several arbitrarily named alias_tranids specifically defined for use with the object server. The first three characters of Obj_Tranid indicate the type of object, and the fourth character, if a ‘B’, indicates the object is larger than 32KB. For the supplied object server, Obj_Tranid may be one of the alias transactions listed in Figure 1, defined for the object indicated.

Note: The transaction IDs will be explicitly referenced in the analyser and the object server programs.

- ‘Obj_Filename’ (specified in the Server_Pgm field of the URL) is used to name the object to be served. That is, it is equal to the name of the template manager PDS (DFHHTML) member that houses the object to be served.

- ‘?token’ may optionally be specified. If the object server is asked to serve HTML, it gives a symbol list for the token to the template manager before the HTML is fetched. If there is no token, the keyword (QUERY_STRING) for the token in the symbol list will be assigned a null value. If a token exists then the keyword will
<table>
<thead>
<tr>
<th>Transaction</th>
<th>Specified to serve</th>
</tr>
</thead>
<tbody>
<tr>
<td>GIF</td>
<td>A GIF image smaller than 32KB</td>
</tr>
<tr>
<td>HTM</td>
<td>An HTML page smaller than 32KB</td>
</tr>
<tr>
<td>HTMB</td>
<td>An HTML page larger than 32KB</td>
</tr>
<tr>
<td>JPG</td>
<td>A JPEG image smaller than 32KB</td>
</tr>
<tr>
<td>TXT</td>
<td>A plain text file smaller than 32KB</td>
</tr>
<tr>
<td>TXTB</td>
<td>A plain text file larger than 32KB</td>
</tr>
</tbody>
</table>

*Figure 1: Alias transactions*

be assigned the value of the token. Since this ensures the keyword QUERY_STRING (delimited by an ampersand and semicolon in the HTML) will always be either nulled or valued, it permits QUERY_STRING to be specified in an HTML page regardless of whether the application requires it.

Sample object server URLs follow:

- To serve JPEG image BANNER1 from the DFHHTML PDS:
  
  http://Host:CICS_Port/JCHWBOS/JPG/BANNER1

- To serve GIF image BANNER2 from the DFHHTML PDS:
  
  http://Host:CICS_Port/JCHWBOS/GIF/BANNER2

- To serve HTML page HTMLPG1 from the DFHHTML PDS:
  
  http://Host:CICS_Port/JCHWBOS/HTM/HTMLPG1

- To serve plain text file TXTFILE1 from the DFHHTML PDS:
  
  http://Host:CICS_Port/JCHWBOS/TXT/TXTFILE1

- To serve HTML page HBIG (larger than 32KB) from the DFHHTML PDS:
  
  http://Host:CICS_Port/JCHWBOS/HTMB/HBIG
The following URL will serve HTML page HX601 from the DFHHTML PDS and substitute the token indicated for symbol list keyword ‘&QUERY_STRING;’ in HX601:

http://Host:CICS_Port/JCHWBOS/HTM/HX601?aABR549z

RESOURCE DEFINITIONS

You will need to create resource definitions in the CSD file for the object server and transactions to indicate the type and size of objects to be served. The following is a suggested definition for the object server program. Note that attributes allowed to default may not be shown.

**PROGram : JCHWBOS**
Group : your_local_CWI_group
DEscription : CWI OBJECT SERVER
Language : Cobol
DAtalocation : Any
EXECKey : User

To support the object server supplied, a definition should be created for tranids GIF, JPG, HTM, HTMB, TXT, and TXTB. With the exception of the tranid, and perhaps a description entered for it, all the definitions would look alike. The following is a suggested template for the transaction definitions. Note that attributes allowed to default may not be shown.

**TRANSaction : ????
Group : your_local_CWI_group
DEscription : ALIAS TRANID FOR OBJ SERVER PGM TO SERVE ??? OBJ
PROGram : DFHWBA
PROFILE : DFHCICST
TASKDATALoc : Any
TASKDATAKey : User
PRIORity : 255

Ensure that access to the transactions defined above is made available to everyone who will need to use them. For instance, you may want them defined to a security profile granting READ access to anyone having access to the CWI region.

**DFHHTML CONSIDERATIONS**

The object server uses the CWI template manager utility to obtain the object it is to serve from the template manager PDS. The template
manager PDS is defined to DDname DFHHTML in your CICS region’s start-up JCL. If you have not defined a DFHHTML PDS you can use the guidelines below to do so. Regardless, you may find the information below useful.

In order to serve a binary object such as a JPEG or GIF image up to 32KB, it is suggested DFHHTML be defined with RECFM=VB, BLKSIZE=32604, and LRECL=32600. The reason the sizes do not precisely reflect 32KB is because the images will be served as part of an HTTP response, and there must be room for its headers. If you care to calculate exactly how big your headers will ever be, you may be able to squeeze another 80 or so bytes out for the image, but you may find cutting it that close can cause headaches later when you find, for some reason, you need a few more bytes for the headers.

You may have noticed previously there were no ‘B’ tranids defined for GIF/JPEG objects larger than 32KB. Now you may also be wondering how what was said in the introduction about sending outbound data larger than 32KB is applicable to image objects. Such objects are special in this regard. This is because a binary object greater than 32KB, put into DFHHTML, will reside there as a multi-line member. And when the template manager fetches a member from DFHHTML it will append CR/LF (carriage return, line feed) bytes to the end of each line in the member. This is OK for HTML or plain text files, but obviously not so for binary objects.

Although an object server may be coded to account for this situation, the supplied object server was not, for several reasons. One is that, for business needs, 32KB is normally ample room for an image. Another reason is that it’s very likely you will, if you haven’t already, evolve to using an MVS Web server (eg Domino Go Webserver for OS/390) as a centralized server for your binary objects. But the main reason is that the code to do so is not easily written in COBOL. Although the object server could have been provided in Assembler, a supplementary goal of this article is to assist COBOL programmers by example with writing CWI applications. For those interested, a short follow-on article is planned on serving DFHHTML binary files greater than 32KB, and Assembler code (less than 25 lines) to do this will be shown.
If you create a DFHHTML PDS, be sure that your CWI region has READ access to it. And, if you define a new DFHHTML PDS to replace an existing one, you can simply copy the members in the ‘old’ PDS to the ‘new’ one.

ANALYSER CONSIDERATIONS

Your CWI analyser program must be modified to recognize when the object server program is being used and determine whether or not to turn ‘off’ ASCII/EBCDIC translation for binary objects (ie GIF/JPEG images). Since the analyser will do the latter by checking the value of Obj_Tranid in the tranid field of the URL, the object server also expects the analyser to verify that Obj_Tranid is one of the valid tranids defined for the object server.

If a binary or plain text object is being served, a quick exit from the analyser is usually in order. However, if an HTML page is being served, it’s expected the analyser will continue scanning the URL for a token.

If you are not using tokens, you can exit the analyser as you wish, and you may want to modify the object server by removing the ‘token relevant’ code, although it won’t hurt to leave it in.

Two blocks of code will need to be added to the analyser. The first should be inserted just after the point at which the analyser has determined that a valid server program has been specified in the CWI URL. It would look like the following pseudo-code:

```pseudo-code
if wbra-converter-program = 'JCHWBOS'
    if wbra-alias-tranid = 'GIF' or wbra-alias-tranid = 'JPG'
        move low-values to wbra-dfhcnv-key
        perform return-to-CICS
    end-if
    if wbra-alias-tranid = 'TXT' or wbra-alias-tranid = 'TXTB'
        perform return-to-cics
    end-if
else
    if wbra-alias-tranid not = 'HTM' and wbra-alias-tranid not = 'HTMB'
        perform invalid-obj-tranid-error
    end-if
end-if
continue processing (ie scan of URL for token)
```

The second block of code to be added to the analyser would be that accounting for the invalid-obj-tranid-error condition. You can model
it after any existing error handling routines in your analyser.

TDQ CONSIDERATIONS
If the object server detects an error, an attempt will be made to write diagnostic information to the CWBO transient data queue. The CWBO TDQ is defined to a region’s DCT table as part of the CWI installation process for the region.

OBJECT SERVER CODE
It may be worthwhile to note that the object server was written as a ‘stand-alone’ converter, rather than a server program, primarily for two reasons:

- An object larger than 32KB can be served only from a converter because it has access to the address of the HTTP request/response storage, and a server program does not.
- As a converter program, the object server has direct addressability to certain information it needs to serve an object whereas a server program, to obtain the same information, would have to perform a scan of the HTTP request headers and/or link to the CWI environment variables utility.

The only negative aspect of a ‘stand-alone’ converter program is that it is needlessly invoked for ENCODE processing despite specifying no server program. If enough people express a concern about this to IBM CICS technical support, perhaps an enhancement for it will be released.

The source for the object server follows. It is provided in COBOL because the CWI, as shipped, does not contain documentation overly friendly to COBOL programmers, and some techniques are used that may come in handy elsewhere during CWI programming. However, you can significantly reduce the program size by converting it to Assembler if you choose to do so.

SOURCE CODE

PROCESS XOPTS(NOLINKAGE)
ID DIVISION.
PROGRAM-ID. JCHWBOS.
*    CWI Object Serving Utility

ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.

*    First the CWI User Replaceable Program Constants and
*    the CWI DFHWBTBL (Template Manager) link parms are copied
COPY DFHWBU0C.
COPY DFHWBTLO.

*    Next are the headers for the HTTP response that will contain
*    the object to be served. Note: the headers are 73 bytes in
*    length, and an equivalent length for the headers is
*    specified in the Linkage Section.
Ø1 WS-HTTP-RESPONSE-HDRS.
    Ø2 WS-HTTP-RESP-LENGTH PIC S9(8) COMP.
    Ø2 FILLER PIC X(15) VALUE 'HTTP/1.0 200 OK'.
    Ø2 FILLER PIC X(2) VALUE X'0D25'.
    Ø2 FILLER PIC X(14) VALUE 'Content-Type: '.
    Ø2 WS-HTTP-RESP-CONTENT-TYPE PIC X(10).
    Ø2 FILLER PIC X(2) VALUE X'0D25'.
    Ø2 FILLER PIC X(16) VALUE 'Content-Length: '.
    Ø2 WS-HTTP-RESP-CONTENT-LENGTH PIC 9(6).
    Ø2 FILLER PIC X(4) VALUE X'0D250D25'.

*    Next, the HTTP response, sent if there is a problem serving
*    the object. Note the response is fixed meaning the length
*    of it may be hardcoded in the first four bytes so it will
*    not have to be computed in the program.
Ø1 WS-HTTP-ERROR-RESPONSE.
    Ø2 FILLER PIC S9(8) COMP VALUE 35.
    Ø2 FILLER PIC X(27) VALUE 'HTTP/1.0 404 File not found'.
    Ø2 FILLER PIC X(4) VALUE X'0D250D25'.

*    The tranid with which we are invoked will indicate the type
*    of object to be served and whether or not it's bigger than
*    32K. Storage for it follows.
Ø1 WS-EIBTRNID.
    Ø2 WS-OBJ-TYPE PIC X(3).
    Ø2 WS-OBJ-SIZE-INDICATOR PIC X(1).

*    Next are two variables needed to manage the storage
*    available in which to fetch the object. By default, a 32K
*    buffer will be available, but to serve an object larger than
*    32K an arbitrary limit must be set for its maximum size
* (250K has been chosen). Note: the precise amount of storage
* available for the object will be the storage available minus
* the length of the headers required for the HTTP response
* that will contain the object. We'll use a variable to
calculate this.

Ø1 WS-MAX-OBJ-STOR-ALLOWED PIC S9(8) COMP VALUE 256000.
Ø1 WS-OBJ-STOR-AVAIL PIC S9(8) COMP.

* Layout for logging record. Used to log error messages in
* particular to the CWI TDQ (CWBO).

Ø1 WS-LOG-RECORD.
 Ø2 FILLER PIC X(10) VALUE ' JCHWBOS: '.
 Ø2 WS-LOG-DATE PIC X(8) VALUE SPACES.
 Ø2 FILLER PIC X(1) VALUE SPACES.
 Ø2 WS-LOG-TIME PIC X(8) VALUE SPACES.
 Ø2 FILLER PIC X(1) VALUE SPACES.
 Ø2 WS-LOG-MSG PIC X(100) VALUE SPACES.

* Layout for error msg built if we have a problem.

Ø1 WS-LOG-ERROR-MSG.
 Ø2 FILLER PIC X(21) VALUE 'ERROR SERVING OBJECT '.
 Ø2 WS-LOG-ERROR-OBJNAME PIC X(8) VALUE 'Unknown'.
 Ø2 FILLER PIC X(8) VALUE '. RESP='. 
 Ø2 WS-LOG-ERROR-RESP PIC 9(4).
 Ø2 FILLER PIC X(8) VALUE ', RESP2='. 
 Ø2 WS-LOG-ERROR-RESP2 PIC 9(4).
 Ø2 FILLER PIC X(1) VALUE SPACES.
 Ø2 WS-LOG-ERROR-TEXT PIC X(46) VALUE SPACES.

* And lastly, WS for DFHCOMMAREA pointer and a work variable to
* pick up the time.

Ø1 DFHCOMMAREA-PTR POINTER.
77 WS-ABSTIME PIC S9(15) COMP-3.

*================================================================

LINKAGE SECTION.

* This program is compiled with XOPTS(NOLINKAGE) so we need to
* copy in the EIB block. Also, we need to copy the CWI
* Converter parms

COPY DFHEIBLC.
COPY DFHWBCDO.
* Next is the COMMAREA for the HTTP response that will contain
* the object to be served. Notice the amount of storage
* reserved for the headers must match its equivalent
* previously defined. The storage area defined for the object
* will be referenced by pointer only, and since nothing is
* COBOL MOVEd to JCHWBOS-COMMAREA larger than the length of
* the headers, the length value of COMMAREA-OBJ-STORAGE is
* irrelevant, hence its PIC X(1). Caution: This will probably
* not be true if you clone this code elsewhere because symbol
* lists for HTML templates are often larger than the length
* of the headers for their HTML.

Ø1 JCHWBOS-COMMAREA.
Ø2 COMMAREA-HTTP-RESP-HDRS                       PIC X(73).
Ø2 COMMAREA-OBJ-STORAGE                          PIC X(1).

*================================================================
PROCEDURE DIVISION USING DFHEIBLK DFHCOMMAREA.

DRIVER.

    PERFORM INITIAL-VERIFICATION.
    PERFORM DETERMINE-OBJ-STOR-AVAILABLE.
    PERFORM DETERMINE-OBJECT-TYPE.
    PERFORM GET-OBJECT.
    PERFORM RETURN-TO-CICS.

*================================================================

INITIAL-VERIFICATION.

* This paragraph primarily verifies we have a COMMAREA and
* we're called with a valid converter function. Additional
* notes follow.
* CWI converters require a response code to be set. We
* expect to complete successfully or otherwise handle any
* "handleable" errors, so once the converter parm area is
* addressed, we will set the converter response by default to
* OK.
* Since this is a "stand-alone" converter program ENCODE
* is a null function for us. That is, if invoked for ENCODE
* we simply return to CICS. This also underlines the fact
* DECODE_SERVER_PROGRAM is really the name of the object we're
* to serve, not a server program. This means we must clear it
* before losing control after (or in) DECODE. And although
* the name of the object is put in the COMMAREA for the
* Template Manager at label WBTL-TEMPLATE-NAME, it's unlikely
* you will be able to retrieve the name of the object from
* there if the Template Manager has problems. So, for error
* handling we will store the name of the object in working
  * storage as well.

  IF EIBCALEN = Ø PERFORM COMMAREA-ERROR.
  EXEC CICS ADDRESS
    COMMAREA(DFHCOMMAREA-PTR)
  END-EXEC.
  SET ADDRESS OF CONVERTER-PARMS TO DFHCOMMAREA-PTR.
  MOVE URPI-OK TO CONVERTER-RESPONSE.
  EVALUATE CONVERTER-EYECATCHER
    WHEN ENCODE-EYECATCHER-INIT PERFORM RETURN-TO-CICS
    WHEN DECODE-EYECATCHER-INIT PERFORM
      MOVE DECODE-SERVER-PROGRAM TO WS-LOG-ERROR-OBJNAME
      WBTL-TEMPLATE-NAME
      MOVE LOW-VALUES TO DECODE-SERVER-PROGRAM
      END-PERFORM
    WHEN OTHER PERFORM CONVERTER-PARM-ERROR
  END-EVALUATE.

*================================================================

DETERMINE-OBJ-STOR-AVAILABLE.

* When invoked, the storage available for our HTTP
  * response containing the object will be equal to
  * DECODE-OUTPUT-DATA-LEN (32K by default), and it will be
  * addressed by DECODE-DATA-PTR. By convention, byte 4 of the
  * object type tranid may equal "B" (for "big") to indicate an
  * object larger than 32K is to be served. If "B" is present,
  * we will FREEMAIN the original storage obtained for the HTTP
  * request and GETMAIN a larger area setting DECODE-DATA-PTR to
  * the address of the new storage obtained. Then the storage
  * available for the object we're to fetch is computed - as the
  * Template Manager will need to know it. It's important to
  * note also it is here we address the storage using our
  * COMMAREA.

  MOVE EIBTRNID TO WS-EIBTRNID.
  IF WS-OBJ-SIZE-INDICATOR = 'B'
    EXEC CICS FREEMAIN
      DATAPORTER(DECODE-DATA-PTR)
      NOHANDLE
    END-EXEC
  EXEC CICS GETMAIN
    SET(DECODE-DATA-PTR)
    FLENGTH(WS-MAX-OBJ-STOR-ALLOWED)
    NOHANDLE
  END-EXEC
  IF EIBRESP NOT = DFHRESP(NORMAL)
    PERFORM GETMAIN-ERROR
END-IF.

COMPUTE WS-OBJ-STOR-AVAIL =
   WS-MAX-OBJ-STOR-ALLOWED -
   LENGTH OF WS-HTTP-RESPONSE-HDRS

ELSE

COMPUTE WS-OBJ-STOR-AVAIL =
   DECODE-OUTPUT-DATA-LEN -
   LENGTH OF WS-HTTP-RESPONSE-HDRS

END-IF.

SET ADDRESS OF JCHWBOS-COMMAREA TO DECODE-DATA-PTR.

*=================================================================

DETERMINE-OBJECT-TYPE.

* By convention, the first 3 bytes of the object type tranid
* will specify the object type, and this is used to establish
* the HTTP Content-Type header. We rely on our CWI analyser
* to ensure we are invoked with a valid object type. Also, if
* HTML is to be served we ensure a security token, if it
* exists, is obtained and inserted into the HTML via a symbol
* list passed to the Template Manager. This follows a
* convention where a name value of &QUERY_STRING; (upper or
* lower case) is used by the creator of the HTML. Notice when
* a token exists, the maximum length of the symbol list built
* for it is 46 bytes and this length is passed to the Template
* Manager. However, as the symbol list is built it will not
* contain any spaces, meaning it could actually be less than
* 46 bytes. This also means the Template Manager could put
* undesired blanks in the HTML for the trailing name in the
* symbol list. To prevent this, an ampersand is appended to
* the last non-blank in it. This technique can come in handy
* in other CWI programs. A reminder: the pointer to our
* COMMAREA, which the symbol list is put into, is
* DECODE-DATA-PTR, set and addressed in the previous
* paragraph.

EVALUATE WS-OBJ-TYPE
   WHEN 'GIF' MOVE 'image/gif ' TO WS-HTTP-RESP-CONTENT-TYPE
   WHEN 'JPG' MOVE 'image/jpeg' TO WS-HTTP-RESP-CONTENT-TYPE
   WHEN 'TXT' MOVE 'text/plain' TO WS-HTTP-RESP-CONTENT-TYPE
   WHEN OTHER PERFORM
      MOVE 'text/html ' TO WS-HTTP-RESP-CONTENT-TYPE
   IF DECODE-USER-TOKEN = LOW-VALUES
      MOVE 'QUERY_STRING=&query_string=' TO JCHWBOS-COMMAREA
      MOVE 27 TO WBTL-SYMBOL-LIST-LEN
   ELSE
      STRING 'QUERY_STRING=?' DELIMITED BY SIZE
      DECODE-USER-TOKEN DELIMITED BY ' '
      '&query_string=?' DELIMITED BY SIZE

GET-OBJECT.

* Now it's time to call the Template Manager to fetch the
* object into our COMMAREA previously addressed by
* DECODE-DATA-PTR. However, to keep from overlaying the part
* of our COMMAREA reserved for the HTTP headers, we give the
* Template Manager a pointer to storage specifically for the
* object. Then we tell the Template Manager the length of
* storage available for the object (previously computed) by
* putting the length in WBTL-HTML-BUFFER-LEN. When the
* Template Manager returns, WBTL-HTML-BUFFER-LEN will contain
* the length of the unused portion of the buffer which we use
* to compute the actual length of the object. This is needed
* to finish the Content-Length header and to compute the total
* length of the HTTP response. Note: the Template Manager
* appends a CR/LF to the object it fetches so if a GIF or JPG
* object is fetched we subtract 2 from the object's length.
* After the length of the object and total HTTP response
* length are computed the headers are moved to storage
* reserved for them in our COMMAREA. Everything is now where
* it should be ready for the return to the CWI server
* controller.

MOVE WBTL-CURRENT-VERSION TO WBTL-VERSION-NO.
MOVE WBTL-BUILD-HTML-PAGE TO WBTL-FUNCTION.
SET WBTL-HTML-BUFFER-PTR TO ADDRESS OF COMMAREA-OBJ-STORAGE.
EXEC CICS LINK PROGRAM('DFHWBTL')
    COMMAREA(DFHWBTL-ARG)
    LENGTH(LENGTH OF DFHWBTL-ARG)
    NOHANDLE
END-EXEC.
IF EIBRESP NOT = DFHRESP(NORMAL)
    PERFORM DFHWBTL-LINK-ERROR
END-IF.
IF WBTL-RESPONSE NOT EQUAL ZERO
    PERFORM DFHWBTL-RESPONSE-ERROR
END-IF.
COMPUTE WS-HTTP-RESP-CONTENT-LENGTH =
WS-OBJ-STOR-AVAIL - WBTL-HTML-BUFFER-LEN.

EVALUATE WS-OBJ-TYPE
WHEN 'GIF' SUBTRACT 2 FROM WS-HTTP-RESP CONTENT-LENGTH
WHEN 'JPG' SUBTRACT 2 FROM WS-HTTP-RESP CONTENT-LENGTH
WHEN OTHER CONTINUE
END-EVALUATE.

 COMPUTE WS-HTTP-RESP-LENGTH =
 LENGTH OF WS-HTTP-RESPONSE-HDRS +
 WS-HTTP-RESP CONTENT-LENGTH.

 MOVE WS-HTTP-RESPONSE-HDRS TO COMMAREA-HTTP-RESP-HDRS.

******************************************************************************
*                    ERROR Handlers                                           *
******************************************************************************

COMMAREA-ERROR.
 MOVE 'ERROR: Invoked with EIBCALEN = Ø.' TO WS-LOG-MSG.
 PERFORM WRITE-LOG-REC.
 PERFORM RETURN-TO-CICS.

CONVERTER-PARM-ERROR.
 MOVE 'ERROR: CONVERTER EYECATCHER NOT ENCODE OR DECODE.' TO
 WS-LOG-MSG.
 PERFORM WRITE-LOG-REC.
 MOVE URP INVALID TO CONVERTER-RESPONSE.
 MOVE URP CORRUPT CLIENT DATA TO CONVERTER-REASON.
 PERFORM RETURN-TO-CICS.

GETMAIN-ERROR.
 MOVE 'RETURNED FROM GETMAIN' TO WS-LOG ERROR-TEXT.
 PERFORM LOG ERROR.

DFHWBTL-LINK-ERROR.
 MOVE 'RETURNED FROM LINK TO DFHWBTL' TO WS-LOG ERROR-TEXT.
 PERFORM LOG ERROR.

LOG-ERROR.
 MOVE EIBRESP TO WS-LOG ERROR RESP.
 MOVE EIBRESP2 TO WS-LOG ERROR RESP2.
 MOVE WS-LOG ERROR MSG TO WS-LOG MSG.
 PERFORM WRITE-LOG REC.
 PERFORM SEND ERROR RESPONSE.

DFHWBTL-RESPONSE-ERROR.
 * The two most common errors are likely to come from trying to
 * serve a template not in the HTML PDS or one too large. As
 * such it can be helpful adding relevant text to the error
 * message logged when they occur. This beats looking up the
 * codes in the CWI Guide.
 EVALUATE WBTL-REASON
 WHEN WBTL-TEMPLATE NOT FOUND
MOVE 'RETURNED BY DFHWBTL (OBJECT NOT FOUND)' TO WS-LOG-ERROR-TEXT
WHEN WBTL-TEMPLATE-TRUNCATED
  MOVE 'RETURNED BY DFHWBTL (OBJECT TOO BIG)' TO WS-LOG-ERROR-TEXT
  WHEN OTHER
  MOVE 'RETURNED BY DFHWBTL' TO WS-LOG-ERROR-TEXT
END-EVALUATE.

MOVE WBTL-RESPONSE TO WS-LOG-ERROR-RESP.
MOVE WBTL-REASON TO WS-LOG-ERROR-RESP2.
MOVE WS-LOG-ERROR-MSG TO WS-LOG-MSG.
PERFORM WRITE-LOG-REC.
PERFORM SEND-ERROR-RESPONSE.

SEND-ERROR-RESPONSE.
* Since we're not sure when this may be invoked we must ensure
* the output area pointer points to our COMMAREA.
SET ADDRESS OF JCHWBOS-COMMAREA TO DECODE-DATA-PTR.
MOVE WS-HTTP-ERROR-RESPONSE TO JCHWBOS-COMMAREA.
PERFORM RETURN-TO-CICS.

*================================================================
WRITE-LOG-REC.
EXEC CICS ASKTIME
  ABSTIME(WS-ABSTIME)
  NOHANDLE
END-EXEC.
EXEC CICS FORMATTIME
  ABSTIME(WS-ABSTIME)
  MMDDYY(WS-LOG-DATE) DATESEP
  TIME(WS-LOG-TIME) TIMESEP
  NOHANDLE
END-EXEC.
EXEC CICS WRITEQ TO QUEUE('CWBO')
  FROM(WS-LOG-RECORD)
  LENGTH(LENGTH OF WS-LOG-RECORD)
  NOHANDLE
END-EXEC.
MOVE SPACES TO WS-LOG-MSG.

*================================================================
RETURN-TO-CICS.
EXEC CICS RETURN END-EXEC.
GOBACK.

*================================================================ That's all! -----------------------------------------------

John Hayes
CICS Systems Programmer (USA) © John C Hayes 1999
Simplifying CICS to JES2 spool functions

DESCRIPTION
The following SPOOLPGM program can be used to simplify CICS to JES2 spool functions.

SPOOLPGM is the main program that processes the specific CICS spool calls. Using the parameter list passed, it:

- Opens a JES2 output spool file with the specified form and remote printer-id.
- Writes records to the JES2 output spool file.
- Closes the JES2 output spool file, which can then be controlled through normal JES2 operations.

SPOOLTST is a sample CICS COBOL calling program and demonstrates the usage of the SPOOLPGM program.

HEX2CHAR is the macro used to convert hexadecimal to character for display.

The parameter passed to SPOOLPGM is used to control the requested spool function, form name, and remote printer-id. The parameter list is passed using standard CICS COMMAREA facilities.

Some basic error processing is handled by SPOOLPGM, specifically ‘SPOOLBUSY’. If this condition occurs, the program will wait for one second, then retry the request up to five times before returning an error message.

All other error conditions are translated for display and returned via SPLPARM to the calling program.

SPOOLPGM

TITLE 'SPOOLPGM - CICS SPOOL INTERFACE'
*
*****************************************************************

* DESCRIPTION:
* SUB-PROGRAM TO SIMPLIFY CICS SPOOL FUNCTIONS.
* CALLING PROGRAM LINKS TO THIS PROGRAM WITH A
* PARAMETER LIST TO CONTROL SPOOL FUNCTIONS.
* IE SPOOLOPEN, SPOOLWRITE, AND SPOOLCLOSE
*
* PARM:
* SPLCMD : REQUESTED FUNCTION
* SPLTOKEN : TOKEN USED BY JES
* SPLPARM : FUNCTION - SUB-PARAMETERS:
  * SPOOLOPEN? - OUTPUT FORMNAME, JES REMOTE ID
  * SPOOLWRITE? - OUTPUT LINE
  * SPOOLCLOSE? - NOT USED
  * NOTE: SPLPARM ALSO USED TO RETURN MESSAGES
* SPLCLASS : OUTPUT CLASS
* SPLREQOK : RETURN REQUEST SUCCESS (Y OR N)
*
*****************************************************************
*
SPOOLPGM CSECT
B     STARTØ
DC    CL8'SPOOLPGM'
DC    CL8'&SYSDATE'
DC    CL8'&SYSTIME'
*
STARTØ EQU *
*** HANDLE CONDITIONS
EXEC  CICS HANDLE ABEND LABEL(ERROR)
L     4,DFHEICAP             GET COMMAREA POINTER
USING SPLD,4                 ADDRESS COMMAREA
SR    9,9                    CLEAR COUNT REGISTER
CLC   SPLCMD,=C'SPOOLOPEN'  SPOOL OPEN?
BE    SPLOPEN
CLC   SPLCMD,=C'SPOOLWRITE' SPOOL WRITE?
BE    SPLWRITE
CLC   SPLCMD,=C'SPOOLCLOSE' SPOOL CLOSE?
BE    SPLCLOSE
B     INVCMD
*
SPLOPEN EQU *
BALR 10,0             SAVE REQUEST START POINT
LA    9,1,(9)        COUNT REQUESTS
MVI   SPLREQOK,C'Y'  INITIALIZE INTERNAL RC
MVC   SPLWORK(88),SPLPARM MOVE PASSED PARM
MVC   0_CLASS(1),SPLCLASS OVERRIDE OUTPUT CLASS
LA    5,OUTDESCR     SET-UP POINTER FOR SPOOLOPEN
ST    5,PARMPTR      SET-UP POINTER FOR SPOOLOPEN
LA    5,PARMPTR      SET-UP POINTER FOR SPOOLOPEN
EXEC  CICS SPOOLOPEN OUTPUT TOKEN(SPLTOKEN)
* SPLWRITE EQU *
  BALR 10,0                  SAVE REQUEST START POINT
  LA 9,1(,9)                  COUNT REQUESTS
  EXEC CICS SPOOLWRITE TOKEN(SPLTOKEN)    X
     FROM(SPLPARAM) LINE NOHANDLE
  L 5,EIBRESP
  LTR 5,5
  BNZ ERROR
  B RETURN
*
* SPLCLOSE EQU *
  BALR 10,0                  SAVE REQUEST START POINT
  LA 9,1(,9)                  COUNT REQUESTS
  EXEC CICS SPOOLCLOSE TOKEN(SPLTOKEN) NOHANDLE
  L 5,EIBRESP
  LTR 5,5
  BNZ ERROR
  B RETURN
*
* SPLBUSY EQU *
  C 9,FULL5                  SPOOLBUSY 5 TIMES?
  BH SPLBUSY5                YES, TELL USER
     EXEC CICS DELAY INTERVAL(000001)
  BR 10
*
* SPLBUSY5 EQU *
  MVI SPLREQOK,C'N'          SAY NOT SUCCESSFUL
  MVC SPLPARAM,SPLBSYX       MOVE MESSAGE
  B RETURN
*
* INVCMD EQU *
  MVI SPLREQOK,C'N'          SAY NOT SUCCESSFUL
  MVC SPLPARAM,INVCMDX       MOVE MESSAGE
  B RETURN
*
ERROR EQU *
  CLI EIBRESP+3,X'58'
  BE SPLBUSY
  MVI SPLREQOK,C'N'          SAY NOT SUCCESSFUL
  HEX2CHAR EIBFN,XEIBFN,6,7,8,10
  HEX2CHAR EIBRESP,XEIBRESP,6,7,8,10
  HEX2CHAR EIBRESP2,XEIBRESP2,6,7,8,10
EXEC CICS RETURN
DS DF
PARMPTR DS F
OUTDESCR DS H
HALF8Ø DC H'8Ø'
SPLWORK DS CL8Ø
FULL5 DC F'5'
D_NODE DC CL8'''
D_USERID DC CL8'''
O_CLASS DC C'A'
DS DF
SPLBSYX DC CL8Ø'<= REQUEST CANCELLED, SPOOL THREAD BUSY - 5 TRIES.'*
INVCMDX DC CL8Ø'<= INVALID SPLCMD..SPOOLOPEN,SPOOLWRITE,SPOOLCLOSE'
* ERROR MAPPING
ERRORX DC CL16'<- ERROR EIBFN-'
XEIBFN DC CL14'XXXX EIBRESP-'
XEIBRESP DC CL19'XXXXXXXX EIBRESP2-'
XEIBRES2 DC CL8Ø'XXXXXXXX'
DC CL23 '
*
SPLD DSECT
DS DF
SPLCMD DS CL10
SPLITOKEN DS CL8
SPLPARM DS CL80
SPLCLASS DS C
SPLREQOK DS C
END

SPOOLTST

ID DIVISION.
PROGRAM-ID. SPOOLTST.
*****************************************************************
* DESCRIPTION:
*
* SAMPLE PROGRAM TO SIMPLIFY CICS SPOOL FUNCTIONS.
*
* CALL 'SPOOLPGM' TO PROCESS SPOOL FUNCTIONS.
* IE SPOOLOPEN, SPOOLWRITE, AND SPOOLCLOSE
*
* PARAMETER LIST (COMMAREA)
* SPLCMD : REQUESTED FUNCTION
* SPLTOKEN : TOKEN USED BY JES
* SPLPARM : FUNCTION - SUB-PARAMETERS:
* SPOOLOPEN? - OUTPUT FORMNAME, JES REMOTE ID
* SPOOLWRITE? - OUTPUT LINE
* SPOOLCLOSE? - NOT USED
* NOTE: SPLPARM ALSO USED TO RETURN MESSAGES
* SPLCLASS : OUTPUT CLASS
* SPLREQOK : RETURN REQUEST SUCCESS (Y OR N)
*
*****************************************************************
ENVIRONMENT DIVISION.
EJECT
DATA DIVISION.
WORKING-STORAGE SECTION.
*****************************************************************
*** SPOOL PARMS
*****************************************************************
Ø1 SPOOL-PARM.
Ø3 SPLCMD                  PIC X(10).
Ø3 SPLTOKEN                PIC X(8) VALUE SPACE.
Ø3 SPLPARM                 PIC X(80) VALUE SPACE.
Ø3 SPLCLASS                PIC X VALUE 'X'.
Ø3 SPLREQOK                PIC X VALUE SPACE.
   88 OK                   VALUE 'Y'.
   88 NOTOK                VALUE 'N'.
Ø1 SPOOL-ODESCR-PARM.
Ø3 OUT-DESCR-FORM      PIC X(15) VALUE 'FORMS(OUTX)    '.
Ø3 OUT-DESCR-DEST      PIC X(15) VALUE 'DEST(RMT12345) '.
*
77 MSG-LEN                  PIC S9(4) COMP VALUE +80.
*
*****************************************************************
* DUMMY PRINT LINES FOR TEST
*****************************************************************
Ø1 TEST-PRINT-LINE1.
Ø3 TEST-LINE1             PIC X(80) VALUE 'LINE 1'.
Ø1 TEST-PRINT-LINE2.
Ø3 TEST-LINE2             PIC X(80) VALUE 'LINE 2'.
*
PROCEDURE DIVISION.
Ø00-START.

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
MOVE 'SPOOLOPEN ' TO SPLCMD.
MOVE 'X' TO SPLCLASS.
MOVE SPOOL-ODESCR-PARM TO SPLPARM.
PERFORM SPOOL-PROGRAM.
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
MOVE 'SPOOLWRITE' TO SPLCMD.
MOVE TEST-PRINT-LINE1 TO SPLPARM.
PERFORM SPOOL-PROGRAM.

* * * * * * * * * * * * * * * * * * * * * * * *
MOVE 'SPOOLWRITE' TO SPLCMD.
MOVE TEST-PRINT-LINE2 TO SPLPARM.
PERFORM SPOOL-PROGRAM.

* * * * * * * * * * * * * * * * * * * * * * * *
MOVE 'SPOOLCLOSE' TO SPLCMD.
PERFORM SPOOL-PROGRAM.

* * * * * * * * * * * * * * * * * * * * * * * *
999-RETURN.
EXEC CICS RETURN END-EXEC.
*
* SPOOL PROGRAM CALL
*
SPOOL-PROGRAM.
EXEC CICS LINK PROGRAM('SPOOLPGM')
COMMAREA(SPOOL-PARM) LENGTH(100) END-EXEC.
IF NOTOK
GO TO ERROR-ROUTINE.
*
* DISPLAY MESSAGE RETURNED BY 'SPOOLPGM'
*
ERROR-ROUTINE.
EXEC CICS SEND TEXT FROM(SPLPARM)
LENGTH(MSG-LEN)
ERASE
FREEKB
END-EXEC.
GO TO 999-RETURN.

HEX2CHAR
MACRO
&LABL	HEX2CHAR &HEX,&CHAR,&R1,&R2,&R3,&BALR
.*
.* MACRO TO CONVERT HEX BYTE(S) TO CHARACTER(S) FOR DISPLAY
.* HEX PARAMETER IS INPUT FIELD OF HEX DATA
.* CHAR PARAMETER IS OUTPUT AREA, LENGTH MUST BE L'HEX * 2
.* BALR PARAMETER IS A REGISTER USED FOR RETURN
.* AFTER THE FIRST CALL, THIS MACRO DOES NOT GET EXPANDED
.*
GBLA &C
&C SETA &C+1

AIF (K'&LABL EQ Ø).NOLABL
&LABL EQU *
.NOLABL ANOP
AIF (K'&HEX NE Ø).P1OK
MNOTE 8, 'PARAMETER ONE MISSING'
MEXIT
.P1OK ANOP
AIF (K'&CHAR NE Ø).P2OK
MNOTE 8, 'PARAMETER TWO MISSING'
MEXIT
.P2OK ANOP
AIF (K'&R1 NE Ø).P3OK
MNOTE 8, 'PARAMETER TRE MISSING'
MEXIT
.P3OK ANOP
AIF (K'&R2 NE Ø).P4OK
MNOTE 8, 'PARAMETER FOR MISSING'
MEXIT
.P4OK ANOP
AIF (K'&R3 NE Ø).P5OK
MNOTE 8, 'PARAMETER FIV MISSING'
MEXIT
.P5OK ANOP
AIF (K'&BALR NE Ø).P6OK
MNOTE 8, 'PARAMETER SIX MISSING'
MEXIT
.P6OK ANOP
BAL &BALR,$H2C&C.BGN
B $H2C&C.END
$H2C&C.BGN STM &R1,&R3,$H2CSAVE
LA &R1,&HEX
LA &R2,&CHAR
LA &R3,L'&HEX
$H2C&C.MVC MVC Ø(1,&R2),Ø(&R1)
LA &R2,1(,&R2)
MVC Ø(1,&R2),Ø(&R1)
LA &R2,1(,&R2)
LA &R1,1(,&R1)
BCT &R3,$H2C&C.MVC
LA &R2,&CHAR
LA &R3,L'&HEX
$H2C&C.TRN TRN Ø1(1,&R2),$H2CTBL1
LA &R2,1(,&R2)
TR Ø1(1,&R2),$H2CTBL2
LA &R2,1(,&R2)
BCT &R3,$H2C&C.TRN
LM &R1,&R3,$H2CSAVE
BR &BALR
DS ØF
AIF (&C GT 1).NOCODE
Dealing with program abends

This program gets control whenever CICS detects a program abend. It then displays the transaction ID under which the program was running, along with the abending program name. If a LINKed or XCTLed program abends, that program name is also displayed. It will also send a Wizard Mail message to the person responsible for the abending program.

Assemble and LNKEDT this program as you would any other Assembler CICS command-level program. There are two generation options, specified in the

'&ABOV SETC 'Y''

and the

'&WIZM SETC 'Y''

shown below.

* $$ JOB JNM=DFHPEP,CLASS=Ø,DISP=D,PRI=8,USER=*BOTSIS*
* $$ LST DISP=H,PRI=8,CLASS=0
// JOB DFHPEP DPØØ ASSEMBLE/CATALOG DFHPEP.
// ON $ABEND OR $CANCEL GOTO SKIP3
// EXEC DTRIATTN,PARM='L LST,*DFHPEP'
* $$ SLI ICCF=(DPMACH),LIB=(Ø2)
/* SETPARAM MACHINE=PROD
PEP     TITLE 'CUSTOMER INFORMATION CONTROL SYSTEM PROGRAM ERROR PROGRAM'

*     GBLC &ABOV
     GBLC &WIZM

*     &ABOV SETC 'Y'            SET TO 'N' IF YOU WANT 24-BIT USAGE.
     &WIZM SETC 'Y'              SET TO 'N' IF NOT USING WIZARD MAIL.

*     AIF ('&ABOV' EQ 'Y').ABOVØØ
     AIF ('&ABOV' EQ 'N').ABOVØØ
     MNOTE 4,'VALUE FOR ''ABOV'' NOT ''Y'' OR ''N'', FORCED TO ''Y''X
     &ABOV SETC 'Y'
     .ABOVØØ ANOP

*     AIF ('&WIZM' EQ 'Y').WIZMØØ
     AIF ('&WIZM' EQ 'N').WIZMØØ
     MNOTE 4,'VALUE FOR ''WIZM'' NOT ''Y'' OR ''N'', FORCED TO ''Y''X
     &WIZM SETC 'Y'
     .WIZMØØ ANOP

You may wish to change these before assembly/LNKEDT:

• If you don’t wish to run this program above the line (ie in 31-bit
  mode) or if you aren’t using high-level Assembler (ie ASMA90),
  specify:

  '&ABOV SETC 'N'.

• If you don’t have Wizard Mail or don’t wish to use it, specify:

  '&WIZM SETC 'N'.

Specifying ‘N’ will still cause a message to be displayed on the system
console whenever a program abends.

The assembly and LNKEDT of this program should end with a $RC
of zero. If the $RC value is any different, you should examine the
assembly listing, determine the problem, fix it, and reassemble the program. Note that, if you specify any value other than ‘Y’ or ‘N’ in the above ‘SETC’ statements, you will receive a $RC value of four.

It is highly recommended, though not essential, that this program be catalogued into a LIB.SUBLIB other than an IBM one. This will allow you to simply delete the .PHASE MEMBERTYPE, and ‘CEMT SET PROG(DFHPEP) NEW’ the program, in which case the original IBM program will be used, if there are any problems. The LIB.SUBLIB into which you choose to catalogue the program must be LIBDEFed ahead of the LIB.SUBLIB in which the original IBM version resides in your CICS start-up JCL.

This program requires no special PPT entry to be added to your RDO (ie DFHCSD) or your macro PPT, as there should already be an entry present. However, if you have H&W Systems’ Wizard Mail, and you wish to send a Wizard Mail to the person responsible for the abending program so that action can be taken to fix it, a PCT entry for ‘DPWI’ and a PPT entry for ‘DPWIZM’ need to be added to either the RDO (ie DFHCSD) or to your macro PCT/PPTs.

Before you assemble/LNKEDT this program, you must also assemble and catalogue the DPEIBC subroutine, which is called by this program. The subroutine must be catalogued as an .OBJ MEMBERTYPE. The LIB.SUBLIB into which the subroutine is catalogued must be LIBDEFed when you assemble and catalogue this program.

NOTES
You should note that:

• This program was taken from IJSYSRS.SYSLIB (ie DFHPEP.A) and modified. One of the modifications was the conversion from macro-level to command-level. If you need to know what the original program looked like, see the one contained in IJSYSRS.SYSLIB.

• This program is currently running on a CICS/VSE 2.3 system, but it has also run on a CICS/VSE 2.2 system. No changes were made to migrate it from CICS/VSE 2.2 to CICS/VSE 2.3.
• This program contains installation-dependent code. This code checks for such things as applids, and assumes that the first two characters of the transaction ID adhere to installation standards. This code is only applicable if you use Wizard Mail or another mainframe e-mail package. Since it is unlikely that these applids and transaction IDs are identical to those in your installation, some of the code contained within the program will need to be changed. There are comments contained within the code to help you make the necessary modifications, and all of the relevant statements contain three equals signs (ie ===) in columns 69-71 to help you identify them. It would be helpful if you had a little knowledge of Assembler.

• This program cannot control the information that is sent to the on-line problem determination file. There are comments within the code to help you limit what is sent.

• Even though this program is set up to use Wizard Mail, it could easily be modified to use any other mainframe e-mail package as long as it has an application interface (ie a means to invoke and send an e-mail via a CICS application program).

• This program is still useful even if you don’t have Wizard Mail or don’t wish to use it. This is because it will inform you whenever a program abends, by displaying the abending transaction ID along with the program that abended. You can then take the appropriate action.

```
DFHPEP

PCTCBAR EQU 8               PCT BASE REGISTER
TCASBAR EQU 9               TCA SYSTEM AREA REGISTER/WIZCOMM.
*
DFHEISTG DSECT
RESP DS F                   RESPONSE CODE. (FROM EXEC CICS COMMA
SPACE DS C                  GETMAIN INITIALIZE FIELD.
OPID DS CL3                 OPID FROM ASSGN COMMAND.
USERID DS CL8               USERID FROM ASSGN COMMAND.
APPLID DS CL8               APPLID FROM ASSGN COMMAND.
WRTSAVE DS CL65             EXEC CICS WRITE OPERATOR SVE AREA.
*
DFHREGS ,                   USE CICS REGISTER EQUATES.
DFHTCA CICSYST=YES ,        SYSTEM TCA.
```
COPY DFHPCTDS PCT DSECT.
DFHTACB TYPE=DSECT, ABNORMAL TERMINATION DSECT.
AIF ('&WIZM' NE 'Y').WIZMØ1
*
WIZCOMM DSECT WIZARD MAIL DSECT.
DS ØCL15ØØ
WIZERRM DS CL6Ø WIZMAIL MESSAGE HEADER.
WIZSUBJ DS CL25 WIZMAIL MESSAGE SUBJECT.
WIZCONF DS C WIZMAIL CONFIRMATION CODE.
WIZDIS1 DS CL7Ø WIZMAIL DISTRIBUTION LINE ONE (1).
WIZDIS2 DS CL7Ø WIZMAIL DISTRIBUTION LINE TWO (2).
WIZDIS3 DS CL7Ø WIZMAIL DISTRIBUTION LINE THREE (3).
WIZDIS4 DS CL7Ø WIZMAIL DISTRIBUTION LINE FOUR (4).
WIZMSG1 DS CL7Ø WIZMAIL MESSAGE LINE ONE (1).
WIZMSG2 DS CL7Ø WIZMAIL MESSAGE LINE TWO (2).
WIZMSG3 DS CL7Ø WIZMAIL MESSAGE LINE THREE (3).
WIZMSG4 DS CL7Ø WIZMAIL MESSAGE LINE FOUR (4).
WIZMSG5 DS CL7Ø WIZMAIL MESSAGE LINE FIVE (5).
WIZMSG6 DS CL7Ø WIZMAIL MESSAGE LINE SIX (6).
WIZMSG7 DS CL7Ø WIZMAIL MESSAGE LINE SEVEN (7).
WIZMSG8 DS CL7Ø WIZMAIL MESSAGE LINE EIGHT (8).
WIZMSG9 DS CL7Ø WIZMAIL MESSAGE LINE NINE (9).
WIZMSGA DS CL7Ø WIZMAIL MESSAGE LINE TEN (10).
WIZMSGB DS CL7Ø WIZMAIL MESSAGE LINE ELEVEN (11).
WIZMSGC DS CL7Ø WIZMAIL MESSAGE LINE TWELVE (12).
WIZMSGD DS CL7Ø WIZMAIL MESSAGE LINE THIRTEEN (13).
WIZMSGE DS CL7Ø WIZMAIL MESSAGE LINE FOURTEEN (14).
WIZMSGF DS CL7Ø WIZMAIL MESSAGE LINE FIFTEEN (15).
WIZOPID DS CL8 WIZMAIL ORIGINATING OPERATOR ID.
DS CL76 NOT USED.
.WIZMØ1 ANOP
*
DFHPEP DFHEIENT CODEREG=(R5,RA),DATAREG=(R6),EIBREG=(R4)
AIF ('&ABOV' NE 'Y').ABOVØ1
DFHPEP AMODE 31
DFHPEP RMODE ANY
.ABOVØ1 ANOP
* THE CODE FROM HERE TO LABEL 'NEXT11', INCLUDING THE FIRST INSTRUCTION
* AT THAT LABEL, DEALS WITH ACQUIRING THE ABEND INFORMATION AND THE
* BUILDING AND DISPLAYING OF THE SYSTEM CONSOLE MESSAGE.
MVC WRTAREA,WRTAREA-1 CLEAR WRTAREA.
MVI SPACE,C' ' SET SPACE.
MVC SVUTCA,Ø(TCACBAR) SVE FIRST 252 BYTES OF USER TCA.
L TCASBAR,TCASYAA LOAD SYSTEM TCA ADDRESS.
USING DFHSYTCA,TCASBAR INFORM ASSEMBLER.
MVC ONE,236(TCACBAR) SVE SOME OF THE SYSTEM TCA AREA.
MVC XCAPCACB,TCAPCACB SVE FIRST 252 BYTES OF ABNORMAL TERM
L R3,TCAPCACB LOAD ADDRESS OF ABNORMAL TERMINATION
ST R3,SVR3 SVE ADDRESS OF IT.
USING DFHABND,R3 INFORM ASSEMBLER.
MVC PROG11,ABNDPRG   MVE ABEND PROGRAM NAME. (X'20').
MVC ABSYSID,ABNDSYST SVE ABEND SYSID. (X'34').
DROP R3               (DFHABND).
MVC SVSTCA,Ø(TCASBAR) SVE FIRST 252 BYTES OF SYSTEM TCA.
MVC TWO,236(TCASBAR)  SVE SOME OF THE SYSTEM TCA AREA.
L PCTCBAR,TCATCPC LOAD ADDRESS OF PCT ADDRESS.
ST PCTCBAR,SVR8    SVE ADDRESS OF IT.
USING DFHPCTDS,PCTCBAR INFORM ASSEMBLER.
MVC SVPCTA,Ø(PCTCBAR) SVE FIRST 252 BYTES OF PCT.
MVC PROG1,PCTIPIA   MVE PCT PROGRAM NAME. (X'14').
MVC PROG2,PCTIPIA   ...
MVC SYSID,PCTSYSID  MVE PCT SYSID. (X'60').
DROP PCTCBAR        (DFHPCTDS).
EXEC CICS ASSIGN OPID(OPID) USERID(USERID) APPLID(APPLID) X
NOHANDLE.           GET SOME INFO.
MVC TERM1,EIBTRMID  MVE TERMINAL ID.
MVC TERM2,EIBTRMID  ...
MVC APPL1,APPLID    MVE APPLID.
MVC APPL2,APPLID    ...
MVC TRAN1,TCAKCOID  MVE TRANSACTION ID. (X'B0').
MVC TRAN2,TCAKCOID  ...
MVC CODE1,TCAPCAC   MVE ABEND CODE. (X'8C').
MVC CODE2,TCAPCAC   ...
DROP TCASBAR        (DFHSYTCA).
CLI PROG11,C'A'     IS THERE A VALID SECOND PROGRAM NAME
BL    NEXT1          NO-BRANCH TO NEXT1.
CLC   PROG1,PROG11   ARE FIRST AND SECOND PROGRAM NAMES E
BE    NEXT1          YES-BRANCH TO NEXT1.
MVC WRTAREA,MESSG1  MVE MESSG1.
MVC WRTSAVE,MESSG1  ...
B    NEXT11         BRANCH TO NEXT11.

* NEXT1 EQU *
MVC WRTAREA,MESSG2  MVE MESSG2.
MVC WRTSAVE,MESSG2  ...

* NEXT11 EQU *
BAL RB,WTOC     PERFORM WTOC ROUTINE.
AIF ('&WIZM' NE 'Y').WIZMØ3
BAL RB,WIZM     PERFORM WIZM ROUTINE.
.WIZMØ3 ANOP
* THE NEXT FOUR (4) INSTRUCTIONS DEAL WITH RUNNING MRO/ISC. IF THE
* TRANSACTION ID IS 'IEMD' AND THE ABEND CODE IS 'ATCH' WE RETURN
* TO CICS SKIPPING ANY FURTHER PROCESSING, INCLUDING THAT FOR ON-LINE
* PROBLEM DETERMINATION. IRRESPECTIVE OF WHETHER OR NOT YOU ARE USING
* MRO/ISC YOU CAN ADD YOUR OWN INSTRUCTIONS, FOR ANY GIVEN TRANSACTION
* ID OR DUMP CODE, THUS LIMITING WHAT IS SENT TO THE ON-LINE PROBLEM DE-
* TERMINATION FILE.
CLC =C'IEMD',TRAN1  IS ABENDED TRANSACTION 'IEMD'.  ===
BNE NEXT3      NO-BRANCH TO NEXT3.  ===
CLC =C'ATCH',CODE1 IS ABEND CODE 'ATCH'.

BE RETURN YES-BRANCH TO RETURN.

* NEXT3 EQU *

CLC =C'AZ',CODE1 DOES ABEND CODE BEGIN WITH 'AZ'.

BNE NEXT4 NO-BRANCH TO NEXT4.

CLI SYSID,C'A' IS THERE A VALID SYSID.

BL NEXT4 NO-BRANCH TO NEXT4.

MVC WRTAREA,MESSG3 MVE MESSG3.

BAL RB,WTOC PERFORM WTOC ROUTINE.

* WHEN WE ARE ALL DONE WE RETURN HERE TO LET THE IUI FURTHER PROCESS
* THE ABEND (IE. PROVIDE INFORMATION TO THE ON-LINE PROBLEM DETERMI-
* NATION FILE/Screens).

NEXT4 EQU *

* CLC =C'ICCFDEVM',APPL1
* BE NEXT5
* CLC =C'ICCFTOR ',APPL1
* BNE RETURN

* EXT5 EQU *

EXEC CICS XCTL PROGRAM('IESOPDC') NOHANDLE. XCTL TO IESOPDC.

* RETURN EQU *

EXEC CICS RETURN. RETURN TO CICS.

* WTOC EQU *

EXEC CICS WRITE OPERATOR TEXT(WRTAREA) X
TEXTLENGTH(L'WRTAREA). SEND TO MESSAGE TO CONSOLE.
MVC WRTAREA,WRTAREA-1 CLEAR WRTAREA.
BR RB RETURN TO CALLER.
AIF ('&WIZM' NE 'Y').WIZMØ7

* HERE WE CHECK TO SEE IF THE ABENDING PROGRAM WAS INVOKED FROM THE
* SYSTEM CONSOLE AND IF SO WE BYPASS THE SENDING OF A WIZMAIL MES-
* SAGE.

WIZM EQU *

EXEC CICS SEND WIZARD MAIL ROUTINE.

CLC =C'CNSL',EIBTRMID ARE WE RUNNING FROM CNSL.

BER RB YES-RETURN TO CALLER.

* HERE WE CHECK TO SEE IF THE ABENDING PROGRAM WAS RUNNING IN A PART-
* TICULAR CICS, IN THIS CASE CICSPRD1 AND CICSPRD3 (AORS) AND IF NOT
* WE BYPASS THE SENDING OF A WIZMAIL MESSAGE. CHANGE THE FOLLOWING CODE
* TO YOUR OWN APPLIDS.

* CLC =C'CICSTEST',APPL1 ARE WE RUNNING IN CICSTEST.
* BE WIZM3 YES-BRANCH TO WIZM3.

* CLC =C'CICSPRD1',APPL1 ARE WE RUNNING IN CICSPRD1.
* BE WIZM3 YES-BRANCH TO WIZM3.

* CLC =C'CICSPRD2',APPL1 ARE WE RUNNING IN CICSPRD2.
* BER RB YES-RETURN TO CALLER.

* BE WIZM3 YES-BRANCH TO WIZM3.

* CLC =C'CICSPRD3',APPL1 ARE WE RUNNING IN CICSPRD3.

BNER RB NO-RETURN TO CALLER.

* Here we check for certain ABEND codes, Transaction IDs and programs that we do not want to send a WIZMAIL message.

WIZM3

EQU *

CLC =C'0OPS',CODE1 WAS ABEND CODE 'OOPS'.
BER RB YES-RETURN TO CALLER.

CLC =C'DFH',PROG1 WAS ABEND PROGRAM 'DFH....'.
BER RB YES-RETURN TO CALLER.

CLC =C'DFH',PROG11 WAS ABEND PROGRAM 'DFH....'.
BER RB YES-RETURN TO CALLER.

CLC =C'APCT',TRAN1 WAS TRANSACTION 'APCT'.
BER RB YES-RETURN TO CALLER.

CLC =C'CUI',TRAN1 WAS TRANSACTION 'CUI'.
BER RB YES-RETURN TO CALLER.

CLC =C'DMSS',TRAN1 WAS TRANSACTION 'DMSS'.
BER RB YES-RETURN TO CALLER.

CLC =C'FAQS',TRAN1 WAS TRANSACTION 'FAQS'.
BER RB YES-RETURN TO CALLER.

CLC =C'WØ4Ø',TRAN1 WAS TRANSACTION 'WØ4Ø'.
BER RB YES-RETURN TO CALLER.

EXEC CICS GETMAIN SET(R9) LENGTH(1500) INITIMG(SPACE).
USING WIZCOMM,R9 INFORM ASSEMBLER.

ST R9,SVR9 SVE REG 9.

MVC WIZSUBJ(16),=C'Program Abend '
MVI WIZCONF,C'N' INDICATE NOT CONFIDENTIAL.

MVC WIZMSG1(61),=C'A CICS application program you are responX
sible for abended in'
MVC WIZMSG1+62(L'APPL1),APPL1
MVC WIZMSG3(35),=C'In your attempt to determine the ca'
MVC WIZMSG4(35),=C'use OLPD in both ICCFDEV and in th'
MVC WIZMSG4+35(35),=C'e above mentioned CICS.
MVC WIZMSG6(35),=C'If there is more than one program l'
MVC WIZMSG6+35(35),=C'isted below, the second is the one '
MVC WIZMSG7(35),=C'that abended. The first one XCTL''ed,'
MVC WIZMSG7+35(35),=C' or LINK''ed to the second. '
MVC WIZMSG9(L'WRTSAVE),WRTSAVE
MVC WIZMSGA(35),=C'OPID=xxx,USERID=xxxxxxx'
MVC WIZMSGA+5(L'OPID),OPID
MVC WIZMSGA+16(L'USERID),USERID
ST RB,SVRB SVE REG 11.
BAL RB,WIZMD PERFORM WIZMD ROUTINE.

L RB,SVRB RESTORE REG 11.
MVC WIZOPID,=CL8'WIZARD' SET OPID TO WIZARD.

* CLC =C'CICSPRD2',APPL1 ARE WE RUNNING IN CICSPRD2.
* BNE WIZM3C NO-BRANCH TO WIZM3C.
* CLC =C'BIM',PROG1 WAS ABEND PROGRAM 'BIM....'.
* BER RB YES-RETURN TO CALLER.
* EXEC CICS START TRANSID('DPWI') FROM(WIZCOMM) LENGTH(1500) X
* RESP(RESP).
* EXEC CICS LINK PROGRAM('WMPAIM1') COMMAREA(WIZCOMM) X
LENGTH(1500) RESP(RESP).
*       B       WIZM3G       BRANCH TO WIZM3G.

WIZM3C EQU *
EXEC      CICS START TRANSID('DPWI') FROM(WIZCOMM) LENGTH(1500) X
         RESP(RESP).

WIZM3G EQU *
CLC      RESP,DFHRESP(NORMAL) WAS RESP NORMAL.
BNE      WIZM8       NO-BRANCH TO WIZM8.

WIZMD EQU *
*       WIZMD ROUTINE.
*       HERE WE SET A DEFAULT WIZARD MAIL DISTRIBUTION NAME AND SET THE OTHER
*       DISTRIBUTION FIELDS TO BLANKS.
       MVC      WIZDIS1(35),=C'RBOTSIS       '    ===
       MVC      WIZDIS1+35(35),=C'                                   '
       MVC      WIZDIS2(35),=C'                                   '
       MVC      WIZDIS2+35(35),=C'                                   '
       MVC      WIZDIS3(35),=C'                                   '
       MVC      WIZDIS3+35(35),=C'                                   '
       MVC      WIZDIS4(35),=C'                                   '

HERE WE CHECK IF THE ABENDING PROGRAM WAS RUNNING IN CICSPRD3 OR IF THE ABENDING PROGRAMS TRANSACTION ID BEGINS WITH 'DP'. IF SO WE EXIT AS THERE IS NO NEED TO LOOK ANY FURTHER BECAUSE WE HAVE COMPLETED SETTING UP THE WIZARD MAIL FIELDS.

CLC =C'CICSPRD3',APPL1 ARE WE RUNNING IN CICSPRD3. ----- 
BER RB YES-RETURN TO CALLER. ----- 
CLC =C'DP',TRAN1 IS THIS ANY 'DP' TRANSACTION. ----- 
BER RB YES-RETURN TO CALLER. ----- 
* CLC =C'CICSPRD2',APPL1 ARE WE RUNNING IN CICSPRD2. ----- 
* BNE WIZMD1 NO-BRANCH TO WIZMD1. ----- 
* MVC WIZDIS1(35),=C'KHOFFMAN' ----- 
* BR RB RETURN TO CALLER. 
* HERE WE LOOK FOR THREE (3) TRANSACTION IDS AND IF ONE OF THE THREE (3) WE BYPASS LOOKING FOR THEM IN THE TABLE. IF WE DON'T FIND THEM WE CONTINUE.

WIZMD1 EQU * 
CLC =C'MENU',TRAN1 IS THIS 'MENU' TRANSACTION. ----- 
BE WIZMD1A YES-BRANCH TO WIZMD1A. ----- 
CLC =C'JSNT',TRAN1 IS THIS 'JSNT' TRANSACTION. ----- 
BE WIZMD1A YES-BRANCH TO WIZMD1A. ----- 
CLC =C'NTST',TRAN1 IS THIS 'NTST' TRANSACTION. ----- 
BNE WIZMD1C NO-BRANCH TO WIZMD1C. ----- 
*

WIZMD1A EQU * 
MVC WIZDIS1(35),=C'KHOFFMAN' ----- 
BR RB RETURN TO CALLER. 
* HERE WE LOOK FOR ONE OTHER TRANSACTION ID BEFORE LOOKING IN THE TABLE.

WIZMD1C EQU * 
CLC =C'E9',TRAN1 IS THIS ANY 'E9' TRANSACTION. ----- 
BE WIZMD1F NO-BRANCH TO WIZMD1F. ----- 
MVC WIZDIS1(35),=C'GNICHOLAS' ----- 
BR RB RETURN TO CALLER. 
*

WIZMD1F EQU * 
LA RE,TABLE LOAD ADDRESS OF TABLE TO REG 14. 
* HERE WE LOOK UP THE TRANSACTION ID AGAINST THE TABLE. IF WE FIND A MATCH WE MOVE THE DISTRIBUTION NAME SO WE CAN SEND THE WIZARD MAIL MESSAGE TO THE RESPONSIBLE PERSON. IF WE DO NOT FIND A MATCH THE WIZARD MAIL WILL BE SENT TO A DEFAULT PERSON (IE RBOTSIS).

WIZMD1I EQU * 
CLC Ø(2,RE),TRAN1 DO WE HAVE A MATCH. 
BE WIZMD9 YES-BRANCH TO WIZMD9. 
LA RE,'L'TABLE(RE) INCREMENT TO NEXT TABLE POSITION. 
CLI Ø(RE),X'FF' ARE WE AT THE OF THE TABLE. 
BER RB YES-RETURN TO CALLER. 
B WIZMDII BRANCH TO WIZMDII. 
*

WIZMD9 EQU *
MVC WIZDIS1(35),2(RE) MVE DISTRIBUTION PERSON/S.
BR RB RETURN TO CALLER.

.WIZMØ7 ANOP

* ABSYSID DS CL8
EIBFN1 DS XL8
EIBOUT1 DS CL17
EIBFN2 DS XL8
EIBOUT2 DS CL17

* DC C' ' DON'T MOVE/REMOVE THIS STATEMENT.
WRTAREA DC CL65 ' ' WRITE AREA.
MESSG1 DS ØCL65
DC CL6'ABEND='
CODE1 DC CL4' ' DC CL5' ,TRN='
TRAN1 DC CL4' ' DC CL5' ,PGM='
PROG1 DC CL8' ' DC C'/'
PROG11 DC CL8' ' DC CL5' ,TRM='
TERM1 DC CL4' ' DC CL5' ,APL='
APPL1 DC CL8' ' DC C' ' MESSG2 DS ØCL65
DC CL6'ABEND='
CODE2 DC CL4' ' DC CL5' ,TRN='
TRAN2 DC CL4' ' DC CL5' ,PGM='
PROG2 DC CL8' ' DC CL5' ,TRM='
TERM2 DC CL4' ' DC CL5' ,APL='
APPL2 DC CL8' ' DC CL11' ' MESSG3 DS ØCL65
DC CL11'FROM SYSID='
SYSID DS CL8' ' DC CL46' ' DS ØD
SVR3 DS F
SVR8 DS F
SVR9 DS F
SVR8 DS F
SAVEAREA DS 18F

AIF ('&WIZM' NE 'Y').WIZMØ9

* THIS IS THE TABLE THAT IS USED TO DETERMINE WHO TO SEND THE WIZARD
* MAIL MESSAGE TO SHOULD A PROGRAM ABEND. THE FIRST TWO (2) BYTES OF
* THE TABLE CONTAIN THE FIRST TWO (2) BYTES OF THE TRANSACTION ID,
* WHICH IS SET UP BY SYSTEM NAME (IE MS=MISC., JS=JUSTICE). THE RE-
* MAINER CONTAINS THE DISTRIBUTION NAME/S. NOTE: AS YOU CAN SEE SOME
* WIZARD MAIL MESSAGES ARE SENT TO MORE THAN ONE PERSON.

<table>
<thead>
<tr>
<th>TABLE</th>
<th>DS</th>
<th>ØCL37</th>
<th>TRANID(1ST 2 CHARS)/DISTRIBUTION NAME/S.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC</td>
<td>C'</td>
<td>MSKHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>JSCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>RWKHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>EQKHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TPVSENECAL</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TSKHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>RSCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>BDSCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>CVCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>HDCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>IACRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>JUCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>JVCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>PRCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>SPCRICHARDSON KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TKJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TRJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>STJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>ACJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>PYJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>PEJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>PUJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>DRJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>RDJHADL VSENECAL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>EMJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>ANJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>CLJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>HRJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>VAJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>VEJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>VNJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>VTJHADL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TAJHADL VSENECAL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TQJHADL VSENECAL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TXJHADL VSENECAL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>TØJHADL VSENECAL KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FARCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FCRCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FHRCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FIRCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FNRCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FORCWIK KHOFFMAN</td>
<td></td>
</tr>
<tr>
<td>DC</td>
<td>C'</td>
<td>FPRCWIK KHOFFMAN</td>
<td></td>
</tr>
</tbody>
</table>
| DC    | X'FF'| END OF TABLE. DON'T MOVE/REMOVE.
.WIZMØ9 ANOP
*       DS  ØD           ALIGN ON DOUBLE WORD BOUNDARY.
*       DS  CL4
DC     C'ONE?'
ONE    DS  CL4
DC     C'TWO?'
TWO    DS  CL4
DC     C'UTCA'
SVUTCA DS  CL252
DC     C'STCA'
SVSTCA DS  CL252
DC     C'PCTA'
SVPCTA DS  CL252
DC     C'XCAPCACB'
XCAPCACB DS  CL252
*
LTORG
*
END   DFHPEP
/*
  CLOSE SYSPCH,FED
// IF $RC > 13 THEN
// GOTO SKIPØ
// DLBL IJSYSIN,'DP.SYSPCH.FILE.CPU—.=='
// EXTENT SYSIPT,DOSRES
  ASSGN SYSIPT,DISK,VOL=DOSRES,SHR
* $$ SLI ICCF=(DPVLBL),LIB=(Ø2)
// OPTION CATAL,LIST
  PHASE DFHPEP,*
    INCLUDE DFHEAI
// EXEC ASMA9Ø,SIZE=(ASMA9Ø,512K),PARM='SIZE(MAX-200K,ABOVE),EXIT(LIBEXX
  IT(EDECKXIT))'
/* EXEC ASMA9Ø,SIZE=(ASMA9Ø,64K),PARM='SIZE(MAX-200K,ABOVE)'
  CLOSE SYSIPT,READER
// IF $RC > 4 THEN
// GOTO SKIP1
// GOTO SKIP3 ????????
// EXEC LNKEDT,SIZE=256K,PARM='AMODE=31,RMODE=ANY'
// EXEC DPCOMD,SIZE=DPCOMD,PARM='1YICF    1ØDFHPEP,DES=TEST,O="DPCOMD",X
  X=N,D=D' DPCOMD
// IF MACHINE=PROD THEN
// GOTO PROD5Ø
// EXEC JCLBCICS
  / ID J=ICCFCICS
  CEMT S PROG(DFHPEP) NEW ENA
/*
// EXEC JCLBCICS
  / ID J=CICSTEST
  CEMT S PROG(DFHPEP) NEW ENA
/*
// GOTO SKIP3
/>. PROD5Ø
// IF MACHINE=TEST THEN
// GOTO SKIP3
// EXEC JCLBCICS
/>. ID J=ICCFDEVM
 CEMT S PROG(DFHPEP) NEW ENA
/*
// EXEC JCLBCICS
/>. ID J=CICSPRD1
 CEMT S PROG(DFHPEP) NEW ENA
/*
// EXEC JCLBCICS
/>. ID J=CICSPRD3
 CEMT S PROG(DFHPEP) NEW ENA
/*
// GOTO SKIP3
/>. SKIPØ
 * COMMAND LEVEL INTERPRETER RC > 13, ASSEMBLY OF DFHPEP ABORTED.
// GOTO SKIP3
/>. SKIP1
 * RETURN CODE > 4, PHASE DFHPEP NOT CATALOGUED.
/>. SKIP3
// EXEC DYNUTIL,SIZE=256K
  DELETE 'DP.SYSPCH.FILE.CPU—.==' VOL=DOSRES
/*
// GOTO SKIP99
/>. PROD9Ø
// IF MACHINE=TEST THEN
// GOTO SKIP99
// EXEC DYNUTIL,SIZE=256K *
  DELETE 'DV.SYSPCH.FILE.CPU—.==' VOL=DOSRES
/*
// GOTO SKIP99
/>. SKIP98
/. JUNK99
 * EXECUTION BEING DONE ON INVALID CPU-JOB TERMINATED.
/. SKIP99
/\n * $$ EOJ

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

Robert Botsis
Senior Systems Programmer (USA)  © Xephon 1999
Displaying CPU usage by TCB – part 2

This month we conclude the article that displays CPU usage by TCB in a CICS region.

PROC-PF6.
* SWAP BETWEEN DELTA/TOTAL MODES
  IF SW-MODE = 'T'
    MOVE 'F6-MODE-T' TO TCBPFØ6O
    MOVE 'MODE : DELTA CPU TIME' TO TCBMESSO
    MOVE 'D' TO SW-MODE
  ELSE
    MOVE 'F6-MODE-D' TO TCBPFØ6O
    MOVE 'MODE : TOTAL CPU TIME' TO TCBMESSO
    MOVE 'T' TO SW-MODE
  END-IF
* .

PROC-PF8.
  IF CNTR > ASCB-NUM
    MOVE 1 TO CNTR
  END-IF
  PERFORM FILL-SCREEN
  .

PROC-PF7.
  MOVE CNTS TO CNTR
  SUBTRACT 15 FROM CNTR
  IF CNTR < 1
    MOVE 1 TO CNTR
  * ADD ASCB-NUM TO CNTR
  IF CNTR < 1
    MOVE 1 TO CNTR
  END-IF
  END-IF
  PERFORM FILL-SCREEN
  .

* ————————————————————————————————————
* WE CHECK FOR A SORT/NOSORT COMMAND
* ————————————————————————————————————

* PROC-ENTER.
  MOVE CNTS TO CNTR
  IF TCBCOMMI = SPACES OR
    TCBCOMMI = LOW-VALUES
    PERFORM ACCESS-CALTAB
  IF CNTR < 1
    MOVE 1 TO CNTR
  END-IF
ELSE
  IF TCBCOMMI(Ø1:Ø4) = 'SORT'
    IF TCBCOMMI(Ø6:Ø4) = 'PROG'
      MOVE 'PROGRAM' TO SORT-FLD
      MOVE SPACES TO TCBCOMMI
    ELSE
      IF TCBCOMMI(Ø6:Ø4) = 'ADDR'
        MOVE 'ADDRESS' TO SORT-FLD
        MOVE SPACES TO TCBCOMMI
      ELSE
        IF TCBCOMMI(Ø6:Ø3) = 'CPU'
          MOVE 'CPU TIME' TO SORT-FLD
          MOVE SPACES TO TCBCOMMI
        ELSE
          MOVE 'INVALID SORT FIELD' TO TCBMESSO
        END-IF
      END-IF
    END-IF
  END-IF
PERFORM SORT-TABLE
ELSE
  IF TCBCOMMI(Ø1:Ø6) = 'NOSORT'
    MOVE SPACES TO SORT-FLD
    MOVE SPACES TO TCBCOMMI
    PERFORM ACCESS-CALTAB
  END-IF
END-IF
PERFORM FILL-SCREEN
.
  *
  * 1ST START OF TRAN
  STARTIT.
  *
  EXEC CICS
    SEND CONTROL
    FREEKB
    ERASE
END-EXEC
  *
    MOVE 'CPU TIME' TO SORT-FLD
    MOVE LOW-VALUES TO IPPDTCBI
    MOVE 'CPU USAGE BY TCB FOR THIS CICS REGION' TO TCBTIT10
    MOVE 'IPPCDTCB' TO TCBPROGO
    MOVE 'IPPDTCB' TO TCBMAPNO
    MOVE EIBTRMID TO TCBTERMO
  *
    EXEC CICS ASSIGN NETNAME(TCBNETNO)
      USERID(TCBUSERO)
      APPLID(TCBAPPLO)
END-EXEC
  *
    MOVE 'F3-END' TO TCBPFØ30
* MOVE 'F6-MODE-D' TO TCBFØ60
* MOVE 'F5-DISP-E' TO TCBFØ50
* MOVE 1 TO CNTR
* IF TCBMESSO = SPACES OR
   TCBMESSO = LOW-VALUES
   PERFORM ACCESS-GTCB
   PERFORM FILL-SCREEN
END-IF
*
ACCESS-GTCB.
*
MOVE ASCB-REC-NEW TO ASCB-REC-OLD
MOVE TCBA-REC-NEW TO TCBA-REC-OLD
MOVE LOW-VALUES TO ASCB-REC-NEW
MOVE 256 TO ASCB-MAX-NEW
MOVE LOW-VALUES TO TCBA-REC-NEW
CALL IPPCGTCB USING ASCB-REC-NEW TCBA-REC-NEW.
IF SW-MODE = 'T'
   MOVE ASCB-REC-NEW TO ASCB-REC
   MOVE TCBA-REC-NEW TO TCBA-REC
ELSE
   MOVE ASCB-REC-NEW TO ASCB-REC
   SUBTRACT ASCB-TCB-OLD FROM ASCB-TCB-NEW GIVING ASCB-TCB
   SUBTRACT ASCB-SRB-OLD FROM ASCB-SRB-NEW GIVING ASCB-SRB
   SUBTRACT ASCB-SUM-OLD FROM ASCB-SUM-NEW GIVING ASCB-SUM
   SUBTRACT ASCB-TIM-OLD FROM ASCB-TIM-NEW GIVING ASCB-TIM
   MOVE TCBA-REC-NEW TO TCBA-REC
   PERFORM VARYING CNTR FROM 1 BY 1 UNTIL CNTR > ASCB-NUM-NEW
   SUBTRACT TCB-CPUT-OLD(CNTR) FROM TCB-CPUT-NEW(CNTR) GIVING TCB-CPUT(CNTR)
END-PERFORM
END-IF
IF RETURN-CODE = 4
   MOVE '# OF TCBS EXCEEDS TABLE SIZE' TO TCBMESSO
ELSE
   IF RETURN-CODE = 8
      MOVE 'SOMETHING BAD HAPPENED' TO TCBMESSO
   END-IF
END-IF
IF SORT-FLD NOT = SPACES
   PERFORM SORT-TABLE
END-IF
*
FILL-SCREEN.
*
EXEC CICS ASKTIME ABSTIME(ABSTIME)
END-EXEC
EXEC CICS FORMATTIME ABSTIME(ABSTIME)
  TIME(TCBTIMEO) TIMESEP(':')
  DDMMYYYY(TCBDATEO) DATESEP('/')
END-EXEC

MOVE 'ASCB-TCB/SRB:' TO TCBACTIO(Ø1:13)
MOVE ASCB-TCB TO CPU-WORK
MOVE CPU-WORK TO TCBACTIO(14:1Ø)
MOVE '/' TO TCBACTIO(24:Ø1)
MOVE ASCB-SRB TO CPU-WORK
MOVE CPU-WORK(3:8) TO TCBACTIO(25:8)
MOVE ' SUM:' TO TCBACTIO(33:Ø5)
MOVE ASCB-SUM TO CPU-WORK
MOVE CPU-WORK TO TCBACTIO(38:1Ø)
IF SW-MODE = 'T'
  IF SW-DISP = 'C'
    MOVE 'TOTAL CPU BY TCB PER TOTAL CPU USED' TO TCBTIT1O
  ELSE
    MOVE 'TOTAL CPU BY TCB PER ELAPSED TIME' TO TCBTIT1O
  END-IF
ELSE
  IF SW-DISP = 'C'
    MOVE 'DELTA CPU BY TCB PER TOTAL CPU USED' TO TCBTIT1O
  ELSE
    MOVE 'DELTA CPU BY TCB PER ELAPSED TIME' TO TCBTIT1O
  END-IF
END-IF
MOVE 'FROM : ' TO TCBTIT2O(18:Ø7)
MOVE CNTR TO TCBTIT2O(25:4)
MOVE CNTR TO CNTS
IF CNTR = 1
  MOVE SPACES TO TCBPFØ7O
  MOVE 'N' TO SW-PF7
ELSE
  MOVE 'F7-BACKWARD' TO TCBPFØ7O
  MOVE 'Y' TO SW-PF7
END-IF
MOVE SPACES TO TCBROW1O TCBROW2O TCBROW3O TCBROW4O TCBROW5O TCBROW6O TCBROW7O TCBROW8O TCBROW9O TCBROWAO
IF CNTR NOT > ASCB-NUM
  PERFORM FILL-ROWS
  MOVE TCBROWS TO TCBROW1O
  MOVE TCBCOLR TO TCBROW1C
  ADD 1 TO CNTR
  IF CNTR NOT > ASCB-NUM
    PERFORM FILL-ROWS
    MOVE TCBROWS TO TCBROW2O
    MOVE TCBCOLR TO TCBROW2C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW30
MOVE TCBCOLR TO TCBROW3C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW40
MOVE TCBCOLR TO TCBROW4C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW50
MOVE TCBCOLR TO TCBROW5C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW60
MOVE TCBCOLR TO TCBROW6C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW70
MOVE TCBCOLR TO TCBROW7C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW80
MOVE TCBCOLR TO TCBROW8C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROW90
MOVE TCBCOLR TO TCBROW9C
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWA0
MOVE TCBCOLR TO TCBROWAC
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWB0
MOVE TCBCOLR TO TCBROWBC
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWC0
MOVE TCBCOLR TO TCBROWCC
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWDO
MOVE TCBCOLR TO TCBROWDC
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWEO
MOVE TCBCOLR TO TCBROWEC
ADD 1 TO CNTR
IF CNTR NOT > ASCB-NUM
PERFORM FILL-ROWS
MOVE TCBROWS TO TCBROWFO
MOVE TCBCOLR TO TCBROWFC
ADD 1 TO CNTR
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF
END-IF

* MOVE 'TOTAL : ' TO TCBTIT20(Ø1:Ø8)
MOVE ASCB-NUM TO NUM-WORK
MOVE NUM-WORK TO TCBTIT20(Ø9:Ø4)
SUBTRACT 1 FROM CNTR
MOVE 'TO : ' TO TCBTIT20(3Ø:Ø5)
MOVE CNTR TO TCBTIT20(35:4)
ADD 1 TO CNTR
IF CNTR > ASCB-NUM
MOVE SPACES TO TCBPFØ8O
MOVE 'N' TO SW-PF8
ELSE
MOVE 'F8-FORWARD' TO TCBPFØ8O
MOVE 'Y' TO SW-PF8
END-IF
.
*
* FILL ROWS WITH DATA
* FILL-ROWS.
MOVE DFHGREEN TO TCBCOLR
MOVE SPACES TO TCBROWS
MOVE TCB-PROG(CNTR) TO TCBROWS(01:08)
MOVE TCB-ADDR(CNTR) TO TCBROWS(10:08)
MOVE TCB-CPUT(CNTR) TO CPU-WORK
MOVE CPU-WORK TO TCBROWS(19:10)
IF TCB-FLG1(CNTR) = '*'
   MOVE DFHRED TO TCBCOLR
ELSE
   IF TCB-FLG2(CNTR) = '*'
      MOVE DFHYELLO TO TCBCOLR
   END-IF
END-IF
MOVE '....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.....+.750
   TO TCBROWS(30:50)
IF SW-DISP = 'C'
   * ASCB-SUM IS IN MILLISECS, TCB-CPUT ALSO ... SO MULTIPLY BY 50
      MULTIPLY TCB-CPUT(CNTR) BY 50 GIVING TCB-CPUTIME
      DIVIDE TCB-CPUTIME BY ASCB-SUM GIVING TCB-PCTU
ELSE
   * ASCB-TIM IS IN 100THS, TCB-CPUT IN MSECS .. SO MULTIPLY BY 5
      MULTIPLY TCB-CPUT(CNTR) BY 5 GIVING TCB-CPUTIME
      DIVIDE TCB-CPUTIME BY ASCB-TIM GIVING TCB-PCTU
END-IF
MOVE '******************************************************************'
   TO TCBROWS(30:TCB-PCTU)
.
*
SORT-TABLE.
*
* A PRETTY SIMPLE BUBBLE SORT
*
MOVE 'N' TO SWSORT
PERFORM VARYING EXT-CNT FROM ASCB-NUM BY -1
   UNTIL EXT-CNT NOT > 1 OR
      SWSORT = 'Y'
MOVE 'Y' TO SWSORT
PERFORM VARYING INT-CNT FROM 1 BY 1
   UNTIL INT-CNT NOT < EXT-CNT
IF SORT-FLD = 'PROGRAM'
   IF TCB-PROG(INT-CNT) > TCB-PROG(INT-CNT + 1)
      MOVE TCBA-ROW(INT-CNT) TO TCBA-ROWS
      MOVE TCBA-ROW(INT-CNT + 1) TO TCBA-ROW(INT-CNT)
      MOVE TCBA-ROWS TO TCBA-ROW(INT-CNT + 1)
      MOVE 'N' TO SWSORT
   END-IF
END-IF
IF SORT-FLD = 'ADDRESS'
   IF TCB-ATCB(INT-CNT) > TCB-ATCB(INT-CNT + 1)
      MOVE TCBA-ROW(INT-CNT) TO TCBA-ROWS
      MOVE TCBA-ROW(INT-CNT + 1) TO TCBA-ROW(INT-CNT)
MOVE TCBA-ROWS TO TCBA-ROW(INT-CNT + 1)
MOVE 'N' TO SWSORT
END-IF
END-IF
IF SORT-FLD = 'CPUTIME'
  IF TCB-CPUT(INT-CNT) < TCB-CPUT(INT-CNT + 1)
    MOVE TCBA-ROW(INT-CNT) TO TCBA-ROWS
    MOVE TCBA-ROW(INT-CNT + 1) TO TCBA-ROW(INT-CNT)
    MOVE TCBA-ROWS TO TCBA-ROW(INT-CNT + 1)
    MOVE 'N' TO SWSORT
  END-IF
  END-IF
END-PERFORM
END-PERFORM.

* RET-TO-CICS.
*
EXEC CICS SEND MAP('IPPDTCB') MAPSET('IPPDTCB')
END-EXEC
MOVE LOW-VALUES TO IPPDTCBI
EXEC CICS RETURN TRANSID(EIBTRNID)
  COMMAREA(COMMAREA)
  LENGTH(LENGTH OF COMMAREA)
END-EXEC.

* STOP THE TRANSACTION
ENDIT.
*
EXEC CICS
  SEND CONTROL
  FREEKB
  ERASE
END-EXEC
PERFORM DISPERR
*

* END THE TRANSACTION
DISPERR.
EXEC CICS
  SEND TEXT
    FROM(TX-QUIT-TO-CICS)
    LENGTH(LENGTH OF TX-QUIT-TO-CICS)
    FREEKB
    ERASE
END-EXEC
EXEC CICS RETURN
END-EXEC.
CONCLUSION

By using the DTCB transaction, we were able to see the amount of CPU consumed by one TCB in our CICS region in relation to the other TCBs in that same region, and explain where the CPU usage came from.

We also noticed that, in our case, CICS is not truly a ‘single’ TCB transaction processing system!

---

Stan Adriaensen
Systems Engineer
Groupe Royale Belge / IPPA (Belgium)

---

High-values for CSP transactions

There is no reserved word for ‘high-values’ in CSP programming.

When the CSP main application is calling a CICS program, the linkage record defined in the CSP additional record list maps the linkage section of the CICS called program.

If your CICS COBOL called program returns ‘high-values’ in the status field, you may code the following to examine it.

Define CSP working storage record for fields WHEX and WHEXFF as follows:

<table>
<thead>
<tr>
<th>Name</th>
<th>Level</th>
<th>Occurs</th>
<th>Type</th>
<th>Length</th>
<th>Bytes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WHEX</td>
<td>10</td>
<td>00001</td>
<td>BIN</td>
<td>00004</td>
<td>2</td>
<td>Working Hex</td>
</tr>
<tr>
<td>*</td>
<td>15</td>
<td>00001</td>
<td>CHA</td>
<td>00001</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>WHEXFF</td>
<td>15</td>
<td>00001</td>
<td>CHA</td>
<td>00001</td>
<td>1</td>
<td>Working High-values</td>
</tr>
</tbody>
</table>

In the CSP Process:

```
MOVE -1 TO WHEX
```

Now, the redefined field WHEXFF contains high-values X'FF', which can be used for further processing in your application.

---

Koh See Kit
Project Manager
Bank of China (Singapore)

---

© Xephon 1999
Neon Systems has announced Affinities Server Version 2.7, which now supports CICS Transaction Server Version 1.3. Affinities Server helps deliver 24x7 parallel sysplex system performance and availability by removing affinity dependencies that can occur when moving CICS applications to a parallel sysplex environment. Version 2.7 is designed to allow users to take advantage of the parallel sysplex environment for CICS application processing using the latest release of CICS. It allows CICS/ESA applications to take advantage of the continuous availability, workload balancing, and lower cost of computing provided by parallel sysplex environments, without requiring application rewrites.

For further information contact:
Neon Systems, 14141 Southwest Freeway, Suite 6200, Sugar Land, TX 77478, USA.
Tel: (281) 491 4200.

CICS users can benefit from Version 3.0 of Sybase’s Enterprise Application Studio (EAStudio), an integrated set of application development and deployment products. Version 3.0 includes Enterprise Application Server 3.0 (EAServer), PowerJ 3.0, and PowerBuilder 7.0.

Specific EAStudio features in Version 3.0 include native PowerBuilder component support, automatic deployment of PowerBuilder and Java components to EAServer, and remote debugging of PowerBuilder and Java components.

There are EAServer Application Integrators for access to CICS and stored procedures, a PowerBuilder user interface, and Java2 support with PowerJ 3.0. There are also high-availability features for clustering and load balancing, plus support for the SSL security for all client types.

For further information contact:
Sybase, 6475 Christie Avenue, Emeryville, CA 94608-9967, USA.
Tel: (510) 922 3500.
Sybase (UK), Sybase Court, Crown Lane, Maidenhead, Berks, SL6 8QX, UK.
Tel: (01628) 597100.

IBM has announced Version 3.1 of its CICS Transaction Server for OS/390. This client/server software includes Java application and Java Virtual Machine support, object-oriented interface to CICS services for C++; support for OS/390 SSL; CORBA client support; CICS Web interface enhancements; CICS Transaction Gateway for OS/390; dynamic routing and load balancing of distributed program link and EXEC CICS START requests; CICSPlex System Manager enhancements; CICSPlex SM Web User interface; and resource definition online for CICS temporary storage.

For further information contact your local IBM representative.