In this issue

3 Temporary storage CPU costs
6 CICS/TS 1.3 and Wireless Mark-up Language
17 TMONCICS transaction record detailed analysis – part 2
34 Wiring CICS regions for desktop sound, e-mail, and alpha paging via TCP/IP – part 1
48 CICS news

© Xephon plc 2000
Subscriptions and back-issues
A year’s subscription to CICS Update, comprising twelve monthly issues, costs £175.00 in the UK; $270.00 in the USA and Canada; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £16.00 ($23.50) each including postage.

CICS Update on-line
Code from CICS Update can be downloaded from our Web site at http://www.xephon.com/cicsupdate.html; you will need the user-id shown on your address label.

Editor
Trevor Eddolls

Disclaimer
Readers are cautioned that, although the information in this journal is presented in good faith, neither Xephon nor the organizations or individuals that supplied information in this journal give any warranty or make any representations as to the accuracy of the material it contains. Neither Xephon nor the contributing organizations or individuals accept any liability of any kind howsoever arising out of the use of such material. Readers should satisfy themselves as to the correctness and relevance to their circumstances of all advice, information, code, JCL, and other contents of this journal before making any use of it.

Contributions
Articles published in CICS Update are paid for at the rate of £170 ($260) per 1000 words and £100 ($160) per 100 lines of code for the first 200 lines of original material. The remaining code is paid for at the rate of £50 ($80) per 100 lines. In addition, there is a flat fee of £30 ($50) per article. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our Notes for Contributors, or you can download a copy from www.xephon.com/contnote.html.

© Xephon plc 2000. 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. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs $36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

Printed in England.
Temporary storage CPU costs

BACKGROUND
In some CICS/ESA 4.1.0 environments, it is noted that CICS CPU usage and temporary storage request response times increase during the running of the CICS system. This is a result of several factors that are reviewed here, and recommendations are given to address the situation.

CICS/ESA 4.1.0 maintains a binary tree structure containing the temporary storage queue objects (TSUTEs) within a CICS system. Each TSUTE represents a unique temporary storage queue. The binary tree holds the TSUTEs as individual nodes within the tree. From the root node, there are 0, 1, or 2 sub-nodes. Each sub-node can in turn have 0, 1, or 2 sub-nodes of its own. The tree is structured so that its nodes (TSUTEs) are held in an alphanumerically increasing order across the tree. Binary trees are maintained such that a given node’s left and right sub-nodes (if they exist) are ‘less than’ and ‘greater than’ the value of the parent node, respectively. CICS will insert/locate/remove nodes within the tree as temporary storage queues are created/updated/deleted.

As an increasing number of temporary storage queues are created during a CICS session, so increasing numbers of TSUTE nodes are inserted into the binary tree structure. CICS systems programmers have always been aware of the need for prudent management of their temporary storage data on the system – each record added to queues results in additional control block relationships being required to represent the new data, and of course there is the need to hold the data items themselves (within the CICS address space for main temporary storage queues, or buffered there until written to VSAM ESDS DFHTEMP for auxiliary destinations). As such, it has always been recognized that redundant temporary storage queues should be deleted when no longer required. In addition to freeing up control block and record/buffer space, this also allows the TSUTE node to be removed from the tree structure. Typical CICS systems can have a great many
temporary storage queues that are still validly required however, and so the tree can grow to hold a large number of TSUTE nodes. As each node can have at most only two sub-nodes, there is the need for CICS to ensure the tree stays well-balanced. An unbalanced tree is one with an asymmetrical branch structure. If CICS determines the tree is unbalanced, it adjusts the nodes to make a different node the root node, and so have a more even number of sub-nodes from each nodal point. CICS performs such an operation (resplay) automatically when a temporary storage request references a TSUTE whose node is greater than 17 sub-nodes deep into the tree.

Such a resplay operation is CPU-intensive because it involves a series of pointer manipulations. It also has the effect of increasing the pathlength of the temporary storage request that triggered it, and hence affecting the task’s response time. Occurring under the QR CICS TCB, it can also momentarily affect the overall transaction throughput of the system. The benefit of a resplay is that the better-balanced tree resulting from it will have its nodes arranged in a way that means they are located quicker by future temporary storage requests.

It can be seen that as the number of temporary storage queues increases during a CICS run, so the tree size grows and the search depth increases. This results in additional CPU costs to process the requests. If such requests also force CICS into resplaying the tree, there is the additional CPU cost of this operation too. Hence CPU usage and response times can grow as the CICS session progresses.

CAUSES OF FREQUENT RESPLAYS

CICS environments that generate many temporary storage queues with identical queue name prefixes can result in unbalanced trees, because CICS will always insert the next instance of such a queue beneath the preceding one within the tree. One example is that of REQIDs specified for EXEC CICS START requests that differ by some fairly random suffix value but which all begin with the same prefix. These can result in resplaying activity being kicked off when the started task retrieves its FROM data from the REQID queue, and
its TSUTE has to be located within the tree. If this retrieval operation exceeds 17 sub-nodes, CICS resplays the tree. However, such a queue will in all likelihood never get referenced again since it will be deleted when the ATI’d task terminates, so the benefit of moving its node will not be seen.

RECOMMENDATIONS

A good rule of thumb is to try and randomize the temporary storage queue name prefixing as much as possible. This allows for a naturally balanced tree structure and so reduces the need for CICS to resplay the tree. If common prefixes are required, one alternative may be to use a GLUE to change the queue name around and have the ‘random’ suffix component of the name at the start of the queue name. For START requests, the REQID passed to XICEREQ as an argument via UEPCLPS could be modified in such a way. Also, always ensure that redundant queues are deleted as soon as possible. This is good for many reasons, and also reduces the size of the tree and hence the search depth for the remaining TSUTEs, so preventing resplays. Finally, ensure APAR PN89684 (PTF UN98081) is applied, because this modified CICS to increase the search depth that can trigger a resplay operation.

One of the many enhancements to CICS Transaction Server was to change the tree into a 16-way structure, providing a naturally shallower tree and so reducing search depths for nodes and avoiding the need for resplays. This helps avoid the temporary storage CPU cost increases over time that can be seen in CICS/ESA 4.1.0.

I hope this article has helped explain the background to CICS temporary storage CPU utilization within CICS/ESA 4.1.0.

_CICS is a registered trademark of International Business Machines Corporation. Readers wishing to discuss the material in this article further are welcome to contact me via e-mail, at address andy_wright@uk.ibm.com._

__Andy Wright__
_CICS Change Team Programmer__
__IBM (UK)__

© IBM 2000
CICS/TS 1.3 and Wireless Mark-up Language

INTRODUCTION
The public’s apparently insatiable demand for smaller mobile handheld devices with Web access (driven on by the marketing men and beloved in the Far East) has given rise to the Wireless Mark-up Language (WML).

This article demonstrates a general purpose CICS program that can be used to send WML data.

WML BASICS
WML is a lightweight mark-up language. It works along similar lines to HTML, except that the language is specifically optimized for specifying presentation and user interaction on limited-capability devices such as palm organizers and mobile phones.

In the same way that HTML supports scripting languages, WML supports a scripting language – WMLScript.

Images can be sent as bitmaps using the Wireless Bitmap Format (WBMP).

WML AND THE CICS DOCUMENT API
The great thing about the document API, available with CICS/TS 1.3, is that it is capable of manipulating many different types of data. Or to put it another way, it is not tied to HTML.

All that is required is a CICS application program that will serve up the appropriate content type when requested.

Example content type headers for WML applications include:

- text/vnd.wap.wml for WML.
- text/vnd.wap.wmlscript for WMLScripts.
Phone.com’s Handheld Device Mark-up Language (HDML) served as the basis for WML and is supported by older devices.

The only problem for CICS is deciding upon a naming convention for CICS templates, because CICS does not support file extensions.

In our example code, the naming convention implemented uses the first three characters of the object name to differentiate between the different flavours of WML data.

SAMPLE PROGRAM T13WAP

This program is invoked with the following URL, and is designed to work with the default analyser program DFHWBADX:

http://ipaddress:port/no_converter/alias/program/object_name

Where:

• ipaddress:port is the DNS or IP address of your MVS system (port relates to a configured TCPIPSERVICE definition).
• No_converter implies a keyword of cics or CICS.
• Alias implies a clone of the alias transaction (default CWBA).
• Program is T13WAP.
• Object_name is the name of the WML object you wish to send (must have a defined and installed doctemplate definition).

The first three characters of the object_name are examined and the appropriate content-type is set as follows:

• wml+++++ – WML
• wbmp+++++ – WBMP (bitmap)
• wls+++++ – WMLS (wlmscript)
• whd+++++ – HDML.
This program can also deliver other types of data – look at the comments in the code.

To display, you will need a WML-capable browser such as the phone emulator available at Phone.com’s Web page for developers – http://www.phone.com/developers/index.html.

![Figure 1: Example phone emulator](image)

**WML TEMPLATE EXAMPLE**

WML is a tag-based document language and is specified as an extensible mark-up language [XML] document type, eg:

```xml
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN" "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
  <card id="Circle" title="Demo">
    <p>WML demo from Circle Educations CICS/TS 1.3 system</p>
  </card>
</wml>
```
PROGRAM T13WAP

*******************************************
* Circle Computer Group
* MODULE NAME      T13WAP.cob
* picture processing for 1.3
* Check the first letter of the object name
* and set the appropriate content-type
* currently supports
* j jpeg, g gif, p pdf, f fdf, m mpg or s for js
*  
* WML processing for 1.3
* Check the first three characters of the object name
* and set the appropriate content-type
* currently supports
* wml wml, wbm wbmp, wlc wmlc, wls wmls, wsc wmlsc
* whd hdml  
*******************************************

PROCESS XOPTS(NOLINKAGE)
IDENTIFICATION DIVISION.
PROGRAM-ID. T13WAP.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
* Symbols table
Ø1 WS-SYMBOLS.
Ø3 filler     pic x(6) value 'docnm='. 
Ø3 WS-DOC-NM PIC X(8) VALUE SPACE.
Ø3 filler     pic x(7) value '&doctn='. 
Ø3 WS-DOC-TN PIC X(48) VALUE SPACE.
Ø3 filler     pic x(7) value '&doccr='. 
Ø3 WS-DOC-CR PIC X(3) VALUE SPACE.
Ø3 filler     pic x(7) value '&docdd='. 
Ø3 WS-DOC-DD PIC X(8) VALUE SPACE.
Ø3 filler     pic x(7) value '&docmn='. 
Ø3 WS-DOC-MN PIC X(8) VALUE SPACE.
Ø3 filler     pic x(7) value '&docds='. 
Ø3 WS-DOC-DS PIC X(56) VALUE SPACE.
Ø3 filler     pic x(7) value '&docy='. 
Ø3 WS-DOC-TY PIC X(6) VALUE SPACE.
Ø3 filler     pic x(5) value '&act='. 
Ø3 WS-DOC-ACT PIC X(6) VALUE SPACES.
Ø3 filler     PIC x(7) value '&hostv='. 
Ø3 ws-host    PIC x(24) value space.
Ø3 filler     PIC x(7) value '&resp='. 
Ø3 ws-resp-message PIC x(24) value space.
Ø3 FILLER     PIC X(9) VALUE '&message='.
Ø3 WS-BAD-MESSAGE PIC X(30) VALUE SPACES.
Ø1 WS-build.

2 ws-object-name PIC X(8) value spaces.
2 filler redefines ws-object-name.
 3 toChar pic x occurs 8 times
     indexed by toIndex.
2 ws-newstring pic x(100).
2 filler redefines ws-newstring.
 3 fromChar pic x occurs 100 times
     indexed by fromIndex, lastIndex, startIndex.
2 ws-object-name-len pic s9(4) comp value 0.

1 WS-HTTP-HEADER.
 5 pragma-hdr PIC X(6) VALUE IS 'Pragma'.
 5 pragma-val PIC X(7) VALUE IS 'nocache'.
 5 content-type-hdr PIC X(12) VALUE IS 'Content-type'.
 5 content-type-val PIC X(30) VALUE IS SPACES.
 5 content-type-len PIC 9(8) COMP VALUE 0.
 77 content-pdf PIC X(15) VALUE IS 'application/pdf'.
 77 content-fdf PIC X(19)
     VALUE IS 'application/vnd.fdf'.
 77 content-gif PIC X(9) VALUE IS 'image/gif'.
 77 content-jpg PIC X(10) VALUE IS 'image/jpeg'.
 77 content-mpg PIC X(10) VALUE IS 'video/mpeg'.
 77 content-text PIC X(9) VALUE IS 'text/html'.
 77 content-wbmp PIC X(18) VALUE IS 'image/vnd.wap.wbmp'.
 77 content-wml PIC X(16) VALUE IS 'text/vnd.wap.wml'.
 77 content-hdml PIC X(11) VALUE IS 'text/x-hdml'.
 77 content-wmlc PIC X(24)
     VALUE IS 'application/vnd.wap.wmlc'.
 77 content-wmls PIC X(22)
     VALUE IS 'text/vnd.wap.wmlscript'.
 77 content-wmlsc PIC X(30)
     VALUE IS 'application/vnd.wap.wmlscriptc'.
 77 content-js PIC X(24)
     VALUE IS 'application/x-javascript'.

1 HTTP-EXTRACT.
 3 HTTP-METHOD PIC X(6).
 3 HTTP-VERSION PIC X(10).
 3 HTTP-PATH PIC X(100).
 3 HTTP-QUERY-STRING PIC X(256).
 3 HTTP-TYPE PIC S9(8) COMP.
 3 HTTP-METHOD-LENGTH PIC S9(8) COMP VALUE +6.
 3 HTTP-VERSION-LENGTH PIC S9(8) COMP VALUE +10.
 3 HTTP-PATH-LENGTH PIC S9(8) COMP VALUE +100.
 3 HTTP-QUERY-STRING-LEN PIC S9(8) COMP VALUE +256.

1 CLIENT-INFO-EXTRACT.
 3 client-name PIC X(100).
 3 client-address PIC X(100).
 3 server-name PIC X(100).
 3 server-address PIC X(100).
 3 client-address-nu PIC 9(8) COMP.
REQUESTTYPE(HTTP-TYPE)
RESP(WS-RESP)
END-EXEC
* extract tcpip info
EXEC CICS EXTRACT TCPIP
CLIENTNAME(CLIENT-NAME)
CNAMELENGTH(CLIENT-NAME-LENGTH)
SERVERNAME(SERVER-NAME)
SNAMELENGTH(SERVER-NAME-LENGTH)
CLIENTADDR(CLIENT-ADDRESS)
CADDRLENGTH(CLIENT-ADDRESS-LENGTH)
CLIENTADDRNU(CLIENT-ADDRESS-NU)
SERVERADDR(SERVER-ADDRESS)
SADDRLENGTH(SERVER-ADDRESS-LENGTH)
SERVERADDRNU(SERVER-ADDRESS-NU)
SSLTYPE(SSL-TYPE)
TCPIPSERVICE(TCPIP-SERVICE)
PORTNUMBER(PORT-NUMBER)
PORTNUMNU(PORT-NUMBER-NU)
RESP(WS-RESP)
END-EXEC
* convert path info to uppercase
   perform with test after varying counter from 1 by +1
   until counter = 27
   inspect http-path replacing all
   lowerc(counter) by upperc(counter)
   end-perform
* need to determine what token was entered
   set toIndex to 1
   set lastIndex to http-path-length
   move http-path to ws-newstring
   perform append
   move ws-object-name to ws-template-name
   ws-doc-nm
* add the header template to master document
EXEC CICS
   DOCUMENT CREATE
   DOCTOKEN(WS-TOKEN)
   DOCSIZE(WS-RETRIEVE-LENGTH)
   TEMPLATE(WS-TEMPLATE-NAME)
   SYMBOLLIST(WS-SYMBOLS)
   LISTLENGTH(LENGTH OF WS-SYMBOLS)
   RESP(WS-RESP)
END-EXEC
* check the response
EVALUATE WS-RESP
WHEN DFHRESP(NORMAL)
   PERFORM GOODCODE
WHEN OTHER
PERFORM BADCODE
END-EVALUATE
EXEC CICS RETURN END-EXEC.
GOODCODE.
* Check the object name
* and set the appropriate content-type
* currently supports
* j jpeg, g gif, p pdf, f fdf, m mpg or s for js
* wml wml, wbm wbmp, wlc wmlc, wls wmls, wsc wmlsc
* whd hdml
if ws-template-name(1:3) = 'WML'
  move content-wml to content-type-val
  move length of content-wml to content-type-len
end-if
if ws-template-name(1:3) = 'WBM'
  move content-wbmp to content-type-val
  move length of content-wbmp to content-type-len
end-if
if ws-template-name(1:3) = 'WLC'
  move content-wmlc to content-type-val
  move length of content-wmlc to content-type-len
end-if
if ws-template-name(1:3) = 'WLS'
  move content-wmls to content-type-val
  move length of content-wmls to content-type-len
end-if
if ws-template-name(1:3) = 'WSC'
  move content-wmlsc to content-type-val
  move length of content-wmlsc to content-type-len
end-if
if ws-template-name(1:1) = 'J'
  move content-jpg to content-type-val
  move length of content-jpg to content-type-len
end-if
if ws-template-name(1:1) = 'G'
  move content-gif to content-type-val
  move length of content-gif to content-type-len
end-if
if ws-template-name(1:1) = 'M'  
  move content-mpg to content-type-val  
  move length of content-mpg to content-type-len
end-if
if ws-template-name(1:1) = 'P'  
  move content-pdf to content-type-val  
  move length of content-pdf to content-type-len
end-if
if ws-template-name(1:1) = 'F'  
  move content-fdf to content-type-val  
  move length of content-fdf to content-type-len
end-if
if ws-template-name(1:1) = 'S'  
  move content-js to content-type-val  
  move length of content-js to content-type-len
end-if
*
* write the content type header
EXEC CICS WEB WRITE
  HTTPHEADER (CONTENT-TYPE-HDR)
  NAMELENGTH(LENGTH OF CONTENT-TYPE-HDR)
  VALUE(CONTENT-TYPE-VAL)
  VALUELENGTH(CONTENT-TYPE-LEN)
END-EXEC
*
* send the master document
evaluate content-type-val
when content-js
  EXEC CICS
    WEB SEND
    CLNTCODEPAGE('819')
    DOCTOKEN(WS-TOKEN)
END-EXEC
when content-wml
  EXEC CICS
    WEB SEND
    CLNTCODEPAGE('819')
    DOCTOKEN(WS-TOKEN)
END-EXEC
when content-hdml
  EXEC CICS
    WEB SEND
    CLNTCODEPAGE('819')
    DOCTOKEN(WS-TOKEN)
END-EXEC
when content-wmls
EXEC CICS
  WEB SEND
  CLNTCODEPAGE('819')
  DOCTOKEN(WS-TOKEN)
END-EXEC
when other
EXEC CICS
  WEB SEND
  DOCTOKEN(WS-TOKEN)
END-EXEC
end-evaluate.
BADCODE.
* we must have got a notfnd from the previous
* create document - send an error html page
* set content-type header for text/html
  move content-txt to content-type-val
  move length of content-txt
    to content-type-len
  MOVE 'Host' to HDR-NAME
  MOVE 'CICS returned notfnd for ' to ws-bad-message
* set base document symbol
EXEC CICS WEB READ
  HTTPHEADER (HDR-NAME)
  NAMELENGTH(4)
  VALUE(HDR-VAL)
  VALUELENGTH(LENGTH OF HDR-VAL)
  RESP(WS-RESP)
END-EXEC
MOVE HDR-VAL TO WS-HOST
* add the error html page
EXEC CICS
  DOCUMENT CREATE
  DOCTOKEN(WS-TOKEN)
  DOCSIZE(WS-RETRIEVE-LENGTH)
  TEMPLATE('T13WAPB')
  SYMBOLLIST(WS-SYMBOLS)
  LISTLENGTH(LENGTH OF WS-SYMBOLS)
  RESP(WS-RESP)
END-EXEC
* set the content-type header
EXEC CICS WEB WRITE
  HTTPHEADER (CONTENT-TYPE-HDR)
  NAMELENGTH(LENGTH OF CONTENT-TYPE-HDR)
  VALUE(CONTENT-TYPE-VAL)
  VALUELENGTH(CONTENT-TYPE-LEN)
END-EXEC
* send the master document but translate to ascii
EXEC CICS
  WEB SEND
append.

* COBOL stuff to work out how many bytes present in object name

perform varying fromIndex from lastIndex by -1 until
   fromChar(fromIndex) equal "/" or
   fromIndex = 1
end-perform
set fromIndex up by 1
perform varying startIndex from fromIndex by 1 until
   startIndex > lastIndex or
   toIndex > 8
   move fromChar(startIndex) to toChar(toIndex)
set toIndex up by 1
end-perform.

AA999-EXIT.
EXIT.
STOP RUN.

T13WAPB HTML TEMPLATE (NOT FOR USE WITH WML DEVICES)

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<html>
<head>
<title> CIRCLE CICS/TS 1.3 Error PAGE</title>
</head>
<body>
<form action="/cics/cwba/docntem">
<h1>There was a problem with your request</h1>
<p>&message;&docnm;</p>
<p>Probable cause - doctemplate not defined</p>
<p>or not defined in template library</p>
<table>
<tr>
<td><input type="submit" value="update Doctemplate"></td>
</tr>
</table>
</form>
</body>
</html>

David Clancy
Circle Computer Group (UK) © Circle Computer Group 2000
TMONCICS transaction record detailed analysis – part 2

This month we conclude the code that extends the functionality of TMCE, to assist in debugging problems encountered on production CICS systems.

address ispexec "tbcreate tmextab nowrite replace,
    names(descr value)"

  do v = 1 to 197
    value = val.v
    descr = des.v
    address ispexec "tbadd tmextab"
  end

address ispexec "tbtop tmextab"

next = 'n'

  do until next = 'y'
    address ispexec "tbdispl tmextab panel(tmtap3)"
    if rc ¬= Ø then do
      address ispexec "tbclose tmextab"
      signal EXIT_EXTAB
    end

  select
    when ZCMD = 'HEX' then do
      call proc_hex_display
    end

    when ZCMD = 'SAVE' then do
      call proc_save_to_file
    end

    when ZCMD = 'FAT' then
      if val.179 ¬= Ø then do
        call proc_fat_segments
    end

    when ZCMD = 'UVL' then
      if val.182 ¬= Ø then do
        call proc_uvl_segments
  end

    when ZCMD = 'UTG' then
      if val.185 ¬= Ø then do
        call proc_utg_segments
  end

    when ZCMD = 'MRO' then
      if val.188 ¬= Ø then do
        call proc_mro_segments
  end

    when ZCMD = 'REQ' then
      if val.191 ¬= Ø then do

call proc_req_segments
end
when ZCMD = 'WAT' then
  if val.194 ≠ Ø then do
    call proc_wat_segments
  end
when ZCMD = 'TCB' then
  if val.197 ≠ Ø then do
    call proc_tcb_segments
  end
otherwise
  next = 'y'
end                             /* end select     */
end                                    /* end do until   */
address ispexec "tbclose tmextab"
EXIT_EXTAB:
return
/*-------------------------------------------------------------------*/
/* Process hex display                                             */
/*-------------------------------------------------------------------*/
proc_hex_display:
address ispexec "tbcreate tmhextab nowrite replace, "
  names(offh offd valhex valdisp)"
offs = 1
reclen = length(tmnrec)
do until offs > reclen
  offh    = d2x(offs - 1)
  offd    = offs - 1
  left    = reclen - offs
  if left < 16 then
    tmnlen = left + 1
  else
    tmnlen = 16
  valhex  = c2x(substr(tmnrec,offs,tmnlen))
  valdisp = substr(tmnrec,offs,tmnlen)
  address ispexec "tbadd tmhextab"
  offs    = offs + 16
end                                   /* end do until   */
address ispexec "tbtop tmhextab"
address ispexec "tbdispl tmhextab panel(tmtap4)"
address ispexec "tbclose tmhextab"
return
/*-------------------------------------------------------------------*/
/* Process save to file command                                    */
 /*-------------------------------------------------------------------*/
proc_save_to_file:
savmem = 'TA'  val.1Ø                        /* Task number      */
"ALLOC FI(TMSAVE) DA(TMTA.SAVE("savmem")) SHR REUSE"
tmsave = 'Off Field Form Decription Value'
push tmsave
"execio 1 diskw tmsave"
call write_blank
do v = 1 to 197
   tmsave = des.v ' ' val.v
   push tmsave
   "execio 1 diskw tmsave"
end
a= 177; depth = 14; segname = 'File access segments'
call init_fat_des
call_sec = 'PROC_FAT_VAL'
call proc_save_segments
a= 180; depth = 7; segname = 'User variables segments'
call init_uv1_des
call_sec = 'PROC_UVL_VAL'
call proc_save_segments
a= 183; depth = 6; segname = 'User target segments'
call init_utg_des
call_sec = 'PROC_UTG_VAL'
call proc_save_segments
a= 186; depth = 9; segname = 'MRO segments'
call init_mro_des
call_sec = 'PROC_MRO_VAL'
call proc_save_segments
a= 189; depth = 7; segname = 'Request segments'
call init_req_des
call_sec = 'PROC_REQ_VAL'
call proc_save_segments
a= 192; depth = 6; segname = 'Wait segments'
call init_wat_des
call_sec = 'PROC_WAT_VAL'
call proc_save_segments
a= 195; depth = 9; segname = 'TCB segments'
call init_tcb_des
call_sec = 'PROC_TCB_VAL'
call proc_save_segments
call write_blank
"execio Ø diskw tmsave (finis"
"FREE FI(TMSAVE)"
return
/***************************************************************************/
/* Process save to file command / segment data */
/***************************************************************************/
proc_save_segments:
b = a + 1
c = b + 1
seg_off = x2d(val.a)
seg_len = x2d(val.b)
seg_num = val.c
seg_off = seg_off + 1
if seg_num > Ø then do
call write_blank
tmsave = segname
push tmsave
"execio 1 diskw tmsave"
call write_blank
do v = 1 to seg_num
    call signal_value
do u = 1 to depth
        tmsave = seg_des.u ' ' seg_val.u
        push tmsave
        "execio 1 diskw tmsave"
    end
end
seg_off = seg_off + seg_len
end
end
return
/*-------------------------------------------------------------------*/
/* Protect the loop structure by CALLing a routine which then issues */
/* Signal Value to a variable.                                       */
/*-------------------------------------------------------------------*/
signal_value:
signal value call_sec
return
/*-------------------------------------------------------------------*/
/* Write a blank line                                                 */
/*-------------------------------------------------------------------*/
write_blank:
tmsave = '
push tmsave
"execio 1 diskw tmsave"
return
/*-------------------------------------------------------------------*/
/* Process fat segment data                                          */
/*-------------------------------------------------------------------*/
proc_fat_segments:
seg_off = x2d(val.177)
seg_len = x2d(val.178)
seg_num = val.179
address ispexec "tbcreate tmsegtab nowrite replace,
    names(value descr)"
call init_fat_des
seg_off = seg_off + 1
do v = 1 to seg_num
    call proc_fat_val
    do u = 1 to 14
        value = seg_val.u
descr = seg_des.u
        address ispexec "tbadd tmsegtab"
    end
end
seg_off = seg_off + seg_len
end
address ispexec "tbtop tmsegtab"
address ispexec "tbdispl tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return
/*-------------------------------------------------------------------*/
/* Process fat segment values                                         */
/*-------------------------------------------------------------------*/
proc_fat_val:
seg_val.1  = substr(tmnrec,seg_off,8)
seg_val.2  = c2x(substr(tmnrec,seg_off+8,1))
seg_val.3  = c2x(substr(tmnrec,seg_off+9,2))
seg_val.4  = substr(tmnrec,seg_off+11,6)
seg_val.5  = substr(tmnrec,seg_off+17,4)
seg_val.6  = c2d(substr(tmnrec,seg_off+21,8))/1000000
seg_val.7  = c2d(substr(tmnrec,seg_off+29,4))
seg_val.8  = c2x(substr(tmnrec,seg_off+33,1))
seg_val.9  = c2d(substr(tmnrec,seg_off+34,4))
seg_val.10 = c2d(substr(tmnrec,seg_off+38,4))
seg_val.11 = c2d(substr(tmnrec,seg_off+42,4))
seg_val.12 = c2d(substr(tmnrec,seg_off+46,4))
seg_val.13 = c2d(substr(tmnrec,seg_off+50,4))
seg_val.14 = ''
return
/*-------------------------------------------------------------------*/
/* Process uvl segment data                                           */
/*-------------------------------------------------------------------*/
proc_uvl_segments:
seg_off = x2d(val.180)
seg_len = x2d(val.181)
seg_num = val.182
address ispexec "tbcreate tmsegtab nowrite replace,
  names(value descr)"
call init_uvl_des
seg_off = seg_off + 1
do v = 1 to seg_num
call proc_uvl_val
do u = 1 to 7
  value = seg_val.u
  descr = seg_des.u
  address ispexec "tbadd tmsegtab"
end
seg_off = seg_off + seg_len
end
/* end do v = */
address ispexec "tbtop tmsegtab"
address ispexec "tbdisp1 tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return
/*-------------------------------------------------------------------*/
/* Process uvl segment values                                         */
/*-------------------------------------------------------------------*/
proc_uvl_val:
seg_val.1  = substr(tmnrec,seg_off,8)
seg_val.2  = c2x(substr(tmnrec,seg_off+8,1))
seg_val.3 = substr(tmnrec,seg_off+12,1)
seg_val.4 = substr(tmnrec,seg_off+13,1)
seg_val.5 = substr(tmnrec,seg_off+14,1)
seg_val.6 = substr(tmnrec,seg_off+15,240)
seg_val.7 = ''
return
/*-----------------------------------------------*/
/* Process utg segment data */
/*-----------------------------------------------*/
proc_utg_segments:
seg_off = x2d(val.183)
seg_len = x2d(val.184)
seg_num = val.185
address ispexec "tbcreate tmsegtab nowrite replace,
names(value descr)"
call init_utg_des
seg_off = seg_off + 1
do v = 1 to seg_num
   call proc_utg_val
   do u = 1 to 6
      value = seg_val.u
      descr = seg_des.u
      address ispexec "tbadd tmsegtab"
   end
   seg_off = seg_off + seg_len
end /* end do v = */
address ispexec "tbtop tmsegtab"
address ispexec "tbdispl tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return
/*-----------------------------------------------*/
/* Process utg segment values */
/*-----------------------------------------------*/
proc_utg_val:
seg_val.1 = c2x(substr(tmnrec,seg_off,1))
seg_val.2 = c2x(substr(tmnrec,seg_off+1,1))
seg_val.3 = c2x(substr(tmnrec,seg_off+2,8))
seg_val.4 = c2d(substr(tmnrec,seg_off+10,4))
seg_val.5 = c2x(substr(tmnrec,seg_off+14,1))
seg_val.6 = ''
return
/*-----------------------------------------------*/
/* Process mro segment data */
/*-----------------------------------------------*/
proc_mro_segments:
seg_off = x2d(val.186)
seg_len = x2d(val.187)
seg_num = val.188
address ispexec "tbcreate tmsegtab nowrite replace,
names(value descr)"
call init_mro_des
seg_off = seg_off + 1
do v = 1 to seg_num
    call proc_mro_val
    do u = 1 to 9
        value = seg_val.u
        descr = seg_des.u
        address ispexec "tbadd tmsegtab"
    end
    seg_off = seg_off + seg_len
end
address ispexec "tbtop tmsegtab"
address ispexec "tbdispl tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return

/*-------------------------------------------------------------------*/
/* Process mro segment values                                        */
/*-------------------------------------------------------------------*/
proc_mro_val:
seg_val.1 = substr(tmnrec,seg_off,4)
seg_val.2 = c2x(substr(tmnrec,seg_off+4,1))
seg_val.3 = c2x(substr(tmnrec,seg_off+5,1))
seg_val.4 = c2x(substr(tmnrec,seg_off+6,1))
seg_val.5 = substr(tmnrec,seg_off+7,8)
seg_val.6 = c2d(substr(tmnrec,seg_off+15,8))/1000000
seg_val.7 = c2d(substr(tmnrec,seg_off+23,4))
seg_val.8 = c2x(substr(tmnrec,seg_off+27,1))
seg_val.9 = ''
return

/*-------------------------------------------------------------------*/
/* Process req segment data                                          */
/*-------------------------------------------------------------------*/
proc_req_segments:
seg_off = x2d(val.189)
seg_len = x2d(val.190)
seg_num = val.191
address ispexec "tbcreate tmsegtab nowrite replace, 
    names(value descr)"
call init_req_des
seg_off = seg_off + 1
do v = 1 to seg_num
    call proc_req_val
    do u = 1 to 7
        value = seg_val.u
        descr = seg_des.u
        address ispexec "tbadd tmsegtab"
    end
    seg_off = seg_off + seg_len
end
address ispexec "tbtop tmsegtab"
address ispexec "tbdispl tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return
/*-------------------------------------------------------------------*/
/* Process req segment values                                    */
/*-------------------------------------------------------------------*/
proc_req_val:
seg_val.1  = c2x(substr(tmnrec,seg_off,2))
seg_val.2  = c2d(substr(tmnrec,seg_off+2,4))
seg_val.3  = c2d(substr(tmnrec,seg_off+6,8))/1000000
seg_val.4  = c2d(substr(tmnrec,seg_off+14,4))
seg_val.5  = c2d(substr(tmnrec,seg_off+18,8))/1000000
seg_val.6  = substr(tmnrec,seg_off+26,16)
seg_val.7  = ''
return
/*-------------------------------------------------------------------*/
/* Process wat segment data                                      */
/*-------------------------------------------------------------------*/
proc_wat_segments:
seg_off = x2d(val.192)
seg_len = x2d(val.193)
seg_num = val.194
address ispexec "tbcreate tmsegtab nowrite replace,
    names(value descr)"
call init_wat_des
seg_off = seg_off + 1
do v = 1 to seg_num
    call proc_wat_val
    do u = 1 to 6
        value = seg_val.u
        descr = seg_des.u
        address ispexec "tbadd tmsegtab"
    end
    seg_off = seg_off + seg_len
end
address ispexec "tbtop tmsegtab"
address ispexec "tbdispl tmsegtab panel(tmtap3)"
address ispexec "tbclose tmsegtab"
return
/*-------------------------------------------------------------------*/
/* Process wat segment values                                    */
/*-------------------------------------------------------------------*/
proc_wat_val:
seg_val.1  = substr(tmnrec,seg_off,8)
seg_val.2  = substr(tmnrec,seg_off+8,8)
seg_val.3  = c2d(substr(tmnrec,seg_off+16,4))
seg_val.4  = c2d(substr(tmnrec,seg_off+20,8))/1000000
seg_val.5  = c2x(substr(tmnrec,seg_off+28,1))
seg_val.6  = ''
return
/*-------------------------------------------------------------------*/
/* Process tcb segment data                                      */
/*-------------------------------------------------------------------*/
proc_tcb_segments:
  seg_off = x2d(val.195)
  seg_len = x2d(val.196)
  seg_num = val.197
  address ispexec "tbcreate tmsegtab nowrite replace, 
                names(value descr)"
  call init_tcb_des
  seg_off = seg_off + 1
  do v = 1 to seg_num
     call proc_wat_val
     do u = 1 to 9
        value = seg_val.u
        descr = seg_des.u
        address ispexec "tbadd tmsegtab"
     end
     seg_off = seg_off + seg_len
  end /* end do v = */
  address ispexec "tbtop tmsegtab"
  address ispexec "tbdispl tmsegtab panel(tmtap3)"
  address ispexec "tbclose tmsegtab"
  return
/*----------------------------------------*/
/* Process tcb segment values */
/*----------------------------------------*/
proc_tcb_val:
  seg_val.1 = substr(tmnrec,seg_off,2)
  seg_val.2 = c2d(substr(tmnrec,seg_off+2,8))/1000000
  seg_val.3 = c2d(substr(tmnrec,seg_off+10,4))
  seg_val.4 = c2d(substr(tmnrec,seg_off+14,8))/1000000
  seg_val.5 = c2d(substr(tmnrec,seg_off+22,4))
  seg_val.6 = c2d(substr(tmnrec,seg_off+26,8))/1000000
  seg_val.7 = c2d(substr(tmnrec,seg_off+34,4))
  seg_val.8 = c2d(substr(tmnrec,seg_off+38,8))/1000000
  seg_val.9 = ''
  return
/*-----------------------------*/
/* Initialize main portion field descriptions */
/*-----------------------------*/
init_des:
  des.1 = '002 TMHDTYPE CL3 Product id and record type '
  des.2 = '005 TMHDFLG1 XL1 Global control flag #1 '
  des.3 = '006 TMHDVREL XL2 Version and release number '
  des.4 = '008 TMHDCDAT CL8 Date record was produced: yyyyymmdd'
  des.5 = '010 TMHDTIME XL4 Time record produced (64ms.) '
  des.6 = '014 TMHDJOBG CL8 User id (CICS jobname etc) '
  des.7 = '024 TASYSID CL4 CICS sysid '
  des.8 = '028 TACICLVL CL4 CICS level of generating system '
  des.9 = '02C TAMONVER XL1 Monitor version of generator '
  des.10 = '02D TATASKID PL4 Task number '
  des.11 = '031 TAOPRID CL3 Operator id '
  des.12 = '034 TAPTRAN CL8 Primary tran id
'03C TAO TRAN CL8 Original tran id

'044 TAT RTER ID CL4 Terminal id

'04B TAT MRMT CL4 Termid in owning region

'04C TAR TMSYS CL4 Sysid of owning region

'050 TAN TLM CL8 VTAM terminal lu name

'05B TAV TMID CL8 Region VTAM applid

'060 TAPGM1 CL8 Transaction base program

'068 TASM FSID CL4 MVS SMF system identification

'06C TAMS ID CL8 MVS image id

'074 TAJOBI D CL8 Job identifier

'084 TAAS ID XL2 Job asid

'08C TAAS ID XL2 Job asid

'090 TAO U OW LN CL8 Uowid luname

'09A TAOUOW CTM XL6 Uowid clock value

'0A0 TAU OWNUM XL2 Uowid sequence number

'0A2 TAU OWID XL8 Local uow

'0AA TAABNCDE CL4 Abend code

'0AE TAFACTY T XL1 Facility type from tca (tcafci)

'0CE TAEN GMTXL8 GMT offset in microseconds

'0F4 TATSKCT XL4 Task originated count

'0F8 TAUSERID CL8 Userid

'100 TAHWMRSN XL2 EIP function exceeding limit

'102 TASTRDTP PL4 Transaction start day

'106 TASTRTOD XL8 Transaction start time

'10E TAENDTP PL4 Transaction end day

'112 TAENDTOD XL8 Transaction end time

'11A TACPRUTM XL8 Task real cpu time

'122 TADBCTHC XL8 DBCTL thread cpu time

'12A TARSPTM XL8 Response time

'132 TARP S CT XL4 Response count

'136 TAOPRW TM XL8 Primary facility wait time

'13E TATRAN CT XL4 Tran count

'142 TATSKWTM XL8 Task wait time

'152 TATSKWTM XL8 Task dispatched time (process)

'15A TADISPCT XL4 Dispatch count

'15E TAMVSCT XL4 Wait MVS count

'162 TAMVSTM XL8 Wait MVS time

'16A TAO LDCCT XL4 Wait oldc count

'16E TAO LDCCTM XL8 Wait oldc time

'176 TAOLDCCT XL4 Wait olddw count

'17A TAOLDCWM XL8 Wait oldw time

'182 TASUSP CT XL4 Suspend count

'186 TASSUSP CT XL8 Suspend time

'18E TAINI TCT XL4 Init wait count

'192 TAINITTM XL8 Init wait time (wait 1st dispatch)
des.64  = '19A TASTCBCT XL4  TCB switch count
des.65  = '19E TASTCBTM XL8  TCB switch time
des.66  = '1A6 TAEXCWCT XL4  Exception wait count
des.67  = '1AA TAEXCWTM XL8  Exception wait time
des.68  = '1B2 TAIQCT XL4  Total i/o count
des.69  = '1B6 TAIOTM XL8  Total i/o time
des.70  = '1BE TAISPRCT XL4  MRO/ISC count
des.71  = '1C2 TAISPRTM XL8  MRO/ISC time
des.72  = '1CA TAPGEICT XL4  Page in count
des.73  = '1CE TAPGEOCt XL4  Page out count
des.74  = '1D2 TAPGETCT XL4  Total page activity
des.75  = '1D6 TATIOHWM XL4  TIOA storage hwm
des.76  = '1DA TAU24HWM XL4  User 24 pool hwm
des.77  = '1DE TAU31HWM XL4  User 31 pool hwm
des.78  = '1E2 TAC24HWM XL4  CICS 24 pool hwm
des.79  = '1E6 TAC31HWM XL4  CICS 31 pool hwm
des.80  = '1EA TAGETMCT XL4  Getmain count
des.81  = '1EE TASTGWTM XL8  Storage suspend wait time
des.82  = '1F6 TAGMAWCT XL4  Storage suspend count
des.83  = '1FA TATERWTM XL8  Terminal suspend wait time
des.84  = '202 TATERWCT XL4  Terminal suspend wait count
des.85  = '206 TATERICT XL4  Terminal input count
des.86  = '20A TATERILG XL4  Terminal input length
des.87  = '20E TATEROCT XL4  Terminal output count
des.88  = '212 TATEROLG XL4  Terminal output length
des.89  = '216 TATERDTP XL1  Terminal device type from tctte
des.90  = '217 TAAIDSTU XL1  Tctte AID byte at task start
des.91  = '218 TAFLAG1 XL1  Flag byte 1
des.92  = '219 TAFLAG2 XL1  Flag byte 2
des.93  = '21A TAFLAG3 XL1  Flag byte 3
des.94  = '21B TAFLAG4 XL1  Flag byte 4
des.95  = '21C TAFLAG5 XL1  Flag byte 5
des.96  = '21D TAFLAG6 XL1  Flag byte 6
des.97  = '21E TAFLAG7 XL1  Flag byte 7
des.98  = '21F TAFLAG8 XL1  Flag byte 8
des.99  = '220 TAFLAG9 XL1  Flag byte 9
  . . .
<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
<th>Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>des.115</td>
<td>'244 TAFCPRTM XL8</td>
<td>FCP file request time</td>
</tr>
<tr>
<td>des.116</td>
<td>'24C TAFCPRCT XL4</td>
<td>FCP request count</td>
</tr>
<tr>
<td>des.117</td>
<td>'250 TAFCPWTM XL8</td>
<td>FCP file wait time</td>
</tr>
<tr>
<td>des.118</td>
<td>'258 TAFCPWCT XL4</td>
<td>FCP wait count</td>
</tr>
<tr>
<td>des.119</td>
<td>'25C TADLIRTM XL8</td>
<td>DL1 request time</td>
</tr>
<tr>
<td>des.120</td>
<td>'264 TADLIRCT XL4</td>
<td>DL1 request count</td>
</tr>
<tr>
<td>des.121</td>
<td>'268 TADLIWTM XL4</td>
<td>DL1 wait time</td>
</tr>
<tr>
<td>des.122</td>
<td>'270 TADLIWCT XL4</td>
<td>DL1 wait count</td>
</tr>
<tr>
<td>des.123</td>
<td>'274 TAPGMRTM XL8</td>
<td>Program fetch request time</td>
</tr>
<tr>
<td>des.124</td>
<td>'27C TAPGMRCT XL4</td>
<td>Program fetch request count</td>
</tr>
<tr>
<td>des.125</td>
<td>'280 TAPGMWTM XL8</td>
<td>Program fetch wait time</td>
</tr>
<tr>
<td>des.126</td>
<td>'288 TAPGMWCT XL4</td>
<td>Program fetch wait count</td>
</tr>
<tr>
<td>des.127</td>
<td>'28C TAJCPRTM XL8</td>
<td>Journal request time</td>
</tr>
<tr>
<td>des.128</td>
<td>'294 TAJCPRCT XL4</td>
<td>Journal request count</td>
</tr>
<tr>
<td>des.129</td>
<td>'298 TAJCPWTM XL8</td>
<td>Journal wait time</td>
</tr>
<tr>
<td>des.130</td>
<td>'2A0 TAJCPWCT XL4</td>
<td>Journal wait count</td>
</tr>
<tr>
<td>des.131</td>
<td>'2A4 TATSQRTM XL8</td>
<td>TS request time</td>
</tr>
<tr>
<td>des.132</td>
<td>'2AC TATSQRCT XL4</td>
<td>TS request count</td>
</tr>
<tr>
<td>des.133</td>
<td>'2B0 TATSQWTM XL8</td>
<td>TS wait time</td>
</tr>
<tr>
<td>des.134</td>
<td>'2B8 TATSQWCT XL4</td>
<td>TS wait count</td>
</tr>
<tr>
<td>des.135</td>
<td>'2BC TATDQRTM XL8</td>
<td>TD req time</td>
</tr>
<tr>
<td>des.136</td>
<td>'2C4 TATDQRCT XL4</td>
<td>TD req count</td>
</tr>
<tr>
<td>des.137</td>
<td>'2C8 TATDQWMT XL8</td>
<td>TD wait time</td>
</tr>
<tr>
<td>des.138</td>
<td>'2D0 TATDQWCT XL4</td>
<td>TD wait count</td>
</tr>
<tr>
<td>des.139</td>
<td>'2D4 TAGDSRTM XL8</td>
<td>GDS req time</td>
</tr>
<tr>
<td>des.140</td>
<td>'2DC TAGDSRCT XL4</td>
<td>GDS req count</td>
</tr>
<tr>
<td>des.141</td>
<td>'2E0 TAGDSWMT XL8</td>
<td>GDS wait time</td>
</tr>
<tr>
<td>des.142</td>
<td>'2E8 TAGDSWCT XL4</td>
<td>GDS wait count</td>
</tr>
<tr>
<td>des.143</td>
<td>'2EC TASPLRRTM XL8</td>
<td>Spool req time</td>
</tr>
<tr>
<td>des.144</td>
<td>'2F4 TASPLRCT XL4</td>
<td>Spool req count</td>
</tr>
<tr>
<td>des.145</td>
<td>'2F8 TASPLWRTM XL8</td>
<td>Spool wait time</td>
</tr>
<tr>
<td>des.146</td>
<td>'300 TASPLWCT XL4</td>
<td>Spool wait count</td>
</tr>
<tr>
<td>des.147</td>
<td>'304 TARDORTM XL8</td>
<td>RDO req time</td>
</tr>
<tr>
<td>des.148</td>
<td>'30C TARDORCT XL4</td>
<td>RDO req count</td>
</tr>
<tr>
<td>des.149</td>
<td>'310 TARDOWTM XL8</td>
<td>RDO wait time</td>
</tr>
<tr>
<td>des.150</td>
<td>'318 TARDOWCT XL4</td>
<td>RDO wait count</td>
</tr>
<tr>
<td>des.151</td>
<td>'31C TAFEPRTM XL8</td>
<td>FEPI req time</td>
</tr>
<tr>
<td>des.152</td>
<td>'324 TAFEPRCT XL4</td>
<td>FEPI req count</td>
</tr>
<tr>
<td>des.153</td>
<td>'328 TAFEPWMT XL8</td>
<td>FEPI wait time</td>
</tr>
<tr>
<td>des.154</td>
<td>'330 TAFEPWCT XL4</td>
<td>FEPI wait count</td>
</tr>
<tr>
<td>des.155</td>
<td>'334 TASQLCTM XL8</td>
<td>SQL calls time</td>
</tr>
<tr>
<td>des.156</td>
<td>'33C TASQLCCT XL4</td>
<td>SQL calls count</td>
</tr>
<tr>
<td>des.157</td>
<td>'340 TANSQCTM XL8</td>
<td>DB2 non-SQL calls time</td>
</tr>
<tr>
<td>des.158</td>
<td>'348 TANSQCCT XL4</td>
<td>DB2 non-SQL count</td>
</tr>
<tr>
<td>des.159</td>
<td>'34C TADB2WTM XL8</td>
<td>DB2 wait time</td>
</tr>
<tr>
<td>des.160</td>
<td>'354 TADB2WCT XL4</td>
<td>DB2 wait count</td>
</tr>
<tr>
<td>des.161</td>
<td>'358 TAUDBRTM XL8</td>
<td>User database request time</td>
</tr>
<tr>
<td>des.162</td>
<td>'360 TAUDBRCT XL4</td>
<td>User database request count</td>
</tr>
<tr>
<td>des.163</td>
<td>'364 TAWMRESP XL8</td>
<td>WLM expected response</td>
</tr>
<tr>
<td>des.164</td>
<td>'36C TAWMPCNT XL4</td>
<td>WLM percent goal</td>
</tr>
<tr>
<td>des.165</td>
<td>'370 TAWMTGTP XL1</td>
<td>WLM target type</td>
</tr>
</tbody>
</table>
des.166 = '371 TAWMIMPT XL1 WLM importance  

des.167 = '372 TADB2ID CL4 DB2 subsys id  

des.168 = '376 TAUSER CL16 User data field  

des.169 = '386 TATCBSCT XL4 Total TCB switches  

des.170 = '38A TATCBSTM XL8 TCB switch time  

des.171 = '392 TAMQSRCT XL4 MQSERIES request count  

des.172 = '396 TAMQSRTM XL8 MQSERIES request time  

des.173 = '39E TAMQSWCT XL4 MQSERIES wait count  

des.174 = '3A2 TAMQSWTM XL8 MQSERIES wait time  

des.175 = '3AA TAINETOD XL8 Normalized interval end time  

des.176 = '3B2 TADBCCJOB CL8 DBCTL region name  

des.177 = '464 TAFILOFF XL2 Offset to fat segments FAT'  

des.178 = '466 TAFIILLEN XL2 Length of fat segments  

des.179 = '468 TAFIILCNT XL2 Number of fat segments  

des.180 = '46A TAUSRUFF XL2 Offset to usr-vl segments UVL'  

des.181 = '46C TAUSRLEN XL2 Length of usr-vl segments  

des.182 = '46E TAUSRRCNT XL2 Number of usr-vl segments  

des.183 = '470 TAUTGOF XL2 Offset to user target segments UTG'  

des.184 = '472 TAUTGLEN XL2 Length of user target segments  

des.185 = '474 TAUTGRCNT XL2 Number of user target segments  

des.186 = '476 TAMRUFF XL2 Offset to MRO segments MRO'  

des.187 = '478 TAMROLEN XL2 Length of MRO segments  

des.188 = '47A TAMROCNT XL2 Number of MRO segments  

des.189 = '47C TAARQOFF XL2 Offset to req segments REQ'  

des.190 = '47E TAAROLEN XL2 Length of req segments  

des.191 = '480 TAARQRCNT XL2 Number of req segments  

des.192 = '482 TAARWTOFF XL2 Offset to wait segments WAT'  

des.193 = '484 TAARWTLLEN XL2 Length of wait segments  

des.194 = '486 TAARWTCNT XL2 Number of wait segments  

des.195 = '488 TADSPOFF XL2 Offset to TCB usage segments TCB'  

des.196 = '48A TADSPLEN XL2 Length of TCB usage segments  

des.197 = '48C TADSPRCNT XL2 Number of TCB usage segments  

return  

/*-------------------------------------------------------------------*/  
/* Init fat segment descriptions */  
 electrocity-------------------------------------------------------------------*/  

/* Init fat segment descriptions */  

init_fat_des:  

seg.des.1 = '000 TAFILID CL8 File/DBD name  

seg.des.2 = '000 TAFILACM XL1 Access method type indicator  

seg.des.3 = '000 TAFILUCB XL2 File/DBD UCB chan/unit address  

seg.des.4 = '000 TAFILVOL CL6 Volume serial number  

seg.des.5 = '011 TAFILTRN CL4 Tran for DB2 rct entry  

seg.des.6 = '015 TAFILETI XL8 File/DBD gross access time  

seg.des.7 = '01D TAFILCT XL4 File/DBD times accessed  

seg.des.8 = '021 TAFILXCL XL1 Exception conditions  

seg.des.9 = '022 TAFILDEL XL4 Delete requests  

seg.des.10 = '026 TAFILGET XL4 File/DBD get requests  

seg.des.11 = '02A TAFILBRW XL4 File/DBD browse requests  

seg.des.12 = '02E TAFILADD XL4 File/DBD add requests  

seg.des.13 = '032 TAFILUPD XL4 File update requests  

seg.des.14 = ''
return
/*-------------------------------------------------------------------*/
/* Init uvl segment descriptions */
/*-------------------------------------------------------------------*/
init_uvl_des:
seg_des.1 = 'ØØØ TAUSRKEY CL8 User data key
seg_des.2 = 'ØØ8 TAUSRSEQ XL4 User data sequence
seg_des.3 = 'ØØC TAUSRFRM CL1 User data format
seg_des.4 = 'ØØD TAUSRFUL CL1 User data full flag
seg_des.5 = 'ØØE TAUSRSLK CL1 User data slack
seg_des.6 = 'ØØF TAUSRDAT XL24ØUser data area
seg_des.7 = ''
return
/*-------------------------------------------------------------------*/
/* Init utg segment descriptions */
/*-------------------------------------------------------------------*/
init_utg_des:
seg_des.1 = 'ØØØ TAUTID XL1 User target id
seg_des.2 = 'ØØ1 TAUTIND XL1 Type indicators
seg_des.3 = 'ØØ2 TAUTDATA XL8 Accumulated data
seg_des.4 = 'ØØA TAUTDCNT XL4 Target event count
seg_des.5 = 'ØØE TAUTSTIM XL8 Pending start time
seg_des.6 = ''
return
/*-------------------------------------------------------------------*/
/* Init mro segment descriptions */
/*-------------------------------------------------------------------*/
init_mro_des:
seg_des.1 = 'ØØØ TATARSID CL4 Sysid of target system
seg_des.2 = 'ØØ4 TATARFG1 XL1 MRO flag byte
seg_des.3 = 'ØØ5 TAMRGRP XL1 Request group id
seg_des.4 = 'ØØ6 TAMRFUN XL1 MRO function
seg_des.5 = 'ØØ7 TATARAID CL8 Applid of target system
seg_des.6 = 'ØØF TATARETM XL8 Total elapsed time
seg_des.7 = 'Ø17 TATARECT XL4 MRO event count
seg_des.8 = 'Ø1B TAMRFLG2 XL1 MRO flag byte 2
seg_des.9 = ''
return
/*-------------------------------------------------------------------*/
/* Init req segment descriptions */
/*-------------------------------------------------------------------*/
init_req_des:
seg_des.1 = 'ØØØ TAARQIDS XL2 EIB function code
seg_des.2 = 'ØØ2 TAARQCTS XL4 Request count
seg_des.3 = 'ØØ6 TAARQRTM XL8 Request time
seg_des.4 = 'ØØE TAARQWCT XL4 Imbedded wait count
seg_des.5 = 'Ø12 TAARQWTM XL8 Imbedded wait time
seg_des.6 = 'Ø1A TAARQTXT CL16 Request description
seg_des.7 = ''
return
/*-------------------------------------------------------------------*/
/* Init wat segment descriptions */
init_wat_des:
seg_des.1 = 'ØØØ TAAWTRTP CL8  Resource type
seg_des.2 = 'ØØ8 TAAWTRNM CL8  Resource name
seg_des.3 = 'Ø1Ø TAAWTCTS XL4  Wait count
seg_des.4 = 'Ø14 TAAWTMS XL8  Wait time
seg_des.5 = 'Ø1C TAAWTFLG XL1  Wait flag
seg_des.6 = ''
return
/* Init tcb segment descriptions */
init_tcb_des:
seg_des.1 = 'ØØØ TADSPID CL2  Dispatch TCB id
seg_des.2 = 'ØØ2 TADSPCPU XL8  TCB cpu time used
seg_des.3 = 'ØØA TADSPDSC XL4  TCB dispatch count
seg_des.4 = 'ØØE TADSPDST XL8  TCB elapsed time
seg_des.5 = 'Ø16 TADSPSC XL4  TCB switch count
seg_des.6 = 'Ø1A TADSPSST XL8  TCB switch time
seg_des.7 = 'Ø22 TADSPQSC XL4  Queued switch count
seg_des.8 = 'Ø26 TADSPQST XL8  Queued switch time
seg_des.9 = ''
return

TMTAP1 ISPF PANEL

)attr
~ type(text) color(yellow) skip(on)
! type(output) intens(low) just(right)
$ type(output) intens(low) just(asis)
# type(output) intens(high)
  type(text) skip(on) intens(low)
)body expand(@@)
@@-@  TMCE TA record display  @@
%Command  ===> _zcmd + %Scroll
  ===> _amt +
%
%Dataset  ===> _dsn +
%Trans    ===> _Z +
%Terminal ===> _Z +
+
)init
.help = tmtaph
.zvars = '(tranmask termmask)'
&zcmd = &z
&ztdmark = ''
)proc
vput (dsn)
)end

TMTAP2 ISPF PANEL

)attr
~ type(text) color(yellow) skip(on)
! type(output) intens(low) just(right)
$ type(output) intens(low) just(asis)
# type(output) intens(high)
    type(text) skip(on) intens(low)
)body expand(@@)
%@-@ TMCE TA record display @@
%Command ===>_zcmd                                     +     %Scroll
===>_amt +
%
%                      _Z +       _Z +
%X Date    Time   Jobname  Tran    Task Term    Resp    CPU  Paging
FileIO Abnd
%
)model
_Z$date $time    $jobn    $tran !tskn $term!resp !cpu !page !ioct
!abnd+
)init
.help = tmtaph
.zvars = '(tranmask termmask exp)'
&zcmd = &z
&ztmdmark = ' ')
)proc
)end

TMTAP3 ISPF PANEL

)attr
~ type(text) color(yellow) skip(on)
! type(output) intens(low) just(right)
$ type(output) intens(low) just(asis)
# type(output) intens(high)
    type(text) skip(on) intens(low)
)body expand(@@)
%@-@ TMCE TA record display - DSECT @@
%Command ===>_zcmd                                     +     %Scroll
===>_amt +
%
%Off  Field   Form Description                          Value
+
)model
$descr                                               +$value
+
)init
.help = tmtaph
&zcmd = &z
&ztmdmark = ' ' )
TMTAP4 ISPF PANEL

)attr
~ type(text) color(yellow) skip(on)
! type(output) intens(low) just(right)
$ type(output) intens(low) just(asis)
# type(output) intens(high)
  type(text) skip(on) intens(low)
)body expand(@@)
%@-@ TMCE TA record display - HEX @@
%Command ===> _zcmd + %Scroll
 ===> _amt +
%
% Offhex Offdec +
)model
  !offh +!offd + $valhex + $valdisp +
)init
 .help = tmtaph
 &zcmd = &z
 &ztmark = ' ',
)proc
)end

TMTAPH ISPF PANEL

)attr
# type(output) intens(high)
  type(text) skip(on) intens(low)
)body expand(@@)
%@-@ TMCE TA record display - Help @@
%Command ===> _zcmd + %Scroll
 ===> _amt +
 +
+ This system is used with The Monitor for CICS/ESA V2.0+(TMCE) from
+ Landmark Systems Corporation. Transaction Detail Records created by
+ TMCE, archived and decompressed, can be selected for display in a
+ detailed format, and this display can be saved to a disk file.
+ Panel TMTAP1 - Enter decompressed record dataset name, optionally
+ limit the subsequent display to specific transactions and/or
+ terminals.
+ Panel TMTAP2 - Select a specific record for detailed display with
+ 'x'. Press enter to get the next set of matching records. Overtype
In recent years, the mainframe has become a significant player in the local area networks and wide area networks of the modern business environment. With the implementation of the TCP/IP SOCKETS interface in CICS, the mainframe can now talk to the desktops within the company or a server on the other side of the world. The following paragraphs describe hypothetical examples of how a mainframe CICS region could play specific sounds on a desktop computer sound card, send e-mails to an SMTP server on the intranet or the Internet, or send an alpha page to a paging application hooked to an SMTP server.

In this example there are five programs, TCPIP00A through TCPIP00E. Each is discussed in its own section, and its workings should be relatively clear with minimal study.

TCPIP00A
This is the root program and is driven by a transaction that is started based upon an ICE timer event. It could be driven directly by
paragraphs in application programs that identify certain error conditions and tie these to some identification number. In this case, the procedure division paragraph MAINLINE retrieves sound, page, and e-mail data in the form of an identifying number and short piece of textual data. Subsequently, each function (sound, e-mail, and page) is processed in sequence followed by a simple posting of an ICE to repeat the process in a few seconds. The functions are processed in separate dedicated programs. This is accomplished via LINKS to each program. The links carry a communications area consisting of the numeric key, the short alpha message, and the IP address required for TCP/IP connection identification. Note that the COMM-IP-ADDRESS field is 9(8) comp. This is an unsigned field. This is critical because some higher IP addresses require the leftmost bit to represent them. To develop the address, convert each part of the COMM-IP-ADDRESS to a hex number of two digits, then combine the four as a single hex value. Finally, convert this to decimal and add it to the COMM-IP-ADDRESS field. (For example 10.64.01.127 on your LAN would be 0A.40.01.7F, or 0A40017F which is 171966847 in decimal. When added to the field, while the field is zero, a full word of 0A40017F results.)

ID DIVISION.
PROGRAM-ID. TCPIPØØA.
DATE-WRITTEN. DEC 1999.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 ABSTIME                  PIC S9(15) COMP-3.
77 RESTART-INTERVAL        PIC S9(8) COMP VALUE +Ø3Ø.
77 LINK-RESPONSE           PIC S9(8) COMP.
77 START-RESPONSE          PIC S9(8) COMP.
77 TRANSDATA-RESPONSE      PIC S9(8) COMP.
77 OPERATOR-RESPONSE       PIC S9(8) COMP.
77 ABEND-CODE              PIC X(Ø4).
77 SYSTEM-ID  PIC X(Ø4).
Ø1 PAGE-COMMAREA.
  Ø2 COMM-PAGE-NUM       PIC 9(6).
  Ø2 COMM-MESSAGE        PIC X(3Ø).
  Ø2 COMM-IP-ADDRESS     PIC 9(Ø8) COMP.
Ø1 ERROR-MESSAGE.
  Ø2 ERR-PROGRAM-ID            PIC X(Ø8).
  Ø2 FILLER                    PIC X(Ø1) VALUE SPACE.
  Ø2 ERR-DATE                  PIC X(Ø8).
  Ø2 FILLER                    PIC X(Ø1) VALUE SPACE.
PROCEDURE DIVISION.
MAINLINE.
PERFORM HANDLE-ABEND
PERFORM HOUSE-KEEPING
PERFORM GET-PAGE-SOUND
PERFORM SEND-PAGE-SOUND
PERFORM SEND-PAGE-PAGER
PERFORM SEND-PAGE-EMAIL
PERFORM START-TRANSACTION
GO TO CLEAR-EM-OUT.

HANDLE-ABEND.
EXEC CICS HANDLE ABEND LABEL(ABEND-ERR) END-EXEC
CONTINUE.

HOUSE-KEEPING.
EXEC CICS ASKTIME
    ABSTIME(ABSTIME)
END-EXEC
EXEC CICS FORMATTIME
    ABSTIME(ABSTIME)
    YYMMDD(ERR-DATE) DATESEP
    TIME(ERR-TIME) TIMESEP
END-EXEC
EXEC CICS ASSIGN SYSID(SYSTEM-ID) END-EXEC
MOVE SYSTEM-ID TO OPR-REGION-ID
MOVE 'TCPIP\00A' TO ERR-PROGRAM-ID
OPR-PROGRAM-ID
MOVE EIBTRNID TO ERR-TRANSACTION
CONTINUE.
GET-PAGE-SOUND.
  MOVE 'LINK    ' TO ERR-CALL-TYPE
  EXEC CICS LINK PROGRAM('TCPIPØØB')
    COMMAREA(PAGE-COMMAREA)
    LENGTH(LENGTH OF PAGE-COMMAREA)
    RESP(LINK-RESPONSE) END-EXEC
  IF LINK-RESPONSE = DFHRESP(NORMAL)
    CONTINUE
  ELSE
    PERFORM EVALUATE-LINK-RESPONSE
    GO TO CLEAR-EM-OUT
  END-IF
  CONTINUE.
SEND-PAGE-SOUND.
  MOVE ZEROES TO COMM-IP-ADDRESS
  ADD 123456789 TO COMM-IP-ADDRESS
  IF COMM-PAGE-NUM NOT = 999000
    MOVE 'LINK    ' TO ERR-CALL-TYPE
    EXEC CICS LINK PROGRAM('TCPIPØØC')
      COMMAREA(PAGE-COMMAREA)
      LENGTH(LENGTH OF PAGE-COMMAREA)
      RESP(LINK-RESPONSE) END-EXEC
    IF LINK-RESPONSE = DFHRESP(NORMAL)
      CONTINUE
    ELSE
      PERFORM EVALUATE-LINK-RESPONSE
      GO TO CLEAR-EM-OUT
    END-IF
  END-IF
  CONTINUE.
SEND-PAGE-PAGER.
  MOVE ZEROES TO COMM-IP-ADDRESS
  ADD 123456789 TO COMM-IP-ADDRESS
  IF COMM-PAGE-NUM NOT = 999000
    MOVE 'LINK    ' TO ERR-CALL-TYPE
    EXEC CICS LINK PROGRAM('TCPIPØØD')
      COMMAREA(PAGE-COMMAREA)
      LENGTH(LENGTH OF PAGE-COMMAREA)
      RESP(LINK-RESPONSE) END-EXEC
    IF LINK-RESPONSE = DFHRESP(NORMAL)
      CONTINUE
    ELSE
      PERFORM EVALUATE-LINK-RESPONSE
      GO TO CLEAR-EM-OUT
    END-IF
  END-IF
  CONTINUE.
SEND-PAGE-EMAIL.
  MOVE ZEROES TO COMM-IP-ADDRESS
  ADD 123456789 TO COMM-IP-ADDRESS
  IF COMM-PAGE-NUM NOT = 999000

MOVE 'LINK    ' TO ERR-CALL-TYPE
EXEC CICS LINK PROGRAM('TCPIPØØE')
    COMMAREA(PAGE-COMMAREA)
    LENGTH(LENGTH OF PAGE-COMMAREA)
    RESP(LINK-RESPONSE) END-EXEC
IF LINK-RESPONSE = DFHRESP(NORMAL)
    CONTINUE
ELSE
    PERFORM EVALUATE-LINK-RESPONSE
    GO TO CLEAR-EM-OUT
END-IF
END-IF
CONTINUE.
START-TRANSACTION.
MOVE 'START   ' TO ERR-CALL-TYPE
EXEC CICS START TRANSID('TCPL') AFTER
    SECONDS(RESTART-INTERVAL)
    REQID('STRTTCPL')
    RESP(START-RESPONSE) END-EXEC
IF START-RESPONSE = DFHRESP(NORMAL)
    CONTINUE
ELSE
    PERFORM EVALUATE-START-RESPONSE
    GO TO CLEAR-EM-OUT
END-IF
CONTINUE.
EVALUATE-START-RESPONSE.
EVALUATE START-RESPONSE
WHEN DFHRESP(INVREQ)
    MOVE 'INVREQ    ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(IOERR)
    MOVE 'IOERR     ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(ISCINVREQ)
    MOVE 'ISCINVREQ ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(LENGERR)
    MOVE 'LENGERR   ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(NOTAUTH)
    MOVE 'NOTAUTH   ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(SYSIDERR)
    MOVE 'SYSIDERR  ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(TRANSIDERR)
    MOVE 'TRANSIDERR' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(TERMIDERR)
    MOVE 'TERMIDERR ' TO ERR-TYPE
    MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(TRANSIDERR)
MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
END-EVALUATE.
PERFORM WRITEQ-TRANSIENT-DATA CONTINUE.
EVALUATE-LINK-RESPONSE.
EVALUATE LINK-RESPONSE WHEN DFHRESP(INVREQ)
  MOVE 'INVREQ ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(LENGERR)
  MOVE 'LENGERR ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(NOTAUTH)
  MOVE 'NOTAUTH ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(PGMIDERR)
  MOVE 'PGMIDERR ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN TERMINATED' TO ERR-TEXT
WHEN DFHRESP(SYSIDERR)
  MOVE 'SYSIDERR ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-
WHEN DFHRESP(TERMERR)
  MOVE 'TERMERR ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-
END-EVALUATE.
PERFORM WRITEQ-TRANSIENT-DATA CONTINUE.
WRITEQ-TRANSIENT-DATA.
EXEC CICS WRITEQ TD QUEUE('CSMT') FROM(ERROR-MESSAGE)
  LENGTH(LENGTH OF ERROR-MESSAGE)
  RESP(TRANSDATA-RESPONSE)
END-EXEC
IF TRANSDATA-RESPONSE = DFHRESP(NORMAL)
  CONTINUE ELSE
  MOVE 'WRITE TD' TO OPR-CALL-TYPE
  MOVE 'SEVERE ERR' TO OPR-TYPE
  MOVE 'TRANSIENT DATA WRITE FAILED' TO OPR-TEXT
  EXEC CICS WRITE OPERATOR
    TEXT(OPERATOR-MESSAGE)
    TEXTLENGTH(LENGTH OF OPERATOR-MESSAGE)
    RESP(OPERATOR-RESPONSE)
END-EXEC
END-IF
CONTINUE.
ABEND-ERR.
EXEC CICS ASSIGN ABCODE(ERR-TYPE) END-EXEC
MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-TEXT
PERFORM WRITEQ-TRANSIENT-DATA
GO TO CLEAR-EM-OUT.
CLEAR-EM-OUT.
  EXEC CICS RETURN END-EXEC
STOP RUN.

TCPIP00B
This program’s source is not included because its function is not germane to what this article is attempting to show. It determines whether an error condition (page) has been added to the file and retrieves the data needed by subsequent modules. The data is returned in the communications area.

TCPIP00C
This program is the shortest and easiest one to study. It is used to send a numeric value to a TCP/IP LISTENER module running on a desktop PC. This could be anywhere on the IP network that your installation can communicate with. In the example, it would likely be on the LAN. The LISTENER on the desktop receives and logs the number (sound key) and uses this key to build the name of the sound file that is to be played (999001.wav). This is accomplished by using the TCPIP SOCKETS interface from within CICS to connect to the desktop and subsequently send the small piece of data. The basic use of the SOCKETS interface is almost identical in the other programs, but they have the necessary code differences interleaved into the appropriate places. If you are not familiar with the SOCKETS interface then I recommend that you study this module to get a basic understanding.

ID DIVISION.
PROGRAM-ID. TCPIP00C.
DATE-WRITTEN. DEC 1999.
****************************************************************
* CICS TCP/IP SOCKET INTERFACE PROGRAM--SEND TO SOUND-LISTENER
****************************************************************
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
  77 ABSTIME                  PIC S9(15) COMP-3.
  77 TRANSDATA-RESPONSE       PIC S9(8) COMP.
  77 OPERATOR-RESPONSE        PIC S9(8) COMP.
  77 ABEND-CODE               PIC X(04).
77 SYSTEM-ID PIC X(04).
01 FILLER PIC X(13) VALUE 'TCPIP PARMS >'.
01 SOCKET-GROUP.
   03 SOC-FUNCTION PIC X(16) VALUE SPACES.
   03 ERNO PIC 9(8) COMP VALUE ZEROES.
   03 RETCODE PIC S9(8) COMP VALUE ZEROES.
   03 AF PIC 9(8) COMP VALUE 2.
   03 SOCTYPE PIC 9(8) COMP VALUE 1.
   03 PROTO PIC 9(8) COMP VALUE 0.
   03 NAME.
      05 FAMILY PIC 9(4) COMP VALUE 2.
      05 PORT PIC 9(4) COMP VALUE 3000.
      05 IP-ADDRESS PIC 9(8) COMP.
      05 RESERVED PIC X(8) VALUE LOW-VALUES.
   03 FLAGS PIC 9(8) COMP VALUE 0.
   03 SOCKET-DESC PIC 9(4) COMP.
   03 NBYTE PIC 9(8) COMP.
01 BUFFER-LITERAL PIC X(11) VALUE 'BUFFER OUT'>'.
01 BUFFER-OUT.
   02 FILLER PIC 9(06).
01 ERROR-MESSAGE.
   02 ERR-PROGRAM-ID PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-DATE PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-TIME PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-TRANSACT PIC X(04).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-TYPE PIC X(10).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-CALL-TYPE PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-TEXT PIC X(40).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-NUMBER PIC 9(8).
   02 FILLER PIC X(01) VALUE SPACE.
   02 ERR-RETURN-CODE PIC 9(8).
01 OPERATOR-MESSAGE.
   02 OPR-REGION-ID PIC X(04).
   02 FILLER PIC X(01) VALUE SPACE.
   02 OPR-PROGRAM-ID PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 OPR-TYPE PIC X(10).
   02 FILLER PIC X(01) VALUE SPACE.
   02 OPR-CALL-TYPE PIC X(08).
   02 FILLER PIC X(01) VALUE SPACE.
   02 OPR-TEXT PIC X(28).
LINKAGE SECTION.
01 DFHCOMMAREA.
   02 COMM-PAGE-NUM PIC 9(6).
PROCEDURE DIVISION.
MAINLINE.
   PERFORM HANDLE-ABEND
   PERFORM HOUSE-KEEPING
   PERFORM GET-SOCKET
   PERFORM CONNECT-SOCKET
   PERFORM SEND-SOCKET
   PERFORM SOCKET-CLOSE-CALL
   GO TO CLEAR-EM-OUT.
HANDLE-ABEND.
   EXEC CICS HANDLE ABEND LABEL(ABEND-ERR) END-EXEC
   CONTINUE.
HOUSE-KEEPING.
   EXEC CICS ASKTIME
      ABSTIME(ABSTIME)
   END-EXEC
   EXEC CICS FORMATTIME
      ABSTIME(ABSTIME)
      YYMMDD(ERR-DATE) DATESEP
      TIME(ERR-TIME) TIMESEP
   END-EXEC
   EXEC CICS ASSIGN SYSID(SYSTEM-ID) END-EXEC
   MOVE SYSTEM-ID TO OPR-REGION-ID
   MOVE 'TCPIP00C' TO ERR-PROGRAM-ID
      OPR-PROGRAM-ID
   MOVE EIBTRNID TO ERR-TRANSACTION
   CONTINUE.
GET-SOCKET.
   MOVE 'SOCKET' TO SOC-FUNCTION
   CALL 'EZASOKET' USING SOC-FUNCTION
      AF
      SOCTYPE
      PROTO
      ERRNO
      RETCODE
   IF RETCODE < Ø
      MOVE 'TCPIP' TO ERR-CALL-TYPE
      MOVE 'GET SOCKET' TO ERR-TYPE
      MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-TEXT
      PERFORM WRITEQ-TRANSIENT-DATA
   ELSE
      MOVE RETCODE TO SOCKET-DESC
   END-IF
   CONTINUE.
CONNECT-SOCKET.
   MOVE COMM-IP-ADDRESS TO IP-ADDRESS
   MOVE 'CONNECT' TO SOC-FUNCTION
   CALL 'EZASOKET' USING SOC-FUNCTION
   SOCKET-DESC
NAME
ERRNO
RETCODE
IF RETCODE = Ø
  CONTINUE
ELSE
  MOVE 'TCPIP ' TO ERR-CALL-TYPE
  MOVE 'CONNECT ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-TEXT
  PERFORM WRITEQ-TRANSIENT-DATA
END-IF
CONTINUE.
SEND-SOCKET.
MOVE COMM-PAGE-NUM TO BUFFER-OUT
MOVE LENGTH OF BUFFER-OUT TO NBYTE
CALL 'EZACIC04' USING BUFFER-OUT NBYTE
MOVE LENGTH OF BUFFER-OUT TO NBYTE
MOVE 'SEND ' TO SOC-FUNCTION
CALL 'EZASOKET' USING SOC-FUNCTION
  SOCKET-DESC
  FLAGS
  NBYTE
  BUFFER-OUT
  ERRNO
  RETCODE
IF RETCODE < Ø
  MOVE 'TCPIP ' TO ERR-CALL-TYPE
  MOVE 'SEND ' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-TEXT
  PERFORM WRITEQ-TRANSIENT-DATA
ELSE
  CONTINUE
END-IF
CONTINUE.
SOCKET-CLOSE-CALL.
MOVE ZEROES TO RETCODE ERRNO
MOVE 'CLOSE ' TO SOC-FUNCTION
CALL 'EZASOKET' USING SOC-FUNCTION
  SOCKET-DESC
  ERRNO
  RETCODE
IF RETCODE < Ø
  MOVE 'TCPIP ' TO ERR-CALL-TYPE
  MOVE 'CLOSESOKET' TO ERR-TYPE
  MOVE 'TRANSACTION HAS BEEN RESCHEDULED' TO ERR-TEXT
  PERFORM WRITEQ-TRANSIENT-DATA
ELSE
  CONTINUE
END-IF
CONTINUE.
TCPIP00D

This program uses the same basic SOCKETS logic to connect to an SMTP server on the LAN at TCP/IP port 25. You can read about SMTP by using an Internet Portal (search engine) to search on RFC788.

Request for Change 788 defines the protocol and can be reviewed in a minimum of time. However, the necessary commands are coded into this module and executed in the correct sequence to accomplish the connection and transmission of a short e-mail message to the server. This e-mail message is special and has a RCPT TO field that uniquely identifies to the SMTP server an ID belonging to an ALPHA paging application. This application receives messages and dials the paging service via a modem. The SUBJECT field is by definition the telephone number that uniquely identifies the pager. The pager number is obtained by using the COMM-PAGE-NUM field as the key to a simple VSAM file. These records contain the key and the pager
number that is to be notified. Multiple pager numbers are processed reading each subsequent record until the key no longer matches.

```
ID DIVISION.
PROGRAM-ID. TCPIPØØD.
DATE-WRITTEN. DEC 1999.
******************************************************************
* CICS TCP/IP SOCKET INTERFACE PROGRAM--EMAIL SMTP PAGING SERVICE
******************************************************************
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
77 ABSTIME          PIC S9(15) COMP-3.
77 EML-REC-LENGTH   PIC S9(4) COMP VALUE +8Ø.
77 EML-KEY-LENGTH   PIC S9(4) COMP VALUE +12.
77 TRANSDATA-RESPONSE PIC S9(8) COMP.
77 OPERATOR-RESPONSE PIC S9(8) COMP.
77 FILE-RESPONSE    PIC S9(8) COMP.
77 ABEND-CODE       PIC X(Ø4).
77 SYSTEM-ID        PIC X(Ø4).
77 PAGER-SS         PIC 9(Ø2).
Ø1 END-FILE-SW      PIC X(3).
Ø8 END-FILE VALUE 'END'.
Ø1 FILLER           PIC X(13) VALUE 'TCPIP PARMS >'.
Ø1 SOCKET-GROUP.
  Ø2 SOC-FUNCTION    PIC X(16) VALUE SPACES.
  Ø2 ERRNO           PIC 9(8) COMP VALUE ZEROES.
  Ø2 RETCODE         PIC S9(8) COMP VALUE ZEROES.
  Ø2 AF              PIC 9(8) COMP VALUE 2.
  Ø2 SOCTYPE         PIC 9(8) COMP VALUE 1.
  Ø2 PROTO           PIC 9(8) COMP VALUE Ø.
  Ø2 NAME.
    Ø4 FAMILY        PIC 9(4) COMP VALUE 2.
    Ø4 PORT          PIC 9(4) COMP VALUE 25.
    Ø4 IP-ADDRESS    PIC 9(8) COMP.
    Ø4 RESERVED      PIC X(8) VALUE LOW-VALUES.
  Ø2 FLAGS          PIC 9(8) COMP VALUE Ø.
  Ø2 SOCKET-DESC    PIC 9(4) COMP.
  Ø2 FILLER         PIC X(1Ø) VALUE 'LENGTH--->'.
  Ø2 NBYTE          PIC 9(8) COMP.
Ø1 FILLER           PIC X(10) VALUE 'BUFFER-IN>'.
Ø1 BUFFER-IN.
  Ø2 SMTP-DATA.
    Ø4 SMTP-RETURN-CODE PIC 9(Ø3).
    Ø4 SMTP-RETURN-MESSAGE PIC X(253).
Ø1 FILLER           PIC X(11) VALUE 'BUFFER-OUT>'.
Ø1 BUFFER-OUT.
  Ø2 FILLER         PIC X(Ø8).
Ø1 MAIL-MESSAGE-HEADER.
  Ø2 MAIL-HEADER-LINEØ1.
    Ø4 FILLER PIC X(18)
```
VALUE 'HELO ACOMPANY.COM '.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø2 MAIL-HEADER-LINEØ2.
Ø4 FILLER PIC X(39)
VALUE 'MAIL FROM:<CICS-12345@ACOMPANY.COM> '.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø2 MAIL-HEADER-LINEØ3.
Ø4 FILLER PIC X(34)
VALUE 'RCPT TO: <EMLPAGER@ACOMPANY.COM>'.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø2 MAIL-HEADER-LINEØ4.
Ø4 FILLER PIC X(Ø5)
VALUE 'DATA '.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø1 MAIL-MESSAGE-BODY.
Ø2 MAIL-BODY-LINEØ1.
Ø4 FILLER PIC X(2Ø)
VALUE 'FROM: CICS 12345 '.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø2 MAIL-BODY-LINEØ2.
Ø4 FILLER PIC X(Ø4) VALUE 'TO: '.
Ø4 PAGE-TO.
Ø6 PAGE-TO-ENTRY OCCURS 8 TIMES PIC X(Ø1).
Ø4 FILLER PIC X(Ø6) VALUE '@PAGE '.
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø2 MAIL-BODY-LINEØ3.
Ø4 FILLER PIC X(Ø8) VALUE 'SUBJECT:'.
Ø4 PAGE-SUBJECT.
Ø6 PAGE-SUBJECT-ENTRY OCCURS 8 TIMES PIC X(Ø1).
Ø4 FILLER PIC X(Ø5) VALUE '@PAGE'.
Ø4 FILLER PIC X(Ø4) VALUE X'ØD15D15'.
Ø2 MAIL-BODY-LINEØ4.
Ø4 FILLER PIC X(3Ø).
Ø4 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø1 MAIL-BODY-END.
Ø2 FILLER PIC X(Ø5) VALUE X'ØD154D15'.
Ø1 MAIL-MESSAGE-END.
Ø2 FILLER PIC X(Ø5)
VALUE 'QUIT '.
Ø2 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø1 MAIL-MESSAGE-RSET.
Ø2 FILLER PIC X(Ø5)
VALUE 'RSET '.
Ø2 FILLER PIC X(Ø2) VALUE X'ØD15'.
Ø1 HOLD-EML-RECORD.
Ø2 EML-KEY.
Ø4 EML-RECTYPE PIC X(1).
Ø4 FILLER PIC X(1).
Ø4 EML-COMPONENT PIC 9(6).
Ø4 FILLER PIC X(1).
Ø4 EML-RECORD PIC X(3).
Editor’s note: this article will be concluded in the next issue.

Robert Bilyeu  
Senior Systems Programmer (USA)  
© Xephon 2000
TBS Software has announced SMTP-Send for OS/390, which makes it easy to send e-mail messages, notifications, reports, etc from CICS applications to the outside world using industry-standard SMTP. CICS applications are now able to send information to conventional e-mail users as well as to pagers, mobile phones, and PDAs.

Using the SMTP-Send CICS interface, an application writes its e-mail request to a CICS Temporary Storage Queue, then links to SMTP-Send. Application programmers are insulated from the need to learn arcane e-mail protocols such as SMTP, POP3, IMAP, MIME, etc.

SMTP-Send works with all current releases of CICS. It requires OS/390 (MVS) Version 1.3 or OS/390 Version 2 or higher and IBM TCP/IP Version 3.2 or OS/390 Version 2 Communications Server, and must run SMTP Server.

For further information contact:
TBS Software Inc, 100 Mural Street, Suite 100, Richmond Hill, Ontario, Canada, L4B 1J3.
Tel: (905) 707 9373.

Both support incremental rapid Java application development, a server-side Java programming environment, WebSphere test environment, and Tool Integrator API.

Also included in the Enterprise version are additional and enhanced Enterprise Access Builders for CICS, MQSeries, Encina, TXSeries, Host-On Demand, and SAP R/3 plus enterprise toolkits for OS/390 and OS/400.

For further information contact your local IBM representative.

IBM has announced Version 3.5 of its VisualAge for Java, Professional and Enterprise Editions, which includes a cross-platform object-oriented Java application development tool for extending current information technology assets to the Web.

Both support incremental rapid Java application development, a server-side Java programming environment, WebSphere test environment, and Tool Integrator API.

Also included in the Enterprise version are additional and enhanced Enterprise Access Builders for CICS, MQSeries, Encina, TXSeries, Host-On Demand, and SAP R/3 plus enterprise toolkits for OS/390 and OS/400.

For further information contact your local IBM representative.

IBM has launched its Fault Analyzer and File Manager for OS/390, tools for developing and testing new and existing applications.

Fault Analyzer helps analyse and fix application and system failures, while File Manager helps create, edit, compare, print, and (re)format data files in OS/390 file formats, including QSAM and VSAM.

Products diagnosed in the abend environment include OS/390, CICS, IMS, DB2, COBOL, PL/I, C/C++, Assembler, Language Environment, and Unix System Services.

For further information contact your local IBM representative.