May 1998

In this issue

3 Automatic screen refresh capability
12 CICS statement tool – part 2
25 Date testing CICS applications
37 Transferring code from the Web to a mainframe
38 Terminal auto-install/PRINTTO modification
48 CICS news

© Xephon plc 1998
Editor
Robert Burgess

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.

Subscriptions and back-issues
A year’s subscription to CICS Update, comprising twelve monthly issues, costs £170.00 in the UK; $260.00 in the USA and Canada; £176.00 in Europe; £182.00 in Australasia and Japan; and £180.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £14.50 ($22.00) each including postage.

© Xephon plc 1998. 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.
Automatic screen refresh capability

This article completes the series examining some of the options and features of the API and SPI. A partial discussion of these commands and programs was presented at Xephon’s CICS Update conference held in London in December 1997.

The main topic of this article is how to implement an automatic screen refresh capability.

The source code language used to illustrate the concepts is COBOL written to ANSI 85 standards. The BMS macros provided can be converted to the SDF II (and probably other) screen ‘painting’ packages.

AUTOMATIC REFRESHING

We have probably all seen, and perhaps used, monitor applications that perform an automatic refresh of the display on a periodic basis. These monitors usually run outside CICS, but we can perform a similar function from within CICS. I have written a program which displays the current tasks in the system, or the currently acquired sessions, at a selected interval.

There are three possible ways the task can be started:

- By inputting an initial transaction code.
- By user input requesting refresh or a change in display.
- By time expiry, based on the default or user-specified interval.

The basic problem is that the task can be started by user input or by a previously scheduled request. So the program must determine which way it was started and take appropriate action. Initially, the program must simply gather the data and display it. However, before ending, the task must reschedule itself at the default interval after the current time.

After the transaction has run once, it may be started by expiration of the Interval Control Element (ICE), or by user input if it occurs before
the ICE expires. In the former case, the data normally saved in the COMMAREA is RETRIEVERed in order to determine what to do. In the latter case, the user may have entered a request to switch the type of data displayed or a different refresh period. So the input (if any) must be RECEIVEd and the previous ICE CANCELled. Because the ICE may have expired, and thus turned into an Automatic Initiate Descriptor (AID), the CANCEL may fail. This will happen if CICS was unable to get a successful response to its BID to initiate a conversation. If the CANCEL has failed, then no subsequent START should be issued; if this check was not made, a queue of ICEs could form.

This program uses the STARTCODE option of the ASSIGN command, as does the non-disruptive message delivery program discussed in *Non-disruptive START command, CICS Update*, Issue 149, April 1998.

A special consideration concerns the CANCEL command, which must identify the original START request. This is done by allowing CICS to generate the REQID required, which is saved from the EIBREQID field.

Note that the user is allowed to input data changing the interval for the refresh and/or the type of data to be displayed. The user also has the ability to request the display of a ‘help’ screen.

Of specific interest is the use of the INQUIRE TASK LIST command. This command returns the number of tasks in the system at the time of the request (stored in HOW-MANY) and two areas of storage (TASK-LIST and TRAN-LIST). The first contains a list of task numbers and the second a list of transaction names – with a correspondence between them in relative entries. You should examine the descriptions of these areas defined in the LINKAGE SECTION.

The other interesting aspect is the use of an INQUIRE TERMINAL NEXT loop, when the user requests the display of acquired sessions rather than tasks in the system. Because of this, and the other SPI commands, the program needs to use the SP translator option – which is why the CBL XOPTS(SP) statement is included as the first line of the program.
There are two other points to note:

- The program uses XXXXMAP as the name of the mapset and one of the maps. If you wish to change this, use a global change for that name. It also uses a map name of ZZZZHLP, which can also be changed if required.

- There are two hard-coded ABCODEs in the program for logically incorrect scenarios. You may want to change these.

**PROGRAM SOURCE**

CBL XOPTS(SP)
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPLE.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-_STORAGE SECTION.
Ø1 FILLER.
    Ø3 SCREEN-LIMIT PIC S9(8) COMP VALUE 38.
    Ø3 HOW-MANY PIC S9(8) COMP.
    Ø3 CURRENT-INDEX PIC S9(8) COMP.
    Ø3 FACTYPE PIC S9(8) COMP.
    Ø3 RESPONSE PIC S9(8) COMP VALUE ZERO.
    Ø3 ACQSTATUS PIC S9(8) COMP.
    Ø3 NATURE PIC S9(8) COMP.
    Ø3 WS-CA-LTH PIC S9(4) COMP.
    Ø3 TERMD PIC X(Ø4).
    Ø3 USERID PIC X(Ø8).
    Ø3 TASKNO PIC S9(8) COMP.
    Ø3 TRANAM PIC X(Ø4).
    Ø3 HOW-STARTED.
        Ø5 HS-1 PIC X(Ø1).
            88 TERMINAL-STARTED VALUE 'T'.
            88 ATI-STARTED VALUE 'S'.
        Ø5 FILLER PIC X(Ø1).
        Ø3 CANCEL-IND PIC X(Ø1) VALUE 'Y'.
            88 CANCELLED-OK VALUE 'Y'.
            88 CANCEL-FAILED VALUE 'N'.
    Ø3 TEMP-INTERVAL-X.
        Ø5 TEMP-INTERVAL PIC 9(Ø2).
    Ø1 WS-COMMAREA.
        Ø3 THE-INTERVAL PIC S9(8) COMP VALUE 10 .
        Ø3 WC-MODE PIC X(Ø1) VALUE 'K'.
            88 DISPLAYING-TASKS VALUE 'K'.
            88 DISPLAYING-TERMS VALUE 'N'.

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.

Ø3 WC-HELP
  88 NOT-HELPING
  88 HELPING
Ø3 WC-START-IND
  88 DID-NOT-ISSUE-START
  88 ISSUED-START
Ø3 WC-REQID
COPY ZZZZMAP.

Ø1 OM-ENTRY
  Ø3 FILLER
  Ø3 OM-TASK
  Ø3 OM-DASH1
  Ø3 OM-TRAN
  Ø3 OM-DASH2
  Ø3 OM-TERM
  Ø3 OM-DASH3
  Ø3 OM-USER
  Ø3 OM-BAR
COPY ZZZZMAP.

Ø1 OOPS-MSG.
  Ø3 FILLER
  Ø3 OOPS-ABCODE
  Ø3 FILLER
COPY DFHAID.

LINKAGE SECTION.
Ø1 DFHCOMMAREA

Ø1 TASK-LIST.
  Ø3 TL-TASK
COPY DFHAID.

PROCEDURE DIVISION.
  MOVE LOW-VALUES TO ZZZZMAP0
  EXEC CICS HANDLE ABEND
EXEC CICS ASSIGN
       STARTCODE(HOW-STARTED)
END-EXEC
IF TERMINAL-STARTED
  IF EIBCALEN = LENGTH OF WS-COMMAREA
    MOVE DFHCOMMAREA TO WS-COMMAREA
    EXEC CICS RECEIVE
       MAP('ZZZZMAP')
       NOHANDLE
  END-EXEC
EVALUATE EIBRESP
  WHEN DFHRESP(MAPFAIL)
    CONTINUE
  WHEN DFHRESP(NORMAL)
    PERFORM PROCESS-INPUT
  WHEN OTHER
    EXEC CICS ABEND
       ABCODE('ZNK2')
  END-EXEC
END-EVALUATE
END-IF
IF ISSUED-START
  EXEC CICS CANCEL
     REQID(WC-REQID)
     NOHANDLE
  END-EXEC
  IF EIBRESP NOT = DFHRESP(NORMAL)
    SET CANCEL-FAILED TO TRUE
  END-IF
END-IF
EVALUATE EIBAID
  WHEN DFHPF1
    SET HELPING TO TRUE
  WHEN DFHPF2
    SET NOT-HELPING TO TRUE
    SET DISPLAYING-TASKS TO TRUE
  WHEN DFHPF3
    MOVE EIBTRNID TO EM-TRAN
    EXEC CICS SEND
       FROM(END-MSG)
       ERASE
  END-EXEC
  EXEC CICS RETURN
  END-EXEC
  WHEN DFHPF4
    SET NOT-HELPING TO TRUE
    SET DISPLAYING-TERMS TO TRUE
  WHEN OTHER
SET NOT-HELPING TO TRUE
END-EVALUATE
ELSE
IF ATI-STARTED
MOVE LENGTH OF WS-COMMAREA TO WS-CA-LTH
EXEC CICS RETRIEVE
    INTO(WS-COMMAREA)
    LENGTH(WS-CA-LTH)
    NOHANDLE
END-EXEC
ELSE
EXEC CICS ABEND
    ABCODE('ZNK1')
END-EXEC
END-IF
END-IF
EVALUATE TRUE
WHEN HELPING
    PERFORM SEND-HELP
WHEN DISPLAYING-TASKS
    PERFORM DO-TASKS
WHEN DISPLAYING-TERMS
    PERFORM DO-TERMS
WHEN OTHER
    SET DISPLAYING-TASKS TO TRUE
    PERFORM DO-TASKS
END-EVALUATE
MOVE WC-MODE TO TYPEO
MOVE THE-INTERVAL TO INTERVLO
MOVE EIBTRNID TO THISTRMO
EXEC CICS SEND
    MAP('ZZZZMAP')
    ERASE
END-EXEC
IF CANCELLED-OK
EXEC CICS START
    TRANSID(EIBTRNID)
    TERMID(EIBTRMID)
    AFTER SECONDS(THE-INTERVAL)
    FROM(WS-COMMAREA)
END-EXEC
MOVE EIBREQID TO WC-REQID
SET ISSUED-START TO TRUE
ELSE
    SET DID-NOT-ISSUE-START TO TRUE
END-IF
PERFORM RET-CA
DO-TASKS.
EXEC CICS INQUIRE TASK LIST
LISTSIZE(HOW-MANY)
SET(ADDRESS OF TASK-LIST)
SETTRANSID(ADDRESS OF TRAN-LIST)
END-EXEC
PERFORM VARYING CURRENT-INDEX FROM 1 BY 1
UNTIL CURRENT-INDEX > HOW-MANY
OR CURRENT-INDEX > SCREEN-LIMIT
EXEC CICS INQUIRE TASK(TL-TASK(CURRENT-INDEX))
FACILITY(TERMID)
FACILITYTYPE(FACTYPE)
USERID(USERID)
END-EXEC
MOVE SPACES TO OM-ENTRY
MOVE TL-TASK(CURRENT-INDEX) TO OM-TASK
MOVE TL-TRAN(CURRENT-INDEX) TO OM-TRAN
MOVE USERID TO OM-USER
IF FACTYPE = DFHVALUE(TERM)
  MOVE TERMID TO OM-TERM
END-IF
MOVE ' - ' TO OM-DASH1
OM-DASH2
OM-DASH3
MOVE '  |' TO OM-BAR
MOVE OM-ENTRY TO ENTRYO(CURRENT-INDEX)
END-PERFORM.

DO-TERMS.
EXEC CICS INQUIRE TERMINAL START
END-EXEC
MOVE ZERO TO HOW-MANY
PERFORM VARYING CURRENT-INDEX FROM 1 BY 1
UNTIL RESPONSE = DFHRESP(END)
OR CURRENT-INDEX > SCREEN-LIMIT
EXEC CICS INQUIRE TERMINAL(TERMID) NEXT
ACQSTATUS(ACQSTATUS)
TASKID(TASKNO)
TRANSACTION(TRANAM)
USERID(USERID)
NATURE(NATURE)
RESP(RESPONSE)
END-EXEC
IF RESPONSE NOT = DFHRESP(END)
  IF (TASKNO > ZERO OR
    ACQSTATUS = DFHVALUE(ACQUIRED))
    AND (NATURE NOT = DFHVALUE(SESSION))
    ADD 1 TO HOW-MANY
  MOVE SPACES TO OM-ENTRY
  MOVE TASKNO TO OM-TASK
  MOVE TRANAM TO OM-TRAN
  MOVE TERMID TO OM-TERM

MOVE USERID TO OM-USER
MOVE ' - ' TO OM-DASH1
   OM-DASH2
   OM-DASH3
MOVE '|' TO OM-BAR
MOVE OM-ENTRY TO ENTRYO(CURRENT-INDEX)
ELSE
   SUBTRACT 1 FROM CURRENT-INDEX
ENDIF
END-IF
END-PERFORM
EXEC CICS INQUIRE TERMINAL END
END-EXEC

PROCESS-INPUT.
   MOVE INTREVLI TO TEMP-INTERVAL-X
   MOVE LOW-VALUES TO ZZZZMAPO
   IF TEMP-INTERVAL-X NUMERIC
   AND TEMP-INTERVAL > ZERO
   AND TEMP-INTERVAL < 61
      MOVE TEMP-INTERVAL TO THE-INTERVAL
   ELSE
      MOVE 'Q' TO INTERVLA
   END-IF
SEND-HELP.
   EXEC CICS SEND
      MAP('ZZZZHLP')
      MAPSET('ZZZZMAP')
      MAPONLY
END-EXEC
   SET DID-NOT-ISSUE-START TO TRUE
   PERFORM RET-CA
SEND-HELP.
   EXEC CICS SEND
      MAP('ZZZZHLP')
      MAPSET('ZZZZMAP')
      MAPONLY
END-EXEC
   SET DID-NOT-ISSUE-START TO TRUE
   PERFORM RET-CA
RET-CA.
   EXEC CICS RETURN
      TRANSID(EIBTRNID)
      COMMAREA(WS-COMMAREA)
END-EXEC
OOPS.
   EXEC CICS ASSIGN
      ABCODE(OOPS-ABCODE)
END-EXEC
   EXEC CICS SEND
      FROM(OOPS-MSG)
      ERASE
END-EXEC
   EXEC CICS RETURN
END-EXEC

BMS MACROS

* Main screen

ZZZZMAP  DFHMSD TYPE=SYSPARM,LANG=COBOL,MODE=INOUT,STORAGE=AUTO,
TIOPFX=YES,CTRL=(FREEKB,FRSET)

ZZZZMAP  DFHMDI SIZE=(24,80)

TYPE  DFHMDF POS=(01,14),LENGTH=13,ATTRB=(ASKIP),
      INITIAL='Display type:'

INTERVL  DFHMDF POS=(01,28),LENGTH=1,ATTRB=(PROT,BRT)

THISTRM  DFHMDF POS=(01,62),LENGTH=4,ATTRB=(PROT,BRT)

ENTRY  DFHMDF POS=(04,01),LENGTH=39,ATTRB=(ASKIP,NORM),OCCURS=38

* Help screen

ZZZZHLP  DFHMDI SIZE=(07,50),COLUMN=18,LINE=6

END
This month we complete the code for a tool to help you import the most common CICS statements into a source program.

AS                 OUTLINE(DATA-AREA)
AS                 PAGENUM(DATA-AREA)
AS                 PARTNPAGE(DATA-AREA)
AS                 PARTNS(DATA-AREA)
AS                 PARTNSET(DATA-AREA)
AS                 PRINSYSID(DATA-AREA)
AS                 PROGRAM(DATA-AREA)
AS                 PS(DATA-AREA)
AS                 QNAME(DATA-AREA)
AS                 RESSEC(DATA-AREA)
AS                 RESTART(DATA-AREA)
AS                 RETURNPROG(DATA-AREA)
AS                 SCRNWD(DATA-AREA)
AS                 SIGDATA(DATA-AREA)
AS                 SOSI(DATA-AREA)
AS                 STARTCODE(DATA-AREA)
AS                 STATIONID(DATA-AREA)
AS                 SYSID(DATA-AREA)
AS                 TASKPRIORITY(DATA-AREA)
AS                 TCTUALENG(DATA-AREA)
AS                 TELLERID(DATA-AREA)
AS                 TERMCODE(DATA-AREA)
AS                 TERMPRIORITY(DATA-AREA)
AS                 TEXTKYBD(DATA-AREA)
AS                 TEXTPRINT(DATA-AREA)
AS                 TRANPRIORITY(DATA-AREA)
AS                 TWALENG(DATA-AREA)
AS                 UNATTEND(DATA-AREA)
AS                 USERID(DATA-AREA)
AS                 USERNAME(DATA-AREA)
AS                 USERPRIORITY(DATA-AREA)
AS                 VALIDATION(DATA-AREA)
AT                 ASKTIME
AT                 ABSTIME(DATA-AREA)
CA                 CANCEL
CA                 REQID(NAME)
CA                 TRANSID(NAME)
CA                 SYSID(SYSTEMNAME)
CO                 CONNECT PROCESS
CO                 CONVID(NAME)
CO                 SESSION(NAME)
CO                 PROCNAME(DATA-AREA)
FT MMDDYY(DATA-AREA)
FT DATE(DATA-AREA)
FT DATEFORM(DATA-AREA)
FT DATESEP(DATA-AREA)
FT DAYCOUNT(DATA-AREA)
FT DAYOFWEEK(DATA-AREA)
FT DAYOFMONTH(DATA-AREA)
FT MONTHOFYEAR(DATA-AREA)
FT YEAR(DATA-AREA)
FT TIME(DATA-AREA)
FT TIMESEP(DATA-VALUE)
GM GETMAIN
GM SET(PTR-REF)
GM LENGTH(DATA-VALUE)|FLENGTH(DATA-VALUE) BELOW
GM INITIMG(DATA-VALUE)
GM SHARED
GM NOSUSPEND
GM USERDATAKEY|CICSDATAKEY
HA HANDLE ABEND
HA PROGRAM(NAME)|LABEL(LABEL)|CANCEL|RESET
HC HANDLE CONDITION
HC CONDITION(LABEL)
IABE ISSUE ABEND
IABE CONVID(NAME)
IABE STATE(cvda)
IABO ISSUE ABORT
IABO DESTID(DATA-VALUE)
IABO DESTIDLENG(DATA-VALUE)
IABO CONSOLE
IABO SUBADDR(DATA-VALUE)
IABO PRINT
IABO CARD
IABO WPMEDIA1
IABO WPMEDIA2
IABO WPMEDIA3
IABO WPMEDIA4
IABO VOLUME(DATA-VALUE)
IABO VOLUMELENG(DATA-VALUE)
IADD ISSUE ADD
IADD DESTID(DATA-VALUE)
IADD DESTIDLENG(DATA-VALUE)
IADD VOLUME(DATA-VALUE)
IADD FROM(DATA-AREA)
IADD VOLUMELENG(DATA-VALUE)
IADD LENGTH(DATA-VALUE)
IADD NUMREC(DATA-VALUE)
IADD DEFRESP
IADD NOWAIT
IADD RIDFLD(DATA-AREA)
IADD RRN
ICON
ICON CONVID(NAME)
ICON STATE(CVDA)
ICOPY ISSUE COPY
ICOPY TERMDIENAME
ICOPY CTLCHAR(DATA-VALUE)
ICOPY WAIT
IDISC ISSUE DISCONNECT
IDISC SESSION(NAME)
IEND ISSUE END
IEND DESTID(DATA-VALUE)
IEND DESTIDLENG(DATA-VALUE)
IEND CONSOLE
IEND SUBADDR(DATA-VALUE)
IEND PRINT
IEND CARD
IEND WPMEDIA1
IEND WPMEDIA2
IEND WPMEDIA3
IEND WPMEDIA4
IEND VOLUME(DATA-VALUE)
IEND VOLUMELENG(DATA-VALUE)
IENDF ISSUE ENDFILE
IENDF ENDOUTPUT
IENDOP ISSUE ENDOUTPUT
IENDOP ENDFILE
IEODS ISSUE EODS
IERASE ISSUE ERASE
IERASE DESTID(DATA-VALUE)
IERASE DESTIDLENG(DATA-VALUE)
IERASE VOLUME(DATA-VALUE)
IERASE RIDFLD(DATA-AREA)
IERASE VOLUMELENG(DATA-VALUE)
IERASE KEYLENGTH(DATA-VALUE)
IERASE KEYNUMBER(DATA-VALUE)
IERASE RRN
IERASE NUMREC(DATA-VALUE)
IERASE DEFRESP
IERASE NOWAIT
IERASEAUP ISSUE ERASEAUP
IERASEAUP WAIT
IERR ISSUE ERROR
IERR CONVID(NAME)
IERR STATE(CVDA)
ILOAD ISSUE LOAD
ILOAD PROGRAM(NAME)
ILOAD CONVERSE
INOTE ISSUE NOTE
INOTE DESTID(DATA-VALUE)
INOTE DESTIDLENG(DATA-VALUE)
INOTE              VOLUME(DATA-VALUE)
INOTE              RIDFLD(DATA-AREA)
INOTE              VOLUMELENG(DATA-VALUE)
INOTE              RRN
IPASS              ISSUE PASS
IPASS              LUNAME(NAME)
IPASS              FROM(DATA-AREA)
IPASS              LENGTH(DATA-VALUE)
IPASS              NOQUIESCE
IPASS              LOGMODE(DATA-VALUE)
IPREP              ISSUE PREPARE
IPREP              CONVID(NAME)
IPREP              STATE(CVDA)
IPRINT             ISSUE PRINT
IQUERY             ISSUE QUERY
IQUERY             DESTID(DATA-VALUE)
IQUERY             DESTIDLENG(DATA-VALUE)
IQUERY             VOLUME(DATA-VALUE)
IQUERY             VOLUMELENG(DATA-VALUE)
IREC               ISSUE RECEIVE
IREC               INTO(DATA-AREA)
IREC               SET(PTR-REF)
IREC               LENGTH(DATA-AREA)
IREPL              ISSUE REPLACE
IREPL              DESTID(DATA-VALUE)
IREPL              DESTIDLENG(DATA-VALUE)
IREPL              VOLUME(DATA-VALUE)
IREPL              VOLUMELENG(DATA-VALUE)
IREPL              FROM(DATA-AREA)
IREPL              LENGTH(DATA-VALUE)
IREPL              NUMREC(DATA-VALUE)
IREPL              RIDFLD(DATA-AREA)
IREPL              KEYLENGTH(DATA-VALUE)
IREPL              KEYNUMBER(DATA-VALUE)
IREPL              DEFRESP
IREPL              RRN
IREPL              NOWAIT
IRESET             ISSUE RESET
ISEND              ISSUE SEND
ISEND              DESTID(DATA-VALUE)
ISEND              DESTIDLENG(DATA-VALUE)
ISEND              CONSOLE
ISEND              SUBADDR(DATA-VALUE)
ISEND              PRINT
ISEND              CARD
ISEND              WPMEDIA1
ISEND              WPMEDIA2
ISEND              WPMEDIA3
ISEND              WPMEDIA4
ISEND              VOLUME(DATA-VALUE)
ISEND    VOLUMELENGTH(DATA-VALUE)
ISEND    FROM(DATA-AREA)
ISEND    LENGTH(DATA-VALUE)
ISEND    NOWAIT
ISEND    DEFRESP
ISIGA    ISSUE SIGNAL
ISIGA    CONVID(NAME)
ISIGA    STATE(CVDA)
ISIGL    ISSUE SIGNAL
ISIGL    CONVID(NAME)
ISIGL    SESSION(NAME)
IWAIT    ISSUE WAIT
IWAIT    DESTID(DATA-VALUE)
IWAIT    DESTIDLENGTH(DATA-VALUE)
IWAIT    CONSOLE-
IWAIT    SUBADDR(DATA-VALUE)
IWAIT    PRINT
IWAIT    CARD
IWAIT    WPMEDIA1
IWAIT    WPMEDIA2
IWAIT    WPMEDIA3
IWAIT    WPMEDIA4-
IWAIT    VOLUME(DATA-VALUE)
IWAIT    VOLUMELENGTH(DATA-VALUE)
IC      IGNORE CONDITION
IC      CONDITION ...
IP      INQUIRE
IP      PROGRAM(DATA-VALUE)
IP      CURRENTSTATUS(CVDA)
IP      COBOLTYPE(CVDA)
IP      COPY(CVDA)
IP      DATALLOCATION(CVDA)
IP      ENTRYPOINT(PTR-REF)
IP      EXECKEY(CVDA)
IP      EXECUTIONSET(CVDA)
IP      HOLDSTATUS(CVDA)
IP      LANGUAGE(CVDA)
IP      LENGTH(DATA-AREA)
IP      LOADPOINT(PTR-REF)
IP      LPASTATUS(CVDA)
IP      PROGTYPE(CVDA)
IP      REMOTENAME(DATA-AREA)
IP      REMOTESYSTEM(DATA-AREA)
IP      RESCOUNT(DATA-AREA)
IP      SHARESTATUS(CVDA)
IP      STATUS(CVDA)
IP      TRANSID(DATA-AREA)
IP      USECOUNT(DATA-AREA)
IT      INQUIRE
IT      TRANSACTION(DATA-VALUE)
IT              CMDSEC(CVDA)
IT              DTIMEOUT(DATA-AREA)
IT              DTB(CVDA)
IT              DUMPING(CVDA)
IT              ISOLATEST(CVDA)
IT              PRIORITY(DATA-AREA)
IT              PROFILE(DATA-AREA)
IT              PROGRAM(DATA-AREA)
IT              PURGEABILITY(CVDA)
IT              REMOTENAME(DATA-AREA)
IT              REMOTESYSTEM(DATA-AREA)
IT              RESSEC(CVDA)
IT              ROUTING(CVDA)
IT              RTIMEOUT(DATA-AREA)
IT              RUNAWAY(DATA-AREA)
IT              RUNAWAYTYPE(CVDA)
IT              SCRNSIZE(CVDA)
IT              SHUTDOWN(CVDA)
IT              STATUS(CVDA)
IT              STORAGECLEAR(CVDA)
IT              TASKDATAKEY(CVDA)
IT              TASKDATALOC(CVDA)
IT              TCLASS(DATA-AREA)
IT              TRANCLASS(DATA-AREA)
IT              TRACING(CVDA)
IT              TRPROF(DATA-AREA)
IT              TWASIZE(DATA-AREA)
LI              LINK
LI              PROGRAM(NAME)
LI              COMMAREA(DATA-AREA)
LI              LENGTH(DATA-VALUE)
LI              DATALENGTH(DATA-VALUE)
LI              INPUTMSG(DATA-AREA)
LI              INPUTMSGLEN(DATA-VALUE)
LI              SYSID(SYSTEMNAME)
LI              SYNCONRETURN
LI              TRANSID(NAME)
PO              POST
PO              INTERVAL(Ø|HHMMSS|TIME(HHMMSS)|
PO              AFTER HOURS(HH) MINUTES(MINS) SECONDS(SECS)|
PO              AT HOURS(HH) MINUTES(MINS) SECONDS(SECS)
PO              SET(PTR-REF)
PO              REQID(NAME)
POH             POP HANDLE
PUH             PUSH HANDLE
QS              QUERY SECURITY
QS              RESTYPE(DATA-VALUE)|RESCLASS(DATA-VALUE)
QS              RESIDLENGTH(DATA-VALUE)
QS              RESID(DATA-VALUE)
QS              LOGMESSAGE(CVDA)
QS
ALTER(CVDA)
QS
CONTROL(CVDA)
QS
READ(CVDA)
QS
UPDATE(CVDA)
RB
RESETBR
RB
FILE(FILENAME)
RB
RIDFLD(DATA-AREA)
RB
KEYLENGTH(DATA-VALUE) GENERIC
RB
REQID(DATA-VALUE)
RB
SYSID(SYSTEMNAME)
RB
GTEQ|EQUAL
RB
RBA|RRN
REC
RECEIVE
REC
INTO(DATA-AREA)|SET(PTR-REF)
REC
LENGTH(DATA-AREA)|FLENGTH(DATA-AREA)
REC
MAXLENGTH(DATA-VALUE)|MAXFLENGTH(DATA-VALUE)
REC
NOTRUNCATE
RECA
RECEIVE
RECA
CONVID(NAME)
RECA
INTO(DATA-AREA)
RECA
SET(PTR-REF)
RECA
LENGTH(DATA-AREA)
RECA
FLENGTH(DATA-AREA)
RECA
MAXLENGTH(DATA-VALUE)
RECA
MAXFLENGTH(DATA-VALUE)
RECA
NOTRUNCATE
RECA
STATE(CVDA)
RECD
RECEIVE
RECD
INTO(DATA-AREA)
RECD
SET(PTR-REF)
RECD
LENGTH(DATA-AREA)
RECD
FLENGTH(DATA-AREA)
RECD
MAXLENGTH(DATA-VALUE)
RECD
MAXFLENGTH(DATA-VALUE)
RECD
NOTRUNCATE
RECD
ASIS
RECD
BUFFER
RECM
RECEIVE
RECM
SESSION(NAME)
RECM
INTO(DATA-AREA)
RECM
SET(PTR-REF)
RECM
LENGTH(DATA-AREA)
RECM
FLENGTH(DATA-AREA)
RECM
MAXLENGTH(DATA-VALUE)
RECM
MAXFLENGTH(DATA-VALUE)
RECM
NOTRUNCATE
RECM
STATE(CVDA)
RF
READ
RF
FILE(FILENAME)
RF  UPDATE
RF  INTO(DATA-AREA)|SET(PTR-REF)
RF  LENGTH(DATA-AREA)
RF  RIDFLD(DATA-AREA)
RF  KEYLENGTH(DATA-VALUE) GENERIC
RF  SYSID(SYSTEMNAME)
RF  RBA|RRN|DEBKEY|DEBREC
RF  GTEC|EQUAL
RL  RELEASE
RL  PROGRAM(NAME)
RM  RECEIVE MAP(NAME)
RM  MAPSET(NAME)
RM  INTO(DATA-AREA)|SET(PTR-REF)
RM  FROM(DATA-AREA) LENGTH(DATA-AREA)
RM  TERMINAL(ASIS)
RM  INPARTN(NAME)
RN  READNEXT
RN  FILE(Filename)
RN  INTO(DATA-AREA)|SET(PTR-REF)
RN  LENGTH(DATA-AREA)
RN  RIDFLD(DATA-AREA)
RN  KEYLENGTH(DATA-VALUE)
RN  REQID(DATA-VALUE)
RN  SYSID(SYSTEMNAME)
RN  RBA|RRN
RP  READPREV
RP  FILE(Filename)
RP  INTO(DATA-AREA)|SET(PTR-REF)
RP  LENGTH(DATA-AREA)
RP  RIDFLD(DATA-AREA)
RP  KEYLENGTH(DATA-VALUE)
RP  REQID(DATA-VALUE)
RP  SYSID(SYSTEMNAME)
RP  RBA|RRN
RQTD  READQ TD
RQTD  QUEUE(NAME)
RQTD  INTO(DATA-AREA)|SET(PTR-REF)
RQTD  LENGTH(DATA-AREA)
RQTD  SYSID(SYSTEMNAME)
RQTD  NOSUSPEND
RQTS  READQ TS
RQTS  QUEUE(NAME)
RQTS  INTO(DATA-AREA)|SET(PTR-REF)
RQTD  LENGTH(DATA-AREA)
RQTS  NUMITEMS(DATA-AREA)
RQTS  ITEM(DATA-AREA)|NEXT
RQTS  SYSID(SYSTEMNAME)
RTR  RETRIEVE
RTR  INTO(DATA-AREA)|SET(PTR-REF)
RTR                LENGTH(DATA-AREA)
RTR                RTRANSID(DATA-AREA)
RTR                RTERMID(DATA-AREA)
RTR                QUEUE(DATA-AREA)
RTR                WAIT
RTU                RETURN
RTU                TRANSID(NAME)
RTU                COMMAREA(DATA-AREA)
RTU                LENGTH(DATA-VALUE)
RTU                IMMEDIATE
RTU                INPUTMSG(DATA-AREA)
RTU                INPUTMSGLEN(DATA-VALUE)
RW                 REWRITE
RW                 FILE(Filename)
RW                 FROM(DATA-AREA)
RW                 LENGTH(DATA-VALUE)
RW                 SYSSID(SYSTEMNAME)
SB                 STARTBR
SB                 FILE(Filename)
SB                 RIDFLD(DATA-AREA)
SB                 KEYLENGTH(DATA-VALUE) GENERIC
SB                 REQID(DATA-VALUE)
SB                 SYSSID(SYSTEMNAME)
SB                 RBA|RRN|DEBKEY|DEBREC
SB                 GTEQ|EQUAL
SC                 SPOOLCLOSE
SC                 NOHANDLE
SC                 KEEP
SC                 RESP
SC                 DELETE
SC                 RESP2
SENDA              SEND
SENDA              CONVID(NAME)
SENDA              FROM(DATA-AREA)
SENDA              LENGTH(DATA-VALUE)
SENDA              FLENGTH(DATA-VALUE)
SENDA              INVITE
SENDA              LAST
SENDA              CONFIRM
SENDA              WAIT
SENDA              STATE (CDVA)
SENDD              SEND
SENDD              CONVID(NAME)
SENDD              FROM(DATA-AREA)
SENDD              LENGTH(DATA-VALUE)
SENDD              FLENGTH(DATA-VALUE)
SENDD              INVITE
SENDD              LAST
SENDD              CONFIRM
SENDD  WAIT
SENDD  STATE(CVDA)
SENDM  SEND
SENDM  SESSION(NAME)
SENDM  WAIT
SENDM  IN VITE
SENDM  LAST
SENDM  ATTACHID(NAME)
SENDM  FROM(DATA-AREA)
SENDM  LENGTH(DATA-VALUE)
SENDM  FLENGTH(DATA-VALUE)
SENDM  FMH
SENDM  DEFRESP
SENDM  STATE(CDVA)
SENDT  SEND TEXT
SENDT  FROM(DATA-AREA)
SENDT  LENGTH(DATA-VALUE)
SENDT  CURSOR(DATA-VALUE)
SENDT  FORMFEED
SENDT  ERASE
SENDT  PRINT
SENDT  FREEKB
SENDT  ALARM
SENDT  NLEOM
SENDT  LDC(NAME)|OUTPARTN(NAME)
SENDT  ACTPARTN(NAME)
SENDT  MSR(DATA-VALUE)
SENDT  SET(PTR-REF)|PAGING|
SENDT  TERMINAL WAIT LAST
SENDT  REQID(NAME)
SENDT  HEADER(DATA-AREA)
SENDT  TRAILER(DATA-AREA)
SENDT  JUSTIFY(DATA-VALUE)|JUSFIRST|JUSLAST
SENDT  ACCUM
SENDT  L40|L64|L80|HONEOM
SM  SEND MAP(NAME)
SM  MAPSET(NAME)
SM  FROM(DATA-AREA) DATAONLY|MAPONLY
SM  LENGTH(DATA-VALUE)
SM  CURSOR(DATA-VALUE)
SM  FORMFEED
SM  ERASE|ERASEAUP
SM  PRINT
SM  FREEKB
SM  ALARM
SM  FRSET
SM  NLEOM
SM  MSR(DATA-VALUE)
SM  FMHPARM
SM  LDC(NAME)|OUTPARTN(NAME) ACTPARTN(NAME)
SM
   ACCUM
SM
   SET(PTR-REF)|PAGING
SM
   TERMINAL WAIT LAST
SM
   REQID(NAME)
SM
   NOFLUSH
SM
   L40|L64|L80|HONEOM
SOF
   SIGNOFF
SON
   USERID(DATA-VALUE)
SON
   PASSWORD(DATA-VALUE)
SON
   NEWPASSWORD(DATA-VALUE)
SON
   OIDCARD(DATA-VALUE)
SON
   ESMREASON(DATA-AREA)
SON
   ESMRESP(DATA-AREA)
SON
   GROUPID(DATA-VALUE)
SON
   LANGUAGECODE(DATA-VALUE)
SON
   LANGINUSE(DATA-AREA)
SON
   NATLANG(DATA-VALUE)
SON
   NATLANGINUSE(DATA-AREA)
SR
   SPOOLREAD
SR
   INTO(DATA-AREA)
SR
   MAXFLENGTH(DATA-VALUE)
SR
   TOFLENGTH(DATA-AREA)
SR
   NOHANDLE
SR
   RESP
SR
   RESP2
ST
   START
ST
   INTERVAL(Ø|HHMMSS)|TIME(HHMMSS)|
ST
   AFTER HOURS(HH) MINUTES (MINS) SECONDS (SECS)|
ST
   AT HOURS(HH) MINUTES (MINS) SECONDS (SECS)
ST
   REQID(NAME)
ST
   FROM(DATA-AREA)
ST
   LENGTH(DATA-VALUE) FMH
ST
   TERMID(NAME)
ST
   SYSID(SYSTEMNAME)
ST
   RTRANSID(NAME)
ST
   USERID(DATA-VALUE)
ST
   RTERMID(NAME)
ST
   QUEUE(NAME)
ST
   NOCHECK
ST
   PROTECT
SU
   SUSPEND
SY
   SYNCPOINT
SW
   SPOOLWRITE
SW
   FROM(DATA-AREA)
SW
   LINE
SW
   NOHANDLE
SW
   FLENGTH(DATA-VALUE)
SW
   PAGE
SW
   RESP
SW                 RESP2
UL                 UNLOCK
UL                 FILE(Filename)
UL                 SYSID(SystemName)
VP                 VERIFY
VP                 PASSWORD(Data-Value)
VP                 USERID(Data-Value)
VP                 CHANGETIME(Data-Area)
VP                 DAYSLEFT(Data-Area)
VP                 ESMREASON(Data-Area)
VP                 ESMRESP(Data-Area)
VP                 EXPIRYTIME(Data-Area)
VP                 INVALIDCOUNT(Data-Area)
VP                 LASTUSETIME(Data-Area)
WF                 WRITE
WF                 FILE(Filename)
WF                 MASSINSERT
WF                 FROM(Data-Area)
WF                 LENGTH(Data-Value)
WF                 RIDFLD(Data-Area)
WF                 KEYLENGTH(Data-Value)
WF                 SYSID(SystemName)
WF                 RBA|RRN
WQTD               WRITEQ TD
WQTD               QUEUE(NAME)
WQTD               FROM(Data-Area)
WQTD               LENGTH(Data-Value)
WQTD               SYSID(SystemName)
WQTS               WRITEQ TS
WQTS               QUEUE(NAME)
WQTS               FROM(Data-Area)
WQTS               LENGTH(Data-Value)
WQTS               NUMITEMS(Data-Area)|ITEM(Data-Area) REWRITE
WQTS               SYSID(SystemName)
WQTS               MAIN|AUXILIARY
WQTS               NOSUSPEND
WO                 WRITE OPERATOR
WO                 TEXT (DATA-VALUE)
XCTL               XCTL
XCTL               PROGRAM(NAME)
XCTL               COMMAREA(Data-Area)
XCTL               LENGTH(Data-Value)
XCTL               INPUTMSG(Data-Area)
XCTL               INPUTMSGLEN(Data-Value)

Paul Jansen (with the permission of Marco Seesing and Martijn Bosschieter)
Systems Programmer
Interpay/BankGiroCentrale (The Netherlands) © M Bosschieter/M Seesing 1998
Date testing CICS applications

As we approach the year 2000, testing CICS applications with a different system date, usually a date in the future, has become a hot issue. There are numerous products available that allow you to change the date for a whole CICS region. This has the drawback that all transactions running in the CICS region will have the same date.

We decided to take another approach. Our IY2K solution allows each user in the CICS region to establish his private ‘future’ date. Now our CICS application people can jump backwards and forwards in time, switching dates as they like, without disturbing other users in the CICS region or having to recycle the CICS region.

CICS programs get access to date and time in two main ways. The first way is by accessing the EIBDATE and EIBTIME fields in the EXEC Interface Block and, after the EIBDATE and EIBTIME have been updated, using the EXEC CICS ASKTIME statement. The second way is by using the EXEC CICS ASKTIME ABSTIME() statement, followed by the EXEC CICS FORMATTIME statement to translate the ABSTIME value into readable date and time values.

The IY2K solution intercepts both ways of accessing date and time, using CICS Global User Exits (GLUEs). The XEIOUT GLUE allows you to change the output from every EXEC CICS statement issued. We use it to change the output from the EXEC CICS ASKTIME and EXEC CICS ASKTIME ABSTIME() statement. The XPCFTCH GLUE gets control every time CICS fetches a program. We use the XPCFTCH exit to change the EIBDATE value at task initiation, so that the task starts with the private ‘future’ date.

Every user of IY2K stores the shifted date in a piece of shared storage. Every time a date is requested, the shared storage is checked to see whether the date needs to be shifted. The same logic is applied at transaction initiation time.

The IY2K solution consists of five programs, one map, and three transaction definitions. Firstly, we coded both GLUE programs – IPPCEIOU is the GLUE which gets control at the XEIOUT exit point,
and IPPPCPCFT gets control at the XPCFTCH exit point. Two other programs were created, allowing us to enable and/or disable both exit programs: IPPCEIEN is the program enabling and starting the exits, and IPPCEIDI is the program needed for disabling and stopping the exits. EIEN is the transaction definition used for IPPCEIEN, and EIDI is the transaction definition for IPPCEIDI.

The exit-enabling program logic is mainly as follows: we enable the IPPCEIEN at exit point XEIOUT (without starting it), and request a Global Work Area (GWA) of 8 bytes (this GWA is passed to the IPPCEIOU exit program at exit invocation). Instead of allocating a large GWA, we prefer to allocate a piece of shared storage and just store the address in the GWA. Next we start the IPPCEIOU exit program. We then enable the IPPPCPCFT at exit point XEIOUT and acquire a GWA of 8 bytes. We also plug the address of the shared storage in the GWA for IPPPCPCFT, and start the IPPPCPCFT exit program.

IPPCEIEN

    TITLE 'IPPCEIEN - ENABLE XEIOUT AND XPCFTCH - IY2K'
    SPACE 2
    IPPCEIEN AMODE 31
    IPPCEIEN RMODE ANY
    SPACE 2

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

* INVOKED BY THE EIEN TRANSACTION AND USED IN PLTP
* ENABLE BOTH EXIT PROGRAMS (IPPCEIOU AND IPPPCPCFT)
* - ENABLE IPPCEIEN PROGRAM
  * - GET SHARED STORAGE
  * - EXTRACT GLOBAL AREA ADDRESS
  * - FILL THE ADDRESS OF THE SHARED STORAGE IN THE GWA
  * - START THE IPPCEIEN EXIT PROGRAM
  * - ENABLE IPPPCPCFT PROGRAM
  * - EXTRACT GLOBAL AREA ADDRESS
  * - FILL THE ADDRESS OF THE SHARED STORAGE IN THE GWA
  * - START THE IPPPCPCFT EXIT PROGRAM

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

    SPACE
    DFHREGS
SPACE
RGA EQU 9
RW EQU 10
EJECT

* CICS WORKING STORAGE - DYNAMIC USER STORAGE *

DFHEISTG DSECT
GALEN DS H
SPACE 2
GASTORD DSECT
GAEYEC DS CL4
GASHAR DS CL4
SPACE 2
SHARD DSECT
SHAREYEC DS CL4
SPACE 2
YES EQU X'FF'
NO EQU X'00'
SPACE 2

* CICS CODING *

SPACE 2
IPPCEIEN DFHEIENT CODEREG=(R11),EIBREG=(R12),DATAREG=(R13)

B START
DC CL9'IPPCEIEN'
DC CL9'&SYSDATE'
DC CL9'&SYSTIME'
START DS 0H

EXEC CICS ADDRESS EIB(R12)

MVC GALEN,=H'8' LENGTH OF GLOBAL EXIT AREA

* ENABLE THE IPPCEIOU EXIT PROG, WITHOUT STARTING IT (WE NEED THE GWA)

EXEC CICS ENABLE PROGRAM('IPPCEIOU') EXIT('XEIOUT') GALENGTH(GALEN) NOHANDLE

CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
BE EXTRGWAE

EXEC CICS WRITE OPERATOR TEXT(ERRMSG1) TEXTLENGTH(OPMSGLEN) NOHANDLE
B RETURN

* OBTAIN THE GWA ADDRESS FROM THE IPPCEIOU EXIT PROGRAM
* EXTRGWAE DS ØH
  EXEC CICS EXTRACT EXIT PROGRAM('IPPCEIOU') C
  GASET(RGA) GALENGTH(GALEN) C
  NOHANDLE
*
  CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
  BE GETSTOR
*
  EXEC CICS WRITE OPERATOR TEXT(ERRMSG2) TEXTLENGTH(OPMSGLEN) C
  NOHANDLE
  B RETURN
*
GETSTOR DS ØH
USING GASTORD,RGA
MVC GAECI,=CL4'GAEI'
*
* NOW GET THE PIECE OF SHARED STORAGE
*
  EXEC CICS GETMAIN FLENGTH(16384) INITIMG(X'ØØ') C
  SHARED C
  SET(R2) C
  NOHANDLE
*
  CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
  BE ENABEI
*
  EXEC CICS WRITE OPERATOR TEXT(ERRMSG3) TEXTLENGTH(OPMSGLEN) C
  NOHANDLE
  B RETURN
*
* SAVE THE ADDRESS IN THE GWA AND START THE IPPCEIOU EXIT PROGRAM
*
ENABEI DS ØH
ST R2,GASHAR SAVE ADDRESS IN GLOBAL STORAGE
USING SHARD,R2
MVC SHAREYEC,=CL4'EISH'
EXEC CICS ENABLE PROGRAM('IPPCEIOU') START C
NOHANDLE
*
  CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
  BE ENABPCFT
*
  EXEC CICS WRITE OPERATOR TEXT(ERRMSG4) TEXTLENGTH(OPMSGLEN) C
  NOHANDLE
  B RETURN
*
ENABPCFT DS ØH
EXEC CICS WRITE OPERATOR TEXT(ERRMSG5) TEXTLENGTH(OPMSGLEN) C
NOHANDLE
* ENABLE THE IPPPCPCFT EXIT PROG, WITHOUT STARTING IT (WE NEED THE GWA) *
EXEC CICS ENABLE PROGRAM('IPPCPCFT') EXIT('XPCFTCH') C
   GALENGTH(GALEN) C
   NOHANDLE
*
CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
   BE EXTRGWAP
*
EXEC CICS WRITE OPERATOR TEXT(ERRMSG6) TEXTLENGTH(OPMSGLEN) C
   NOHANDLE
   B RETURN
*
* OBTAIN THE GWA ADDRESS FROM THE IPPPCPCFT EXIT PROGRAM *
EXTRGWAP DS ØH
EXEC CICS EXTRACT EXIT PROGRAM('IPPCPCFT') C
   GASET(RGA) GALENGTH(GALEN) C
   NOHANDLE
*
CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
   BE ENABPC
*
EXEC CICS WRITE OPERATOR TEXT(ERRMSG7) TEXTLENGTH(OPMSGLEN) C
   NOHANDLE
   B RETURN
*
* SAVE THE SHARED STORAGE ADDRESS IN THE GWA AND START IPPPCPCFT *
ENABPC DS ØH
   USING GASTORD,RGA
   MVC GAEYEC,=C'GAPC'
   ST R2,GASHAR
EXEC CICS ENABLE PROGRAM('IPPCPCFT') START C
   NOHANDLE
*
CLC EIBRESP,DFHRESP(NORMAL) RESPONSE NORMAL?
   BE ALLOK
*
EXEC CICS WRITE OPERATOR TEXT(ERRMSG8) TEXTLENGTH(OPMSGLEN) C
   NOHANDLE
   B RETURN
*
ALLOK DS ØH
EXEC CICS WRITE OPERATOR TEXT(ERRMSG9) TEXTLENGTH(OPMSGLEN) C
   NOHANDLE
   B RETURN
EJECT ERRMSG1 DC CL60'IPPCEIEN - IPPCEIOU - INITIAL ENABLE FAILED'

The shared storage is big enough to allow 500 terminals to set their own date.

IPPCEIOU

The IPPCEIOU exit program is fairly simple. It analyses the EXEC CICS request, and intercepts the ASKTIME and ASKTIME ABSTIME() requests. If one of those is entered, the piece of shared storage is accessed to see whether we need to adjust the EIBDATE field for this terminal. If the request was issued with the ABSTIME parameter, we adjust the parameter passed to return the date requested by the user.

```
TITLE 'IPPCEIOU - EXEC INTERFACE EXIT PROGRAM'
SPACE
*
* THIS PROGRAM IS A GLUE AT THE XEIOUT EXIT POINT. IT INTERCEPTS THE EXEC CICS ASKTIME AND EXEC CICS ASKTIME ABSTIME() CALLS, AND RETURNS THE REQUESTED PRIVATE DATE, IF NEEDED
* DO *NOT* USE THE CICS TRANSLATOR FOR THIS PROGRAM !!!
* MAIN LOGIC
* - CHECK IF IT WAS ASKTIME OR ASKTIME ABSTIME()
* - CHECK THE SHARED STORAGE FOR THIS TERMINAL ..
* - IF FOUND, ADJUST THE EIBDATE FIELD
* - IF IT WAS AN ABSTIME REQUEST, ADJUST THE PARAMETER PASSED
DISPLAY
*SPACE
```
RØ       EQU   Ø                   NOT USED
R1       EQU   1                   INITIAL USER EXIT PARAMETER LIST
R2       EQU   2                   USER EXIT PARAMETER LIST
R3       EQU   3                   XEIOUT  GLOBAL WORK AREA ADDRESS
R4       EQU   4                   NOT USED
R5       EQU   5                   NOT USED
R6       EQU   6                   NOT USED
R7       EQU   7                   NOT USED
R8       EQU   8                   NOT USED
R9       EQU   9                   NOT USED
R10      EQU   10                  NOT USED
R11      EQU   11                  NOT USED
R12      EQU   12                  PROGRAM BASE
R13      EQU   13                  SAVE AREA
R14      EQU   14                  RETURN ADDRESS
R15      EQU   15                  INITIAL PROGRAM BASE

*———* THIS MACRO ESTABLISHES THE GLUE *———*

SPACE 2
DFHUEXIT TYPE=EP,ID=(XEIOUT)
EJECT
DFHEIBR EQU   R6
COPY DFHEIBLK
EJECT

*———* THE LAYOUT OF THE GWA FOR THE IPPCEIOU PROGRAM *———*

SPACE 2
GASTORD DSECT
GAEYEC   DS    CL4    AN EYECATCHER
GASHAR   DS    CL4    THE ADDRESS OF THE SHARED STORAGE
EJECT

*———* THE LAYOUT OF THE SHARED STORAGE *———*

SPACE 2
SHSTORD DSECT
SHEYEC   DS    CL4    AN EYECATCHER
SHCNT    DS    CL2    THE NUMBER OF ACTIVE ENTRIES
SHLINE   DS    CL1    ADDRESS OF FIRST "ROW"
EJECT

*———* THE LAYOUT OF A ROW IN THE SHARED STORAGE *———*

SHLINED DSECT
SHFTER   DS    CL4    TERMINAL ID
SHFDAT   DS    PL4    FUTURE DATE (IN EIBDATE FORMAT)
SHFABS   DS    PL8    ABSTIME DIFFERENCE
SHFDAY   DS    CL2    FUTURE DAY
SHFMON   DS    CL2    FUTURE MONTH
SHFYEA DS Cl4 FUTURE YEAR
SHLINEL EQU -*SHFTER
EJECT

* THE LAYOUT OF THE COMMAND PARAMETER LIST (POINTED BY UEPARG)
*

SPACE 2
EPARGD DSECT
EPARGØ DS F ARGUMENT Ø (STARTS WITH THE 2-BYTE FUNCTION CODE)
EPARG1 DS F ARGUMENT 1 (THE OUTPUT FOR THE ASKTIME PARAM)
EJECT

* START AS A NORMAL PROGRAM
*

SPACE 2
IPPCEIOU CSECT
SPACE 2
IPPCEIOU AMODE 31
IPPCEIOU RMODE ANY
SPACE 2
SAVE (14,12) SAVE REGS
LR R12,R15 SET-UP BASE REGISTER
USING IPPCEIOU,R12 ADDRESSABILITY
LR R2,R1 GET UEP PARAMETER LIST
USING DFHUEPAR,R2 ADDRESSABILITY
SPACE 2
L R3,UEPGAA GET GWA ADDRESS
USING GASTORD,R3 ADDRESSABILITY
SPACE
L R4,UEPARG ADDRESS THE EXEC CICS ARGUMENTS
USING EPARGD,R4
L R5,EPARGØ
*

CLC Ø(2,R5),=X'1002' IF THIS IS ASKTIME
BE CHCKDATE
CLC Ø(2,R5),=X'4A02' OR THIS IS ASKTIME ABSTIME
BNE RETURN
*

CHCKDATE DS ØH
*

L R6,UEPEXECB ADDRESS THE EIB
USING DFHEIBLK,R6
*

CLC EIBTRNID,=C'IY2K' IY2K ALWAYS RETURNS SYSTEM DATE
BE RETURN
*

L R7,GASHAR ADDRESS OF THE SHARED STORAGE
USING SHSTORD,R7
LH R8,SHCNT NUMBER OF ACTIVE ENTRIES
LA R9,SHLINE ADDRESS THE FIRST ROW
USING SHLINED,R9

* CHECKLOOP DS ØH   TILL ALL ENTRIES CHECKED
  LTR R8,R8
  BZ RETURN

*  CLC SHFTER,EIBTRMID DO WE HAVE A HIT?
  BNE CHECKNEXT CHECK THE NEXT ONE

*  MVC EIBDATE(4),SHFDAT MOVE THE FUTURE DATE FOR THIS TERM

*  CLC Ø(2,R5),=X'4AØ2' IF THIS WAS AN ASKTIME ABSTIME
  BNE DATEDONE EIBDATE IS UPDATED

*  L R5,EPARG1 ADDRESS THE ABSTIME ARGUMENT
  AP Ø(8,R5),SHFABS ADD THE OFFSET (MIGHT BE NEGATIVE)

* DATEDONE DS ØH
  XR R8,R8
  B RETURN GET OUT

* CHECKNEXT DS ØH
  LA R9,SHLINEL(9) POINT TO NEXT ROW
  BCTR R8,RØ ONE MORE PROCESSED
  B CHECKLOOP

* DROP R9
  DROP R7
  DROP R6
  DROP R4
  DROP R3
  EJECT

*SPACE

* END AS A NORMAL PROGRAM

*SPACE

RETURN DS ØH RETURN TO THE CALLER
  L R13,UEPEPSA ADDRESS OF EXIT SAVE AREA
  RETURN (14,12),RC=UERCNORM RESTORE REGS AND RETURN

*SPACE

IPPCPCFT
The IPPCPCFT exit program uses almost the same logic, the only
difference being that access to the EIB is not so straightforward in the
XPCFTCH exit point. We need to use an XPI call to access the EIB. Once we have the EIB address, we can change the EIBDATE value to the date requested by the user.

```
TITLE 'IPPCPCFT - PROGRAM FETCH - EXIT PROGRAM'
SPACE 2

* ————————————————————————————————————
 *
* THIS PROGRAM IS A GLUE AT THE XPCFTCH EXIT POINT. IT MODIFIES THE
* EIBDATE FIELD, RETURNING THE REQUESTED PRIVATE DATE, IF NEEDED
*
* DO *NOT* USE THE CICS TRANSLATOR FOR THIS PROGRAM !!!
*
* MAIN LOGIC
* - CHECK IF THE LOGICAL LEVEL WAS LESS OR EQUAL 1
* - ADDRESS THE EIB USING THE XPI DFHAPIQX CALL
* - CHECK THE SHARED STORAGE FOR THIS TERMINAL ..
* - IF FOUND, ADJUST THE EIBDATE FIELD
*
* ————————————————————————————————————
EJECT                                                     @L1A
* ————————————————————————————————————
*
* THIS MACRO ESTABLISHES THE GLUE
*
SPACE
DFHUEXIT TYPE=EP,ID=(XPCFTCH)
EJECT                                                     @L1A
* ————————————————————————————————————
*
* THIS MACRO ESTABLISHES THE XPI ENVIRONMENT
*
SPACE
DFHUEXIT TYPE=XPIENV
EJECT
*
* ————————————————————————————————————
*
* THIS MACRO ESTABLISHES THE LAYOUT FOR THE XPCFTCH GLUE PARAMETERS
*
SPACE
COPY DFHPCUE
EJECT
*
* ————————————————————————————————————
*
* THIS MACRO ESTABLISHES THE LAYOUT FOR THE INQ_APPLICATION_DATA XPI
*
SPACE
COPY DFHAPIQY
EJECT
COPY DFHEIBLK
DFHEIBR EQU 11
EJECT
*
* ————————————————————————————————————
*
* THE LAYOUT OF THE GWA FOR THE IPPCEIOU PROGRAM
*
```

GASTORD DSECT
  GAEYEC DS CL4 AN EYECATCHER
  GASHAR DS CL4 THE ADDRESS OF THE SHARED STORAGE
EJECT

* THE LAYOUT OF THE SHARED STORAGE
*

SHSTORD DSECT
  SHEYEC DS CL4 AN EYECATCHER
  SHCNT DS CL2 THE NUMBER OF ACTIVE ENTRIES
  SHLINE DS CL1 ADDRESS OF FIRST "ROW"
EJECT

* THE LAYOUT OF A ROW IN THE SHARED STORAGE
*

SHLINED DSECT
  SHFTER DS CL4 TERMINAL-ID
  SHFDAT DS PL4 FUTURE DATE (IN EIBDATE FORMAT)
  SHFABS DS PL8 ABSTIME DIFFERENCE
  SHFDAY DS CL2 FUTURE DAY
  SHFMON DS CL2 FUTURE MONTH
  SHFYEA DS CL4 FUTURE YEAR
  SHLINEL EQU *-SHFTER
EJECT

* START AS A NORMAL PROGRAM
*

IPPCPCFT CSECT
  SPACE
  IPPPCPCFT AMODE 31
  IPPPCPCFT RMODE ANY
  SPACE
  SAVE (14,12) SAVE REGS
  LR R12,R15 SET-UP BASE REGISTER
  USING IPPPCPCFT,R12 ADDRESSABILITY
  B START
  DC CL9'IPPPCPCFT'
  DC CL9'&SYSDATE'
  DC CL9'&SYSTIME'
  *
  START DS 0H
  *
    LR R2,R1 GET UEP PARAMETER LIST
    USING DFHUEPAR,R2 ADDRESSABILITY
    SPACE 2
    L R3,UEPGAA GET GWA ADDRESS
    USING GASTORD,R3 ADDRESSABILITY
    SPACE
    L R4,UEPPCDS ADDRESS THE XPCFTCH PARAM LIST
    USING DFHPCUE,R4

*  CLC  PCUE_LOGICAL_LEVEL,=F'1'  CHECK ON LOGICAL LEVEL  
  BH  RETURN  
*  CLC  PCUE_PROGRAM_NAME,=CL8'IPPCIY2K'  IY2K ALWAYS SYSTEM DATE  
  BE  RETURN  
*  L  R5,UEPXSTOR  PREPARE FOR THE XPI CALL  
  USING  DFHAPIQ_ARG,R5  ADDRESS THE PARAM LIST  
  L  R13,UEPSTACK  REQUIRED BY XPI INTERFACE  
*  INQ APPLICATION DATA RETURNS THE ADDRESS OF THE EIB  
*  DFHAPIQX CALL,CLEAR,IN,FUNCTION(INQ_APPLICATION_DATA),  
  X  OUT,EIB((R11)),RESPONSE(*),REASON(*)  
*  CLI  APIQ_RESPONSE,APIQ_OK ERROR OCCURRED ...  
  BE  CHKSTOR  
*  WTO  'IPPCPCFT - INQ APPL FAILED',ROUTCDE=(11),DESC=(7)  
  B  RETURN  
*  CHCKSTOR  DS  ØH  
  USING  DFHEIBLK,R11  ADDRESS THE EIB  
  L  R7,GASHAR  ADDRESS THE SHARED STORAGE AREA  
  USING  SHSTORD,R7  
  LH  R8,SHCNT  NUMBER OF ACTIVE ENTRIES  
  LA  R9,SHLINE  ADDRESS THE FIRST ROW  
  USING  SHLINED,R9  
*  CHCKLOOP  DS  ØH  
  LTR  R8,R8  TILL ALL ENTRIES CHECKED  
  BZ  RETURN  
*  CLC  SHFTER,EIBTRMID  DO WE HAVE A HIT ?  
  BNE  CHCKNEXT  
*  MVC  EIBDATE(4),SHFDAT  MOVE THE FUTURE DATE  
  XR  R8,R8  STOP THE LOOP  
  B  RETURN  
*  
** Editor’s note: this article will be continued next month.**  

Stan Adriaensen  
Systems Engineer  
Groupe Royale Belge/IPPA (Belgium)  
© Xephon 1998
Transferring code from the Web to a mainframe

Editor’s note: although this article was written by an MVS Update subscriber, the ISPF edit macro, or a modified version (once you’ve identified the ‘before’ and ‘after’ hex codes at your site), can be used to overcome problems experienced when downloading Update code to a mainframe.

When a colleague of mine recently downloaded an MVS Update article from the Xephon Web site to his PC and then uploaded it to his MVS system, he found to his disappointment that the program code would not run properly.

It was a REXXX program, and, when he executed it, he received the following message:

IRX0013I Error running XXXXXXXX, line nn: Invalid character in program

This was rather puzzling, but a quick look at the code revealed that the offending character was a REXXX ‘not’ (that is ^, in a ^= expression), which should be a hex value X'5F', but was instead a X'B0'. The REXXX interpreter was rejecting this value. Another odd character turned out to be the ‘|’ operator, which should be X'4F', but was X'6A'.

Having discovered this, it was easy to code an ISPF edit macro to fix this and to cater for it in future uploads:

ISREDIT MACRO
ISREDIT CHANGE ALL X'B0' X'5F'
ISREDIT CHANGE ALL X'6A' X'4F'
EXIT

The PC was running IBM Personal Communications 3270 Version 4.1 for Windows with an IEEE 802.2 connection to the host, code page 037. The upload was achieved using the IBM 3270 PC File Transfer Program for MVS/TSO Release 1.1.1 using the following command:

IND$FILE PUT XEPHFILE.TEXT ASCII CRLF RECFM(V) LRECL(133)

It seems that the ASCII to EBCDIC conversion taking place works fine for alphanumeric characters, but is suspect for unusual ones. Readers should be aware of this when transferring code.

Patrick Mullen
MVS Systems Consultant (Canada) © Xephon 1998
Terminal auto-install/PRINTTO modification

If you use auto-install to install terminal definitions you will probably have found that it is difficult to maintain terminal to PRINTTO printer relationships. I have modified the following program, mostly supplied by IBM as DFHZATDX in SDFHSAMP, to read a DB2 table at auto-install time to get the appropriate PRINTTO printer. The DB2 table consists of two four-byte columns. Column one is the four byte terminal-id, which we get from the last four bytes of the VTAM address. The second four bytes are the CICS TERMID of a defined CICS printer. This is plugged into the PRINTTO field for the auto-installed terminal.

There are three scenarios that can develop when a look-up is performed against this table for a given terminal-id:

- The requested terminal is found in the table and the associated printer is retrieved and plugged into the PRINTTO field.
- The requested terminal-id is not found. If this is the case, check whether the controller that the terminal is plugged into is in the table. For our installation we put the control unit address in the first two bytes of the CICS terminal-id. We then append XX to these two bytes to get the entry for the control unit. Assuming that all users not otherwise defined are located in the same general area by control unit, we take the associated printer and plug that into the PRINTTO field and put out a message to that effect to CSSL.
- The terminal and the control unit are not defined. In this case, we put out descriptive messages to CSSL and plug nothing into the PRINTTO field and continue with the auto-install.

Here is an example of these entries in this table:

- LAXXL75E – definition for control unit with address LA.
- LA0BLB1F – definition for terminal LA0B printer LB1F.

If these were the only entries in the table, then terminal LA0B would screen print to printer LB1F. All other terminals plugged into the LA
controller will print to printer L75E. All other terminals logging on to this CICS will have no PRINTTO printer defined.

To my knowledge, this auto-install program only works with CICS 4.1.0. I had to convert my CICS 3.3 system to this prior to bringing up my first 4.1.0 system.

DPKCS101

******************************************************************************
* MODULE NAME = DPKCS101
* DESCRIPTIVE NAME = CICS/ESA(SAMPLE) Terminal auto-install user program (COBOL) @P2A*
* 5655-018
* COPYRIGHT = NONE
* STATUS = 4.1.0
* FUNCTION = Provide user input to terminal auto-install processing. @P2A*
* This module must be compiled with COBOL II compiler.*
* This module is a component of ZCP.*
* It is called via an DFHPC CTYPE=LINK-URM, from DFHZATA (INSTALL) and DFHZATD (DELETE).*
* Input to the module is a parameter list addressed by DFHEICAP.*
* The program is invoked when:
* 1) An auto-install INSTALL is in progress
* 2) An auto-install DELETE has just completed
* The function to be performed is indicated via the passed parameter list. This is evaluated during common initialization processing, and control passed to the appropriate routine.
* Function 1 - INSTALL
* The primary purpose of this function is to complete the SELECTED-PARMS fields. These are used as input to an auto-install resource 'builder' request.

* The following fields may already have been supplied by MTS:
  * SELECTED-MODELNAME
  * SELECTED-PRINTER-NETNAME
  * SELECTED-ALTPRINTER-NETNAME
* The following fields should be set (if not supplied by MTS):
  * SELECTED-MODELNAME
* The following fields should be set:
  * SELECTED-TERM-ID
  * SELECTED-RETURN-CODE
* The following fields may be set:
  * SELECTED-PRINTER-ID
  * SELECTED-ALTPRINTER-ID
* The default action of this program is:
* - If the modelname list contains no elements, then return
* - If the first character of SELECTED-MODELNAME is blank
  *(Not supplied by MTS), then copy the first modelname in
  * MODELNAME-LIST into SELECTED-MODELNAME.
* - Copy last 4 non-blank characters of the passed netname to
  * SELECTED-TERM-ID.
* - Set the SELECTED-RETURN-CODE to RETURN-OK to indicate that
  * a selection has been made.
* - Return to the calling program.
* EXIT-NORMAL =
  * Exit is via an EXEC CICS RETURN command.
  * Status is set to zero if all processing completes normally.
* EXIT-ERROR =
  * Exit is via an EXEC CICS RETURN command.
  * RETURN-CODE is non-zero on entry to this module and is
  * untouched if any error occurs, hence, a non-zero return
  * code is passed back to the calling program.
* Function 2 - auto-install DELETE
* ———————————————
* This function gives the user the opportunity to perform
* processing when an auto-installed terminal has been deleted.
* The default action of this program is to establish
* addressability to the parameter list, and RETURN.
* EXIT-NORMAL =
  * Exit is via an EXEC CICS RETURN command.
* Function 7 & 8 - auto-install of a shipped definition
* ————————————————————————————
* The primary purpose of this function is to validate the
* SELECTED_TERM field. This is used as input to an auto-
* install resource 'builder' request.

* The fields are described in more detail in DFHTCUDS.

* The following input fields are supplied:
  * INSTALL_SHIPPED_CLASH -> Y/N
  * INSTALL_SHIPPED_NETNAME_PTR -> NETNAME_FIELD
  * INSTALL_SHIPPED_TERMID_PTR -> incoming TERMID
  * INSTALL_SHIPPED_APPLID_PTR -> APPLID of TOR
  * INSTALL_SHIPPED_SYSID_PTR -> SYSID of incoming request
  * INSTALL_SHIPPED_CORRID_PTR -> Correlation token

* The following fields should be set on exit:
  * SELECTED_TERM_ID
  * SELECTED_RETURN_CODE

* EXIT-NORMAL =
  * Exit is via an EXEC CICS RETURN command.
  * Status is set to zero if all processing completes normally.

* EXIT-ERROR =
  * Exit is via an EXEC CICS RETURN command.
  * RETURN_CODE is non-zero on entry to this module and is
  * untouched if any error occurs, hence, a non-zero return
  * code is passed back to the calling program.

* Function 1Ø & 11 - auto-install delete of shipped definition

* This function gives the user the opportunity to perform
  * processing when an auto-installed terminal has been deleted.

* The default action of this program is to establish
  * addressability to the parameter list, and RETURN.

* EXIT-NORMAL =
  * Exit is via an EXEC CICS RETURN command.

* ENTRY POINT = DPKCS1Ø1

* PURPOSE = All Functions
  * The request type is analysed, and control passed to the
  * appropriate routine.

* EXTERNAL REFERENCES = None
ROUTINES =
   EXEC CICS RETURN - return to calling program

CONTROL BLOCKS =
   See FUNCTION section for description of input parameters

DESCRIPTION

A check is made to ensure the presence of the input parameters (passed via COMMAREA). If these do not exist, then return is made to the calling program.

The type of request(INSTALL|DELETE) is then determined, and a branch taken to the appropriate function routine(see 'FUNCTION' above for details).

CHANGE ACTIVITY:

PN= REASON REL YYMMDDD HDXXIII : REMARKS
$D1= I06615 410 950614 HD6NPRW: Shipped URM
$P0= 170 850514 : Created.
$P1= M90474 330 910807 HDBWSH: Prologue fixed.
$P2= M83127 410 930709 HDAFDRB: Correct prologue comments.

IDENTIFICATION DIVISION.
PROGRAM-ID. DPKCS101.

ENVIRONMENT DIVISION.

DATA DIVISION.
WORKING-STORAGE SECTION.

* CODES SUPPLIED BY COMMAREA:

   77 install-code PIC X(1) VALUE IS '0'.
   77 delete-code PIC X(1) VALUE IS '1'.
   77 install-sterm PIC X(1) VALUE IS '7'.
   77 install-srse PIC X(1) VALUE IS '8'.
   77 delete-sterm PIC X(1) VALUE IS x'FA'.
   77 delete-srse PIC X(1) VALUE IS x'FB'.

* RETURN CODES:

   77 return-ok PIC X(1) VALUE IS LOW-VALUES.
77 reject PIC X(1) VALUE IS x'Ø1'.

Ø1 WM-WRITEQ-MESSAGE1.
    Ø5 FILLER PIC X(15) VALUE '***** TERMINAL '.
    Ø5 WM-TERMINAL PIC X(4) VALUE SPACES.
    Ø5 FILLER PIC X(22) VALUE ' HAS NO ENTRY IN TABLE'.
    Ø5 FILLER PIC X(13) VALUE ' FOR DPKCS1Ø1'.

Ø1 WM-WRITEQ-MESSAGE2.
    Ø5 FILLER PIC X(16) VALUE '***** CNTL UNIT '.
    Ø5 WM-CNTLUNIT PIC X(2) VALUE SPACES.
    Ø5 FILLER PIC X(22) VALUE ' HAS NO ENTRY IN TABLE'.
    Ø5 FILLER PIC X(13) VALUE ' FOR DPKCS1Ø1'.

* STRUCTURE TO ALLOW THE LAST FOUR CHARACTERS TO BE USED AS
* THE NETNAME.

Ø1 net-sub1 pic s9(2) comp value 8.
Ø1 net-sub2 pic s9(2) comp value Ø.
Ø1 netname-bits.
    Ø2 net-chr pic x(1) occurs 8.

* TERMINAL IDENTIFIER IS BUILT HERE BEFORE BEING PLACED IN THE
* RETURN FIELD.

Ø1 term-idnt.
    Ø2 term-chr pic x(1) occurs 4.

Ø1 TI-TERM-IDNT-ALT REDEFINES term-idnt.
    Ø5 TI-TERM-CHR1-CHR2 PIC X(2).
    Ø5 FILLER PIC X(2).

Ø1 DT-DUMMY-TERM.
    Ø5 DT-CNTL-UNIT-ID PIC X(2) VALUE SPACES.
    Ø5 FILLER PIC X(2) VALUE 'XX'.

Ø1 DE-DB2-ERROR-MSG.
    Ø5 FILLER PIC X(34) VALUE
        'DB2 ERROR - DPKCS1Ø1 - SQLCODE = ('.
    Ø5 DE-SQLCODE PIC Z(8)9- VALUE ZERO.
    Ø5 FILLER PIC X VALUE ')'.
    Ø5 FILLER PIC X(8) VALUE
        'ERRMC = '.
    Ø5 DE-SQLERRMC PIC X(72) VALUE SPACES.

******************************************************************
*    STANDARD SQLCA2 COPY MEMBER
******************************************************************

COPY SQLCA2.
EXEC SQL
   INCLUDE TTRMØØØ
END-EXEC.
EXEC SQL
   INCLUDE SQLCA
END-EXEC.
linkage section.
Ø1 dfhcommarea.
copy dfhtcuds.
Ø1  sterm-idnt.
   Ø2  sterm-chr       pic x(1) occurs 4.
*
* The IBM supplied structure for MODELNAME-LIST is for a single
* modelname. If you need to select the 2nd or subsequent
* modelname you can use a structure similar to the following:
*
* Ø1  modelname-list.
*     Ø2  modelname-count           PIC X(2).
*     Ø2  modelname-names           PIC X(8) occurs 1 to 999
*                                          depending on modelname-count.
*
PROCEDURE DIVISION.
*
* CHECK THAT WE HAVE A COMMAREA, IF NOT THEN EXIT
*
    if eibcalen not equal Ø
*
* EXECUTE THE APPROPRIATE PARAGRAPh FOR INSTALL OR DELETE:
*
    if install-exit-function equal install-code then
       perform install-paragraph
    end-if
*
* IF THE REQUEST WAS AN INSTALL REQUEST THEN THE NEXT TEST
* WILL FAIL ANYWAY, IE FANCY LOGIC NOT REQUIRED!
*
    if delete-exit-function equal delete-code then
       perform delete-paragraph
    end-if
*
    if install-shipped-exit-function equal install-sterm then
       perform install-shipped-paragraph
    end-if
*
    if install-shipped-exit-function equal install-srse then
       perform install-shipped-paragraph
    end-if
*
    if delete-exit-function equal delete-sterm then
       perform delete-paragraph
    end-if
*
    if delete-exit-function equal delete-srse then
       perform delete-paragraph
    end-if
*
* RETURN TO CICS.
*
end-if.
return-line.
exec cics return end-exec.
goback.
*
*
install-paragraph.
*
* SET UP ADDRESSABILITY TO THE COMMAREA.
*
  set address of netname-field to install-netname-ptr.
*
  set address of modelname-list to install-modelname-ptr.
*
  set address of selected-parms to install-selected-ptr.
*
* CHECK IF WE HAVE MODELS TO USE, IF NOT THEN EXIT.
*
  if modelname-count not equal Ø
*
* MOVE THE NETNAME SO THAT IT CAN BE DEALT WITH ON A CHARACTER TO
* CHARACTER BASIS.
*
  move netname to netname-bits
*
* RESET NETNAME LENGTH IF THERE ARE TRAILING SPACES.
*
  perform with test before
    varying net-sub1 from netname-length by -1
    until (net-chr(net-sub1) not = space)
    or (net-sub1 = 4)
  end-perform
*
  subtract 3 from net-sub1
*
  perform with test after
    varying net-sub2 from 1 by 1
    until net-sub2 = 4
    move net-chr(net-sub1) to term-chr(net-sub2)
    add 1 to net-sub1
  end-perform
*
* PLACE TERM-IDNT INTO SELECTED Parameterd
*
  move term-idnt to selected-term-id
    TTRM000-TERMINAL-NMBR
*
* GET PRINTER INFO FROM TABLE
*
  EXEC CICS HANDLE ABEND END-EXEC
EXEC SQL
SELECT TERMINAL_NMBR,
       PRINTER_NMBR
INTO :TTRM000-TERMINAL-NMBR,
       :TTRM000-PRINTER-NMBR
FROM TTRM000
   WHERE TERMINAL_NMBR = :TTRM000-TERMINAL-NMBR
END-EXEC
MOVE SQLCODE TO DE-SQLCODE
MOVE SQLERRMC TO DE-SQLERRMC
IF SQLCODE = 0
   MOVE TTRM000-PRINTER-NMBR TO selected-printer-id
ELSE
   IF SQLCODE = +100
      MOVE term-idnt TO WM-TERMINAL
      EXEC CICS
      WRITEQ TD
      QUEUE('CSML')
      FROM (WM-WRITEQ-MESSAGE1)
      LENGTH(54)
      END-EXEC
      MOVE TI-TERM-CHR1-CHR2 TO DT-CNTL-UNIT-ID
      MOVE DT-DUMMY-TERM TO TTRM000-TERMINAL-NMBR
      EXEC SQL
      SELECT TERMINAL_NMBR,
             PRINTER_NMBR
      INTO :TTRM000-TERMINAL-NMBR,
             :TTRM000-PRINTER-NMBR
      FROM TTRM000
         WHERE TERMINAL_NMBR = :TTRM000-TERMINAL-NMBR
      END-EXEC
      IF (SQLCODE < 0 OR SQLCODE > +99)
         MOVE TI-TERM-CHR1-CHR2 TO WM-CNTLUNIT
         EXEC CICS
         WRITEQ TD
         QUEUE('CSML')
         FROM (WM-WRITEQ-MESSAGE2)
         LENGTH(53)
         END-EXEC
      ELSE
         MOVE TTRM000-PRINTER-NMBR TO selected-printer-id
      END-ELSE
   ELSE
   ELSE

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
MOVE SQLCODE TO DE-SQLCODE
MOVE SQLERRMC TO DE-SQLERRMC
EXEC CICS
   WRITEQ TD
   QUEUE('CSML')
   FROM (DE-DB2-ERROR-MSG)
   LENGTH(125)
END-EXEC
END-IF
END-IF

* SELECT THE MODEL FROM THE LIST SUPPLIED (THE FIRST MODEL IS
* SELECTED).
*
  if selected-modelName = spaces
      move modelName to selected-modelName
  end-if

* SET RETURN CODE Ø
*
  move return-ok to selected-return-code
*  
  end-if.
*
install-shipped-paragraph.
*
* INSTALL CODE HERE.
*  This sample accepts the selected term-id value. If however
*  a term-id clash has occurred then this value has been
*  selected by the caller module DFHZATS.
*  There is no guarantee that this value will be the same
*  once a restart has occurred.
*  Special consideration MUST be given to how this term-id
*  will be used.
*  This sample will update the selected term-id value to
*  the original incoming value. If a clash has occurred and
*  the definition is not busy then it will be replaced.
*
  set address of install-shipped-selected-parms to
  install-shipped-selected-ptr.
  set address of sterm-idnt to install-shipped-termid-ptr.
  move sterm-idnt to selected-shipped-termid.
  move return-ok to selected-shipped-return-code.
*
delete-paragraph.
*
* DELETE CODE IS PLACED HERE.
*

Bruce Borchardt
Senior Systems Programmer (USA)   © Xephon 1998
Borland has announced Java support for CICS enterprise developers with JBuilder, its visual Java development environment. By using IBM’s CICS Gateway for Java product with JBuilder and JavaBeans, CICS support can be integrated into Java and Web-based applications.

CICS Gateway for Java provides the means for applications to exploit CICS servers, providing integration and interoperability between Java applets and CICS through the use of defined CICS/ECI Java classes.

For further information contact:
Borland International, 100 Borland Way, Scotts Valley, CA 95066-3249.
Tel: (408) 431 1000.
Borland International (UK), 8 Pavilions, Ruscombe Business Park, Twyford, Berks. RG10 9NN.
Tel: (01734) 320022.

Sterling Software has announced additions and enhancements to its Vision:Simulate date simulation tool to allow testing at the program level for CICS, batch (MVS/ESA, OS/390, and VSE), and IMS/DC/TM, without disrupting the normal operation of other programs on the system.

Included in Vision:Simulate is a program date/time analyser for locating date/time routines in batch and CICS load modules. It supports COBOL, PL/I, Assembler, and Natural, and includes an optional add-on for testing DB2 and other applications.

For further information contact:
Sterling Software, 1800 Alexander Bell Drive, Reston, VA 22091, USA.
Tel: (703) 264 8000.
Sterling Software, 1 Longwalk Road, Stockley Park, Uxbridge, Middlesex, UB11 1DB.
Tel: (0181) 867 8000.

Available now for CICS, IBM has announced Version 3.1 of ImagePlus for OS/390, which provides a client/server architecture. The workstation portion of ImagePlus Folder Application Facility (IPFAF) is connected to the host portion via TCP/IP, is available on Windows 95/NT and OS/2 workstations, and supports both synchronous and asynchronous API calls.

IBM has also announced TME 10 Performance Reporter for OS/390. Performance features apply to CICS; system; network; IMS; workstations; and AS/400; and there are two OS/2-based features that help with reporting and resource management.

For further information contact your local IBM representative.