March 1998

In this issue

3 CICS date simulator for year 2000 testing
15 CICS Transaction Server for OS/390 Version 1.2
23 Using the LINK/XCTL commands
28 Setting the VSE return code – part 3
39 Converting macros to define statements – part 2
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 £165.00 in the UK; $250.00 in the USA and Canada; £171.00 in Europe; £177.00 in Australasia and Japan; and £175.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1994 issue, are available separately to subscribers for £14.50 ($21.50) each including postage.

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

© 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.
CICS date simulator for year 2000 testing

Here is a CICS date simulator that allows year 2000 testing to be conducted without changing the underlying MVS system date. It can be turned on and off by issuing a transaction and the date can be set to any day from 01/01/1900 to 31/12/2099. A nice feature is the ability to move backwards and forwards by one day at a time simply by hitting PF7 and PF8. It also displays all available CICS date formats.

Optionally, the simulator can be activated in the PLTPI to allow a full CICS cycle to run with a consistent date. This is driven by using the CICS SIT parameter or override INITPARM. Both modes of control can be used together. It might make sense to limit access to the control transaction – or developers may get into an interesting conflict!

This simulator was developed to run under MVS/ESA with CICS Version 4.1, which is the minimum CICS level for year 2000 compliance. The author also has a CICSV ersion 2.1.2 date simulator which allows testing in that environment, although the validity of testing year 2000 changes in a non-compliant CICS is suspect.

The simulator works by changing the value of EIBDATE or the receiving field for ASKTIME/ABSTIME requests. It does this by intercepting all command level calls in the command level call exit XEIOUXT. It does not intercept dates that are not obtained from CICS, for example by use of direct calls from applications to the operating system.

EXEC INTERFACE BLOCK

The EXEC interface block passed to application programs contains the field EIBDATE, which is a 4-byte packed field containing 0cyyddd, where:

- ‘c’ is zero for years up to 2000, and 1 thereafter.
- ‘yy’ is the two-digit year.
- ‘ddd’ is the day within the year.
CICS DATE FORMATS

Applications can request the current date and time from CICS by issuing the EXEC command:

EXEC CICS ASKTIME ABSTIME(data-field)

ABSTIME is optional and, if not specified, CICS merely refreshes the EIBTIME and EIBDATE fields.

If ABSTIME is specified, CICS will place an absolute time in the data-field. This is an 8-byte packed field. ABSTIME is the number of milliseconds since the beginning of 1 January 1900.

So that applications can convert ABSTIME to a meaningful or displayable format, there is another EXEC command – EXEC CICS FORMATTIME ABSTIME(data-field).

Up to CICS Version 4.1.0 there were no four-digit year formats available. From CICS Version 4.1.0, in addition to the previously supported formats, the following new formats are available:

- YYYYDDD
- YYYYMMDD
- YYYYDDMM
- DDMMYYYY
- MMDDYYYY.

The year, month, and day can be separated by a slash (/) in any of these formats by including the optional parameter DATESEP in the FORMATTIME call. The slash can be replaced by any other character by including the required character in the DATESEP parameter, eg DATESEP(‘.’).

The old two-digit formats are, of course, still supported. For example, DDMMYY will return 01/01/00 on 1 January 2000.

DATE SIMULATOR INSTALLATION

- Assemble/link the supplied programs (some are optional) and place them in a library in the DFHRPL concatenation.
• Resource definitions:
  – Define the following programs:
    SY2000 Assembler EXECKEY CICS (main on-line program).
    MSY2000 map (map for SY2000).
    SYX2000 Assembler EXECKEY CICS (XEIOUT exit).
  – Optionally define the following programs:
    SY2FE Assembler (transaction front end).
    SY2FETAB Assembler (front end table).
    SY2PLT Assembler EXECKEY CICS (PLTPI program).
  – Define the following transaction:
    HDAT program SY2000.

Figure 1 shows an HDAT transaction sample display. HDAT is used to set or alter the date within CICS and to turn off the date simulator. It can be used without an entry in the PLTPI (see below):

• PLTPI table (optional):

If you wish to set the system date as CICS comes up, add an entry to your PLTPI table and an override to your CICS run-deck (or SIT parameter). You can turn off the date simulator later by use of the HDAT transaction.

The required PLTPI entry is SY2PLT, which should be placed before any application-related program entry that you wish to use the altered date. SY2PLT can reside in the PLTPI without invoking the date simulator – this is controlled by the override below.

• CICS start-up (optional):

Add an override or SIT parameter, INITPARM=(SY2PLT='dd/mm/yyyy') where ‘dd/mm/yyyy’ is the date you would like to come up with.

An invalid date will cause an error message to be displayed.
DATE SIMULATION
Enter DD/MM/YYYY or STOP —> 01/01/2000 Welcome to year 2000! A leap year.
Today is a SATURDAY
Press PF7 to go one day backwards, PF8 to go one day forwards.
The following formats are available from the FORMATTIME EXEC Command:

<table>
<thead>
<tr>
<th>Format</th>
<th>Example</th>
<th>Format</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>YYDDD</td>
<td>00001</td>
<td>YYYYDDD</td>
<td>20000001</td>
</tr>
<tr>
<td>YYMMDD</td>
<td>000101</td>
<td>YYYYMDM</td>
<td>20000101</td>
</tr>
<tr>
<td>YYDDMM</td>
<td>000101</td>
<td>YYYYDDM</td>
<td>20000101</td>
</tr>
<tr>
<td>DDMYYM</td>
<td>010100</td>
<td>DDMYYM</td>
<td>01012000</td>
</tr>
<tr>
<td>MMDDYY</td>
<td>010100</td>
<td>MMDDYY</td>
<td>01012000</td>
</tr>
</tbody>
</table>

YYYY formats are only available at CICS Version 4.1.0 or higher.
This date simulator will not alter any dates that are not obtained from CICS.
This includes dates obtained using MVS, SVCs, or illegal COBOL verbs under CICS.
TIME: 08:53:05  DATE: 01/01/2000  APPLID: CLCBDLA

Figure 1: HDAT transaction sample display

If the parameter is missing, a ‘Simulator inactive’ message will be displayed.

- Application front-end (optional):
  There is a brief window from the beginning of a task when EIBDATE will contain the unmodified date. This is set to the modified value as soon as there is an EXEC call. A front end has been supplied to close this window, which will ensure that the EIBDATE is correct before the application is given control. This requires transaction definitions to be changed to point to program SY2FE. In addition, SY2FE loads a table, SY2FETAB, which should contain a list of transactions and their matching programs.

MESSAGES AND CODES
The following messages and codes are issued within the HDAT transaction:
- ‘Welcome to the simulator’ – issued on entry to the HDAT transaction.
• ‘DD not valid’ – DD must be in the range 01 to 28, 29, 30, or 31, depending on month/year.

• ‘MM not 01 to 12’ – the month is numeric, but out of the accepted range.

• ‘Not DD/MM/YYYY or STOP’ – the data entered is not in a recognizable format, ie not numeric, ‘/’, or STOP.

• ‘Simulator turned off’ – issued in response to a STOP command.

• ‘Simulator not active’ – the STOP command was issued when the simulator had not started.

• ‘Year not 1900 to 2099’ – 1900 to 2099 is the range of years supported by the simulator.

• ‘One day backwards’ – PF7 issued to move back a day.

• ‘No 19th century support’ – PF7 issued from 01/01/1900.

• ‘One day forwards’ – PF8 issued to move a day forwards.

• ‘No 22nd century support’ – PF8 issued from 31/12/2099.

• ‘Welcome to year yyyy!’ – DD/MM/YYYY date simulation started successfully.

• ‘A leap year.’ – appended to previous message when YYYY is a leap year.

The following messages are issued within application transactions:

• ‘xxxx is not defined to SY2FETAB. Please contact support’ – where ‘xxxx’ is the active transaction. HDAT006E is also written to the console (see below).

The following messages are issued on the console:

• ‘HDAT001I Date simulation started for dd/mm/yyyy’ – issued by programs SY2000 and SY2PLT.

• ‘DAT002I Date simulation terminated’ – issued by program SY2000.

• ‘HDAT003I Date simulation SY2PLT invocation’ – issued by
SY2PLT.

- ‘HDAT004E SY2PLT Initparm error...followed by text...’ – issued by SY2PLT, where the text can be:
  - ‘DD not valid’ – DD must be in range 01 to 28, 29, 30, or 31, depending on month/year.
  - ‘MM not 01 to 12’ – the month is numeric but out of the accepted range.
  - ‘Not DD/MM/YYYY’ – parameter present but not of correct format.
  - ‘Year not 1900 to 2099’ – 1900 to 2099 is the range of years supported.
- ‘HDAT005I Date simulation not active’ – issued by SY2PLT.
- ‘HDAT006E xxxx not defined to SY2FETAB’ – issued by SY2FE, where ‘xxxx’ is a transaction name. This means that transaction ‘xxxx’ points to program SY2FE, but there is no matching transaction in the table SY2FETAB.

PROGRAM SY2000

```assembly
SY2ØØØ RMODE ANY
*                        *
* PROGRAM : SY2ØØØ       *
* DESCRIPTION : THIS MODULE CHANGES THE DATE SEEN BY APPLICATIONS *
* BY ENABLING OR DISABLING EXIT SYX2ØØØ.  *
*                        *
R2 EQU 2 Used by TRT instruction
EIBREG EQU 3 EIB REG
DATAREG EQU 4 DATA REG
BASE1 EQU 5 Base register
R6 EQU 6 Exit global area pointer
R7 EQU 7 Work register
COPY DFHAID
DFHEISTG DSECT
ATIME DS PL8 Absolute time
DATE DS CL1Ø
GMTIME DS CL8
YEAR DS CL1Ø
DAYNUM DS F
DAYCNT DS F
HEXDATE DS F
COMDATE DS PL1Ø
```

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
DIVDATE DS PL9
DAYCNTP DS D
DAYCNTQ DS PL10
YRDIFF DS PL2
DYDIFF DS PL2
MMWK DS PL2
DDWK DS PL2
MESSIND DS X
MESSØ DS CL48
LEAPIND DS X
XSTATUS DS F
DLENG DS H
COMMAS DS ÙH Commarea start
TSTAT DS X
COMMAE EQU * Commarea end
COMMAL EQU COMMAE-COMMAS Commarea length
COPY MSY2ØØØ
EISTGEND EQU *
SY2ØØØ DFHEIENT CODEREG=(BASE1), X
EIBREG=(EIBREG), X
DATAREG=(DATAREG)
BEGIN
DC CL12'PROGRAM ID: '
DC CLB'SY2ØØØ'
DC CL4': '
DC CL24'ASSEMBLY TIME AND DATE:'
DC CLB'&SYSTIME'
DC CLB'&SYSDATE'
BEGIN
D ÙH
MVC COMDERRO(24),MSGØØ Welcome message
MVI MESSIND,X'00'
CLC EIBCALEN,=H'0'
BE SENDMAP First time through
EXEC CICS HANDLE AID PF3(RETURN1) CLEAR(RETURN1)
EXEC CICS RECEIVE MAP('MSY2ØØØ')
*
* Input field validation section
*
MVC YEAR,INCOMDI Save input year
CLC YEAR(4),=C'STOP' Turn off simulator
BE TURNOFF
TRT YEAR(2),TRANTAB1 Numeric DD?
BNZ INERR No
CLI YEAR+2,C'/'
BNE INERR
TRT YEAR+3(2),TRANTAB1 Numeric MM?
BNZ INERR No
CLI YEAR+5,C'/'
BNE INERR
TRT YEAR+6(4),TRANTAB1 Numeric YYYY?
BNZ INERR No
PACK DDWK,YEAR(2)
CP DDWK,=PL1'0'
BE DDERR
ZAP DYDIFF,DDWK
PACK MMWK,YEAR+3(2)
LA R7,MONTAB

MONLOOP DS ØH
CLI Ø(R7),X'FF'
BE MMERR
CP MMWK,Ø(2,R7)
BE MONMATCH
LA R7.6(R7)

B MONLOOP

MONMATCH DS ØH
CP DDWK,2(2,R7)
BH DDERR
AP DYDIFF,4(2,R7)

VALIDMM

DDERR DS ØH
MVC COMDERRO(24),MSGØ1
B SENDMAP

MMERR DS ØH
MVC COMDERRO(24),MSGØ2
B SENDMAP

INERR DS ØH
MVC COMDERRO(24),MSGØ3
B SENDMAP

TURNOFF DS ØH
EXEC CICS HANDLE CONDITION PGMIDERR(NOTURNOF)
EXEC CICS INQUIRE EXITPROGRAM('SYX2ØØØ')
STARTSTATUS(XSTATUS)

CLC XSTATUS,DFHVALUE(STARTED)
BNE NOTURNOF
EXEC CICS DISABLE EXITALL PROGRAM('SYX2ØØØ')
MVC COMDERRO(24),MSGØ4
MVC INCOMDO,=CL10'STOP
EXEC CICS WRITE OPERATOR TEXT(MESS2)
B SENDMAP

NOTURNOF DS ØH
MVC COMDERRO(24),MSGØ5
B SENDMAP

VALIDMM DS ØH
ZAP HEXDATE,DYDIFF
CLC YEAR+6(2),=C'19'
BNE VYRØ1

MVI HEXDATE,X'ØØ'
ZAP YRDIFF,=P'0'
B VYRØ2

VYRØ1 DS ØH
CLC YEAR+6(2),=C'20'
BNE NRANGE
MVI HEXDATE,X'Ø1'
ZAP YRDIFF,=P'100'
B       VYRØ2
NRANGE  DS   ØH
MVC     COMDERRO(24),MSGØ6
B       SENDMAP
VYRØ2  DS   ØH
MVC     HEXDATE+1(1),YEAR+9
MVO     HEXDATE+1(1),YEAR+8(1)
MVO     YRDIFF+1(1),YEAR+9(1)
MVN     YRDIFF(1),YEAR+8
ZAP     COMDATE,YRDIFF
ZAP     DIVDATE,YRDIFF  For leap day calculation
MP      COMDATE,=PL4'31536Ø'  ABSTIME year difference
MP      COMDATE,=PL2'100'    Prevent spec exception
MP      COMDATE,=PL3'1000'  Prevent spec exception
MVI     LEAPIND,X'00'     Leap year indicator
DP      DIVDATE,=PL1'4'    Divide year diff by 4
CP      DIVDATE+8(1),=PL1'Ø'  Remainder zero - so leap year
BNZ     NOTLEAP
CP      DIVDATE(8),=PL1'0'  1900 was not a leap year
BZ     NOTLEAP
MVI     LEAPIND,X'FF'    Set leap year indicator
AP      HEXDATE,=PL1'1'    Increase EIBDATE by one
CLC     YEAR+3(2),=C'02'  After February?
BH      NOTLEAP  Include this years leap day
SP      HEXDATE,=PL1'1'    Decrease EIBDATE by one
SP      DIVDATE(8),=PL1'1'  Else take a day off
NOTLEAP DS   ØH
AP      DIVDATE(8),DYDIFF  Add in DD/MM contribution
CLI     EIBAID,DFHPF7
BNE     NOTBWD
SP      HEXDATE,=PL1'1'    Decrease EIBDATE by one
CP      HEXDATE+2(2),=PL1'Ø'  Days now zero?
BNE     BWDSCK2
SP      HEXDATE,=PL3'1000'  Reduce year by one
CP      HEXDATE,=PL1'Ø'    Year now zero?
BM      OFFSTRT  No pre-1900 support
BE     BWDSCKØ   1900 not a leap year
CP      DIVDATE+8(1),=PL1'1'  Was this year a leap year?
BE     BWDSCK1
BWDSCKØ DS   ØH
ZAP     HEXDATE+2(2),=PL2'365'  No - set day to 365
B       BWDSCK2
OFFSTRT DS   ØH
MVC     COMDERRO(24),MSGØ7
B       SENDMAP
BWDSCK1 DS   ØH
ZAP     HEXDATE+2(2),=PL2'366'  Set day to 366
BWDSCK2 DS   ØH
SP      DIVDATE(8),=PL1'1'   Go into past
MVC     COMDERRO(24),MSGØ8
B       NOTFWD
NOTBWD  DS   ØH
CLI     EIBAID,DFHPF8
BNE    NOTFWD      Go into future
AP     HEXDATE,=PL1'1'    Increase EIBDATE by one
CP     HEXDATE+2(2),=PL2'366'
BL     FWDSOK
CP     HEXDATE,=PL4'199366'
BNE    FWDSCK1
MVC    COMDERRO(24),MSG09
B      SENDMAP
FWDSCK1 DS ØH
CP     HEXDATE+2(2),=PL2'366'
BH     YRUP
CLI    LEAPIND,X'FF'    Leap year?
BE    FWDSOK
YRUP    DS ØH
AP    HEXDATE,=PL3'1000'    Add one to year
ZAP   HEXDATE+2(2),=PL1'1'    Set day to one
FWDSOK DS ØH
MVC    COMDERRO(24),MSG10
B    NOTTOD
NOTFWD DS ØH
SP    DIVDATE(8),=PL1'1'    Take today off!
NOTTOD DS ØH
MP    DIVDATE(8),=PL3'86400'
MP    DIVDATE(8),=PL3'1000'    Tot days in milliseconds
AP    COMDATE,DIVDATE(8)    + leap day difference
CLI    LEAPIND,X'00'
BNE    BYPCHK
CLC    YEAR+3(2),=C'02'    February?
BNE    BYPCHK
CP    DDWK,=PL2'29'    DD = 29 but not a leap year
BE    DDERR
BYPCHK DS ØH
EXEC CICS HANDLE CONDITION PGMIDERR(EXITSTRT)
EXEC CICS INQUIRE EXITPROGRAM('SYX2000')
      * STARTSTATUS(XSTATUS)
      CLC    XSTATUS,DFHVALUE(STARTED)
      BNE    EXITSTRT
EXEC CICS DISABLE EXITALL PROGRAM('SYX2000')
EXITSTRT DS ØH
EXEC CICS ASKTIME ABSTIME(ATIME)
EXEC CICS FORMATTIME ABSTIME(ATIME) DAYCOUNT(DAYCNT)
      L       R7,DAYCNT
      CVD    R7,DAYCNETP
      ZAP   DAYCNETQ,DAYCNETP
      SP    DAYCNETQ,=PL1'1'    Last night not tonight
      MP    DAYCNETQ,=PL3'86400'
      MP    DAYCNETQ,=PL3'1000'    Absolute time last midnight
      SP    COMDATE,DAYCNETQ
EXEC CICS ENABLE EXIT('XEIOUT') PROGRAM('SYX2000')
      * GALENGTH(12)
EXEC CICS EXTRACT EXIT PROGRAM('SYX2000') GASET(R6)  *
GALENGTH(DLENG)
MVC Ø(4,R6),HEXDATE
OI 3(R6),X'OF'
MVC 4(B,R6),COMDATE+2
EXEC CICS ENABLE PROGRAM('SYX2000') START
MVI MESSIND,X'FF'
CLI EIBAID,DFHPF7
BE SENDMAP
CLI EIBAID,DFHPF8
BE SENDMAP
MVC COMERRO(24),MSG11
MVC COMERRO+16(4),YEAR+6
CLI LEAPIND,X'00'
BE SENDMAP Not a leap year
MVC LEAPO(12),MSG12
SENDMAP DS ØH
EXEC CICS ASKTIME ABSTIME(ATIME)
EXEC CICS FORMATTIME ABSTIME(ATIME) DDMYYYY(DATE)
* TIME(GMTIME) DATESEP TIMESEP
EXEC CICS FORMATTIME ABSTIME(ATIME)
* YYDDD(VAR0) YYYYDDD(VAR10)
* YYMMDD(VAR20) YYYYMMDD(VAR30)
* YYDDMM(VAR40) YYYYDDMM(VAR50)
* DDMYYY(VAR60) DDMYYYY(VAR70)
* MMDDYY(VAR80) MMDDYYYY(VAR90) DAYOFWEEK(DAYNUM)
CLI MESSIND,X'FF'
BNE NOWTO
MVC MESS0,MES1
MVC MESS0+37(10),DATE
EXEC CICS WRITE OPERATOR TEXT(MES0)
NOWTO DS ØH
MVC DATEO,DATE
MVC TIMEO,GMTIME
CLI EIBAID,DFHPF7
BNE ORPF8
MVC INCOMDO,DATE
B NOPF8
ORPF8 DS ØH
CLI EIBAID,DFHPF8
BNE NOPF8
MVC INCOMDO,DATE
NOPF8 DS ØH
LA R7,DAYTAB
DAYLOOP DS ØH
CLC DAYNUM+3(1),Ø(R7)
BE DAYFND
LA R7,10(R7)
B DAYLOOP
DAYFND DS ØH
MVC TODAYO(9),1(R7)
EXEC CICS ASSIGN APPLID(APPLO)
EXEC CICS SEND MAP ('MSY2000') ERASE FREEKB
* * RETURN BUT COME BACK *
  
  RETURNØ DS ØH
  EXEC CICS RETURN TRANSID('HDAT')
  * COMMAREA(COMMAS) LENGTH(COMMAL)
  *
  * RETURN AND FINISH *
  *
  RETURN1 DS ØH
  EXEC CICS SEND CONTROL ERASE FREEKB
  EXEC CICS RETURN
  *
  * CONSTANTS *
  *
  TRANTAB1 DS ØF Ø 1 2 3 4 5 6 7 8 9 A B C D E F
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  DC X'FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF'
  *
  DAYTAB DC XL1'Ø0',CL9'SUNDAY'
  DC XL1'Ø1',CL9'MONDAY'
  DC XL1'Ø2',CL9'TUESDAY'
  DC XL1'Ø3',CL9'WEDNESDAY'
  DC XL1'Ø4',CL9'THURSDAY'
  DC XL1'Ø5',CL9'FRIDAY'
  DC XL1'Ø6',CL9'SATURDAY'
  
  MONTAB DC PL2'Ø1',PL2'31',PL2'Ø00'
  DC PL2'Ø2',PL2'29',PL2'Ø31'
  DC PL2'Ø3',PL2'31',PL2'Ø59'
  DC PL2'Ø4',PL2'30',PL2'Ø90'
  DC PL2'Ø5',PL2'31',PL2'120'
  DC PL2'Ø6',PL2'30',PL2'151'

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

Mick Masters  
Principal Consultant  
Cap Gemini (UK)  

© Xephon 1998
IBM recently made available CICS Transaction Server for OS/390 Version 1 Release 2. This integrated software package is the company’s premier offering for the deployment of high-volume transaction processing applications.

CICS Transaction Server is one of the new software servers that share OS/390 as a foundation. They all have the same objectives as OS/390 – to reduce the total cost of computing; to reduce installation and test time; integration with other OS/390 servers; and full interoperability with remote platforms.

CICS Transaction Server does not just contain the latest release of the CICS/ESA product, it is an integrated package of related products that collectively improve the management of your transactional environment and offer an extensive range of services for workstation and Internet access. The package is orderable with one program number and delivered as one product with one price.

The focus for Version 1 Release 2 has been easing the migration from previous CICS releases and extending Transaction Server’s capabilities as an application server in the Internet-based e-business environment.

MIGRATION FROM PREVIOUS RELEASES
The previous release of Transaction Server (Version 1 Release 1) exploited the MVS logger for CICS logging. This implicitly called for a Coupling Facility, which is required by the MVS logger. In CICS/ESA Version 4.1, the release prior to Transaction Server Version 1 Release 1, CICS log streams were written directly to disk.

The Coupling Facility/parallel sysplex configuration is hugely successful, but IBM has recognized that its benefits are limited for some users, usually those running a single MVS image or stand-alone OS/390 system.

To address this, Release 2 has introduced the ability to log either directly to disk or through the MVS logger. This flexibility for
INTERNET ENHANCEMENTS
Release 2 introduces a number of significant enhancements that position CICS Transaction Server as the application engine for commercial Internet solutions known as e-business applications. There are three ways of accessing CICS Transaction Server on OS/390 from a Web environment:

- The CICS Java gateway for OS/390.
- The CICS Web Interface (CWI).
- Direct Web access to the CICS address space.

The CICS Java gateway for OS/390
For sometime now, the CICS client products have packaged a Java application, the Java gateway, that enables Web browsers or network computers to interact with new CICS applications (via the External Call Interface, or ECI) or existing 3270 CICS transactions (via the External Presentation Interface, or EPI). The Java gateway, driven by a set of supplied Java classes, receives Web-based requests and issues ECI/EPI calls to the CICS server. The response from CICS is converted back into a Java structure and returned to the desktop. The gateway currently runs on OS/2, Windows NT, Sun Solaris, and AIX. CICS Transaction Server Release 2 now includes a Java gateway implemented directly on OS/390, thanks to the existence of the MVS Web server, called the Internet Connection Secure Server (ICSS), and Java for OS/390.

This new gateway means you can drive CICS programs/transactions from the Web, by going through a Web server and gateway program on a distributed platform (in a three-tier environment), or directly through the Web server and gateway program on MVS (a two-tier environment). Communication from the OS/390 CICS Java gateway application to CICS is through the EXCI. All of the well-known Java benefits are available. The code is downloaded to the client from the
server when it is needed, so there are no problems of version control. Java allows a full range of GUI techniques, and applications can run unchanged on almost any client, independently of the server.

The CICS Web Interface (CWI)

The CWI is another two-tier option for connecting Web interfaces to CICS Transaction Server. An ICAPI DLL is provided for the ICSS in the CICS Transaction Server package. The DLL interfaces with CICS, again using the EXCI. The interface provides a route into CICS from browsers that don’t support Java. It uses the ICSS to provide support for secure Web protocols, allowing encryption of data across the network.

The CWI was available in the previous release of Transaction Server, with support limited to executing new CICS programs (a CICS LINK with COMMAREA interface). Now, both COMMAREA-based programs and existing 3270-based CICS transactions can be driven. Going through a Web server means you need ‘Web master’ skills to manage the environment, but it does have the advantage of allowing you to manage all Web content from one place.

Direct Web access to the CICS address space

CICS Transaction Server includes a built-in HTTP listener. This means it can receive requests from the Web directly, without needing to go via an intermediate Web server. This direct TCP/IP connection into CICS provides a high-performance CICS connection to a browser.

Which way from Internet to CICS Transaction Server?

The three interfaces described are all driven from a common architected URL structure. This directs the browser request to the appropriate interface, tells it whether the executable code will be a CICS 3270 transaction or a linkable CICS program, and provides details about the target CICS user program or transaction name.

To facilitate direct access to CICS from ‘non-3270’ environments the concept of ‘bridging’ was born. This allows you to create a virtual 3270 model terminal, against which your terminal-based 3270 CICS
application runs – thinking it is a real CICS terminal. When CICS performs terminal I/O, the data is intercepted by a CICS exit known as the ‘bridge’. The bridge gains addressability to the CICS terminal control command, for example EXEC CICS SEND MAP. It can address and manipulate the datastream and is exposed to the parameters that have been set for the command. Usually, it would strip off the 3270-specific parameters from the datastream and replace them with those of another transport.

Within the Transaction Server package there are two supplied bridges – one to convert the 3270 CICS datastream to an MQSeries message and return it over MQ, and the other to convert the datastream to HTTP for a browser or network computer. You have the ability to write your own bridge exit if you wish to handle other transports. The common URL interface looks like this:

http://machine:port/converter/alias transID/application/trancode/

Where:

- ‘Machine’ is the TCP/IP address of the machine.
- ‘Port’ is the port number of the HTTP listener. This is optional, the default being 80.
- ‘Converter’ asks whether you want to drive a CICS user program to perform datastream manipulation outside the user program that contains the business logic (a more detailed description is below). This should be set to ‘CICS’ if no conversion is to be driven.
- ‘Alias transID’ has the default CWBA. This is the transaction code relating to DFHWBA, the CICS supplied program that receives the requests from the various Web-based sources.
- ‘Application’ is the name of the CICS user program to be called. Communication is carried out through the CICS COMMAREA. If the target program contains 3270 datastream calls (eg SEND/RECEIVE MAP), this parameter should be set to DFHWBTDA. DFHWBTDA is the supplied bridge program that takes care of bridging 3270 CICS transactions.
Figure 1: Option interactions with CICS
• ‘Trancode’ is only relevant if the application parameter has been set to DFHWBTTA. This parameter would then be the user transaction code to be driven.

Figure 1 clarifies how the various options interact with CICS.

If the request comes via the MVS Web server, it will have originated from the Java gateway or the ICAPI interface. As previously mentioned, the Java gateway can also run on a distributed platform, with the latest request coming into CICS through the ECI or EPI interface. Both of these routes result in an EXCI call being issued into the CICS address space, which drives the supplied program DFHWBA. If the request has come in via the direct HTTP listener, the listener will remove datastream headers and then also pass the request into DFHWBA.

Driving a user program with COMMAREA

If the request is to drive a user program (the specified application parameter in the URL is not set to DFHWBTTA), a CICS LINK with COMMAREA is issued to the program. If the converter parameter has not been set to ‘CICS’, Transaction Server assumes you would like to drive a user CICS program which can manipulate the user COMMAREA prior to, and following, the target program (specified in the application parameter field) being executed. Once the program has completed, the resulting COMMAREA is returned to the requester.

Driving an existing 3270 CICS transaction

To drive an existing 3270 CICS transaction, you must first use a new off-line utility which automatically creates HTML templates from existing BMS maps. New macros are also included, which allow you to include HTML in BMS maps (DFHWBOUT), and specify Web characteristics within your definition, such as PF key button names, background GIF for a map, and Javascript options (DFHMDX).

At execution time, you tell the interface that you wish to drive a 3270 CICS transaction by setting the user program name to DFHWBTTA. Again, the manipulation exits are driven if the converter parameter is set. The ‘encode’ entry point of the user conversion program is commonly used with 3270 interfaces to set the initial cursor position...
and DFHWBTTA is then driven. It uses a ‘dummy’ TCTTE (CICS terminal) to run the transaction and receives back the resulting datastream. This bridging interface also manages state, if the transaction is in a pseudo-conversation. The ‘decode’ entry point is then driven. This allows you to optionally manipulate the HTML layout that was automatically generated by the off-line utility. The HTML is then returned to the requester.

OTHER ENHANCEMENTS

The release also features the following functional enhancements:

- Closer coupling between CICS and DB2 – the CICS DB2 adapter has been enhanced to improve performance. There have also been systems management enhancements to allow Resource Definition On-line (RDO) of DB2 resources. This feature provides an alternative to the Resource Control Table (RCT) macro definitions where CICS had to be recycled before new definitions were recognized. Definitions are now dynamic, allowing 7 by 24 operations.
- Further exploitation of the parallel sysplex.

THE CICS TRANSACTION SERVER PACKAGE

Along with the latest CICS/ESA technology, the following products are packaged with Release 2:

- CICSpLex System Manager (CPSM) 1.3 which provides a single interface for managing your multi-region CICS environment.

Features include a view of multiple CICS regions that allows definition or installation of CICS resources across multiple CICS occurrences from a single point; CPSM is built on the workload management services available with MVS to manage CICS systems to your service class goals. Workload is balanced across AORs and, in the event of a failing region, CICS redistributes to ensure that availability and throughput are maximized. CPSM’s user interface can be either 3270 or GUI based.
The CICS clients provide lightweight access from DOS, Windows, OS/2, Apple Macintosh, Sun Solaris, or AIX, into your CICS server. The client package includes the CICS Java gateways for distributed platforms and an interface to allow integration and easy exchange of data between Lotus Notes and CICS. They also include TCP62, allowing direct TCP/IP or LU6.2 connectivity from the client desktop into CICS.

- Transaction Server for WARP 4.0 – the OS/2 version of the CICS Server product.
- REXX for CICS, which allows development and runtime support for REXX CICS transactions.
- Remote Procedure Call (RPC) interfaces, which enables DCE or ONC RPC calls to drive CICS programs.
- Transaction affinities utility, which allows you to analyse your CICS programs and transaction for the existence of affinities that could affect their deployment on parallel sysplex configurations.

**KEY PREREQUISITES**

CICS Transaction Server Version 1 Release 2 requires OS/390 or MVS/ESA SP Version 5.2 or later, and either OS/390 Version 2 Release 4 DASD-only logging for single-system sysplexes or a Coupling Facility for parallel sysplex.

Rob K Lamb  
Manager, Client Services, Hursley Services & Technology  
IBM UK Laboratories Ltd (UK)  
© Xephon 1998

---

Why not share your expertise and earn money at the same time? CICS Update is looking for JCL, macros, program code, etc, that experienced CICS users have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2. Why not call now for a free copy of our Notes for contributors?
Using the LINK/XCTL commands

This article, the third in the series, continues to examine 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 the use of the LINK/XCTL commands with the INPUTMSG option.

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.

LINK/XCTL INPUTMSG

Many applications are menu-based. This is an obvious boon to users, who do not need to know cryptic codes for the functions they wish to perform. On the other hand, a complex system may have many levels of menu, which can lead to frustration for a user quite familiar with the functions of the application and who uses the same ones consistently.

Some organizations have gone further and incorporated disparate applications into a single menu system controlled by some ‘middleware’. This approach was frequently adopted early on in the evolution of IT development in the organization, meaning that the front-end program was originally written at the macro level. The technique used was to address the Terminal Input Output Area (TIOA) to examine the input, to determine which application program should handle that request. The menu program would then LINK or XCTL to the target program, which would then use the RECEIVE or RECEIVE MAP() command of the command level API to place the input data INTO a program-provided area.

When this type of macro level menu handler was converted to command level, there was no means of examining the TIOA without causing CICS to free it. So IBM added the INPUTMSG option to the
LINK and XCTL commands to overcome this limitation.

I have written an application program which has no purpose other than to illustrate the use of various LINK and XCTL techniques. However, a couple of comments are in order. Observe that the map and mapset names may need to be changed. The program uses YYYYMAP as the name of both. If you wish to change it, use a global change for that name. Also note that the program has no real purpose other than to illustrate this article.

The first of the techniques is the use of the INVOKINGPROGRAM option of the ASSIGN command. This option allows an application to vary its action according to the originator of the request. Usually it is better to include a function as part of the parameters passed to the program, although this is not always feasible. If the program is to be used as the first in a transaction, as well as a sub-module, or if the application is being adapted to allow interfaces to newer applications whilst keeping the changes to the existing programs to a minimum, then this approach may be the easiest to use. WS-WHAT-PROG will contain either the name of the program requesting its services or spaces if it was directly invoked by CICS. The use of INVOKINGPROGRAM can aid in providing an ‘expert’ mode for experienced users of a menu-driven system.

The PROGRAM option is also quite helpful. The reason this is handy is to ensure that an application has as few name dependencies as possible. If some sort of naming convention is established, which allows related programs to vary in only a few set positions (eg the last two characters), then changing the names of the programs and transactions, etc, is easily done. This provides maximum configurability and is particularly important for software intended to be sold to others.

The second technique to observe is the actual use of the XCTL command with the INPUTMSG option. Here the data obtained from the user via a RECEIVE command is passed on to the program which is to process it. In this case it happens to be the same program but invoked again as discussed below.

The third technique I wish to mention is the fact that the program is recursively called as part of the application design. The ability to use
Recursion is often advantageous. The use of this technique makes the coding a bit obscure so I shall ‘walk’ through it:

- Find the name of this program and who invoked it (ASSIGN).
- If it is the first time in, invite the user to input data (IF EIBCALEN = ZERO).
- If it is not the first time in, restore the saved data and arbitrarily keep a count of how many times the program has been invoked.
- If the program was invoked by CICS, get the input data (RECEIVE) and invoke the next program (which happens to be the same as this one in this case) re-instating the TIOA for it to map the input (XCTL).
- If the program was invoked by some other program (which happens to be the same as this one in this case), map the input data (RECEIVE MAP) and simply echo the input back to the user.
- Note that the program insists on having some input.

**PROGRAM SOURCE**

```sparl
IDENTIFICATION DIVISION.
PROGRAM-ID. SAMPLE.
ENVIRONMENT DIVISION.
DATA DIVISION.
WORKING-STORAGE SECTION.
COPY YYYYMAP.
COPY DFHBMSCA.
COPY DFHAID.

01 WS-INPUT PIC X(2000).

01 FILLER.
   03 FATAL-MSG PIC X(24) VALUE 'FATAL ERROR ENCOUNTERED!'.
   03 END-MSG.
   05 FILLER PIC X(24) VALUE 'Transaction terminated:'.
   05 EM-OUT PIC X(30).

01 FILLER.
   03 WS-INPUT-LTH PIC S9(4) COMP.
   03 WS-PROGRAM PIC X(08).
```
PROCEDURE DIVISION.
  EXEC CICS ASSIGN
    PROGRAM(WS-PROGRAM)
    INVOKINGPROG(WS-WHAT-PROG)
  END-EXEC
  IF  EIBCALEN = ZERO
    EXEC CICS SEND
      MAP('YYYYMAP')
      MAPONLY
      ERASE
    END-EXEC
    PERFORM RET-CA
  ELSE
    MOVE DFHCOMMAREA TO WS-COUNT-X
    ADD 1 TO WS-COUNT
  END-IF
  IF  INVOKED-BY-CICS
    MOVE LENGTH OF WS-INPUT TO WS-INPUT-LTH
    EXEC CICS RECEIVE
      INTO(WS-INPUT)
      LENGTH(WS-INPUT-LTH)
      NOHANDLE
  END-EXEC
  IF  EIBRESP NOT = DFHRESP(EOC)
    PERFORM FATAL-ERROR
  END-IF
  EXEC CICS XCTL
    PROGRAM(WS-PROGRAM)
    COMMAREA(WS-COUNT)
    INPUTMSG(WS-INPUT)
    INPUTMSGLEN(WS-INPUT-LTH)
  END-EXEC
  ELSE
    EXEC CICS RECEIVE
      MAP('YYYYMAP')
      NOHANDLE
  END-EXEC
  EVALUATE EIBRESP
    WHEN DFHRESP(NORMAL)
      MOVE INPDATAI TO EM-OUT
    WHEN DFHRESP(MAPFAIL)
      MOVE 'No data entered!' TO MSGO
MOVE DFHBMASB TO MSGA
MOVE -1 TO INPDATAL
EXEC CICS SEND
   MAP('YYYYMAP')
   DATAONLY
   FREEKB
   CURSOR
END-EXEC
PERFORM RET-CA
WHEN OTHER
   PERFORM FATAL-ERROR
END-EVALUATE
EXEC CICS SEND
   FROM(END-MSG)
   ERASE
END-EXEC
PERFORM RET
END-IF
.
RET-CA.
EXEC CICS RETURN
   TRANSID(EIBTRNID)
   COMMAREA(WS-COUNT)
END-EXEC
.
RET.
EXEC CICS RETURN
END-EXEC
.
FATAL-ERROR.
EXEC CICS SEND
   FROM(FATAL-MSG)
   ERASE
END-EXEC
PERFORM RET
.

BMS MACROS

YYYYMAP DFHMSD TYPE=SYSPARM,LANG=COBOL,MODE=INOUT,STORAGE=AUTO,
   TIOAPFX=YES,CTRL=(FREEKB,FRSET)
YYYYMAP DFHMDI SIZE=(24,80)
   DFHMDF POS=(01,19),LENGTH=40,ATTRB=(ASKIP,BRT),
   INITIAL='Sample Use of INPUTMESSAGE and Recursion'
   DFHMDF POS=(02,19),LENGTH=40,ATTRB=(ASKIP,BRT),
   INITIAL='Key arbitrary data:'
INPDATA DFHMDF POS=(03,39),LENGTH=20,ATTRB=(UNPROT,IC)
The next article in this series will continue the theme of using some of the useful but uncommonly used options and features of the API and SPI.

Jerry Ozaniec  
Circle Computer Group (UK)  © Xephon 1998

Setting the VSE return code – part 3

This month we conclude the program to set the VSE return code during CICS start-up and normal shut-down, so that conditional JCL can be used to restart it automatically if the CICS system has terminated abnormally. It also determines whether DTSANALS needs to be run and, if it does, submits a job to perform a RECOVER function.

```
CKMTC9 EQU *  
MVI MTCSW,C'1' INDICATE MATCH.  
BR RB RETURN TO CALLER.  
DPCKJPS DC C'DPCKJP STORAGE HERE. ' INSERT EYE CATCHER.  
NUMPRM DC X'00' NUMBER OF PASSED PARAMETERS.  
ESASW DC C'0' VSE/ESA SWITCH. (0=NO, 1=YES).  
DYNSW DC C'0' RUNNING IN DYNAMIC PARTITION SWITCH.  
MTCSW DC C'0' MATCH SWITCH.  
PTYSW DC C'0' PRTY SWITCH.  
WAISW DC C'0' WAIT SWITCH.  
CKJPMS DC C'JOB/PROGRAM'  
JOBNAME DS CL8 EXECUTING JOB NAME SAVE AREA.  
PGMNAME DS CL8 EXECUTING PROGRAM NAME SAVE AREA.  
PIBLOGID DC C' ' PIBLOGID.  
DS ØD MUST BE DOUBLE WORD ALIGNED.  
CPUID DS ØCL1Ø  
DS PL8  
PART DC X'FFFF'  
PIDS DS ØCL2  
DC C'BGFAFBF1F2F3F4F5F6F7F8F9'```
PIDSD    DC    CL128' '            ROOM FOR 64 DYNAMIC PARTITIONS.
     DC    X'FF'                   END OF TABLE. (DON'T MOVE/REMOVE).
     DC    C'PCBS HERE. '         INSERT EYE CATCHER.
     DS    0F
PCBSTAP DS    0CL4
     DC    48X'00000000'
PCBDYNP DS    0CL4
     DC    256X'00000000'
     DC    X'FFFFFFFF'
OPTN1    DS    C                   OPTION ONE SAVE AREA.
OPTN2    DS    C                   OPTION TWO SAVE AREA.
OPTN3    DS    CL25                OPTION THREE SAVE AREA.
OPTN123S DS    CL27
INPFLD1S DC    CL90' '
INPPIDSS DS    CL8 @@
INPFUNCS DS    C   @@
     DC    C'PARMS HERE. '         INSERT EYE CATCHER.
INPFLD1 DS    0CL45               PARAMETER ONE.
INPFUNC DS    C                   FUNCTION CODE. (C=CHECK, W=WAIT).
INPJOBN DS    CL8                 JOB NAME.
INPPGMN DS    CL8                 PROGRAM NAME.
INPRFLD DS    0CL26               RETURN CODE, OPTION AND PARTITION-ID
INPRCDE DS    C                   RETURN CODE.
INPOPTN DS    C                   OPTION CODE.
INPPIDS DS    CL24                PARTITION-IDS.
INPPCNT DS    CL2                 PARTITION-IDS COUNTER.
     DS    0H                   ALIGN ON HALFWORD BOUNDARY.
INPFLD2 DS    0CL240              PARAMETER TWO.
INPJBPGL DS    CL192              PARTITION JOB/PROGRAM NAME(S).
INPCOMR DC    12F'0'               PARTITION COMMUNICATION REGION ADDRE
     DC    C'PIK HERE. '         INSERT EYE CATCHER.
     DS    0F
PIK      DS    CL36               ROOM FOR TWELVE 3 BYTE ENTRIES.
     DC    X'FF'                   END OF TABLE. (DON'T MOVE/REMOVE).
BLANKS   DC    CL255' '
     DC    C'H/F/D HERE. '        INSERT EYE CATCHER.
     DS    H'0'
COUNT    DC    X'0000002C4'
ASYSCOM DS    F
ACOMRG  DS    F
AIJABFCB DS    F
APCBATAB DC    X'0000002C4'
EPCBATAB DC    X'FFFFFFFF'
AMODE    DS    X
SVR1    DC    F'0'
SVR5    DC    F'0'
SVR5A   DC    F'0'
SVR5B   DC    F'0'
SVR5C   DC    F'0'
SVR5D   DC    F'0'
SVR6A   DC    F'0'
SVR6B DC F'0'
SVR6C DC F'0'
SVR6X DC F'0'
SVREGS DC 4F'0'
SVRA DC F'0'
SVRB DC F'0'
SVRB1 DC F'0'
SVRC1 DC F'0'
SVRC2 DC F'0'
SVRD DC F'0'
SAVEAREA DC 9D'0'
   DC C'SVAPCB  '
SVAPCB DS CL184
   DC C'SVPCBA  '
SVPCBA DS CL100
TIMOUT TECB
   DC C'TAB HERE. '       INSERT EYE CATCHER.
   DS 0F
TAB    DS 0CL25       ROOM FOR TWELVE 25 BYTE ENTRIES.
   DC X'01',C' ',X'0000',CL16',X'00000000' XX,ID,P1K,JO0/
   DC X'02',C' ',X'0000',CL16',X'00000000' PROGRAM NAME &
   DC X'03',C' ',X'0000',CL16',X'00000000' COMRG ADDR.
   DC X'04',C' ',X'0000',CL16',X'00000000'
   DC X'05',C' ',X'0000',CL16',X'00000000'
   DC X'06',C' ',X'0000',CL16',X'00000000'
   DC X'07',C' ',X'0000',CL16',X'00000000'
   DC X'08',C' ',X'0000',CL16',X'00000000'
   DC X'09',C' ',X'0000',CL16',X'00000000'
   DC X'0A',C' ',X'0000',CL16',X'00000000'
   DC X'0B',C' ',X'0000',CL16',X'00000000'
   DC X'0C',C' ',X'0000',CL16',X'00000000'
   DC X'0D',C' ',X'0000',CL16',X'00000000'
   DC X'0E',C' ',X'0000',CL16',X'00000000'
   DC X'0F',C' ',X'0000',CL16',X'00000000'
   DC X'10',C' ',X'0000',CL16',X'00000000'
   DC X'11',C' ',X'0000',CL16',X'00000000'
   DC X'12',C' ',X'0000',CL16',X'00000000'
   DC X'13',C' ',X'0000',CL16',X'00000000'
   DC X'14',C' ',X'0000',CL16',X'00000000'
   DC X'15',C' ',X'0000',CL16',X'00000000'
   DC X'16',C' ',X'0000',CL16',X'00000000'
   DC X'17',C' ',X'0000',CL16',X'00000000'
   DC X'18',C' ',X'0000',CL16',X'00000000'
   DC X'19',C' ',X'0000',CL16',X'00000000'
   DC X'1A',C' ',X'0000',CL16',X'00000000'
   DC X'1B',C' ',X'0000',CL16',X'00000000'
   DC X'1C',C' ',X'0000',CL16',X'00000000'
   DC X'1D',C' ',X'0000',CL16',X'00000000'
   DC X'1E',C' ',X'0000',CL16',X'00000000'
   DC X'1F',C' ',X'0000',CL16',X'00000000'
   DC X'20',C' ',X'0000',CL16',X'00000000'
| DC   | X'21', C', X'0000', CL16' | X'00000000' |
| DC   | X'22', C', X'0000', CL16' | X'00000000' |
| DC   | X'23', C', X'0000', CL16' | X'00000000' |
| DC   | X'24', C', X'0000', CL16' | X'00000000' |
| DC   | X'25', C', X'0000', CL16' | X'00000000' |
| DC   | X'26', C', X'0000', CL16' | X'00000000' |
| DC   | X'27', C', X'0000', CL16' | X'00000000' |
| DC   | X'28', C', X'0000', CL16' | X'00000000' |
| DC   | X'29', C', X'0000', CL16' | X'00000000' |
| DC   | X'2A', C', X'0000', CL16' | X'00000000' |
| DC   | X'2B', C', X'0000', CL16' | X'00000000' |
| DC   | X'2C', C', X'0000', CL16' | X'00000000' |
| DC   | X'2D', C', X'0000', CL16' | X'00000000' |
| DC   | X'2E', C', X'0000', CL16' | X'00000000' |
| DC   | X'2F', C', X'0000', CL16' | X'00000000' |
| DC   | X'30', C', X'0000', CL16' | X'00000000' |
| DC   | X'31', C', X'0000', CL16' | X'00000000' |
| DC   | X'32', C', X'0000', CL16' | X'00000000' |
| DC   | X'33', C', X'0000', CL16' | X'00000000' |
| DC   | X'34', C', X'0000', CL16' | X'00000000' |
| DC   | X'35', C', X'0000', CL16' | X'00000000' |
| DC   | X'36', C', X'0000', CL16' | X'00000000' |
| DC   | X'37', C', X'0000', CL16' | X'00000000' |
| DC   | X'38', C', X'0000', CL16' | X'00000000' |
| DC   | X'39', C', X'0000', CL16' | X'00000000' |
| DC   | X'3A', C', X'0000', CL16' | X'00000000' |
| DC   | X'3B', C', X'0000', CL16' | X'00000000' |
| DC   | X'3C', C', X'0000', CL16' | X'00000000' |
| DC   | X'3D', C', X'0000', CL16' | X'00000000' |
| DC   | X'3E', C', X'0000', CL16' | X'00000000' |
| DC   | X'3F', C', X'0000', CL16' | X'00000000' |
| DC   | X'40', C', X'0000', CL16' | X'00000000' |
| DC   | X'41', C', X'0000', CL16' | X'00000000' |
| DC   | X'42', C', X'0000', CL16' | X'00000000' |
| DC   | X'43', C', X'0000', CL16' | X'00000000' |
| DC   | X'44', C', X'0000', CL16' | X'00000000' |
| DC   | X'45', C', X'0000', CL16' | X'00000000' |
| DC   | X'46', C', X'0000', CL16' | X'00000000' |
| DC   | X'47', C', X'0000', CL16' | X'00000000' |
| DC   | X'48', C', X'0000', CL16' | X'00000000' |
| DC   | X'49', C', X'0000', CL16' | X'00000000' |
| DC   | X'4A', C', X'0000', CL16' | X'00000000' |
| DC   | X'4B', C', X'0000', CL16' | X'00000000' |
| DC   | X'4C', C', X'0000', CL16' | X'00000000' |
| DC   | X'4D', C', X'0000', CL16' | X'00000000' |
| DC   | X'4E', C', X'0000', CL16' | X'00000000' |
| DC   | X'4F', C', X'0000', CL16' | X'00000000' |

**END OF TABLE. (DON'T MOVE/REMOVE).**

**DC**

**X'FC'**

MATCHSV  DS F  SAVE AREA FOR 'MATCH' SUBROUTINE.

MATCHSV1 DS 5F

MATCHSTK DS 8CL12  STACK FOR MATCH SUBROUTINE.

**********************************************************************
* DETERMINE IF A GIVEN STRING MATCHES A GIVEN PATTERN. 'CEMT' RULES  *
* ARE USED FOR WILD CARDS. A '+' MATCHES ANY NON-BLANK CHARACTER. AN  *
* ASTERISK ('*') MATCHES ZERO OR MORE CHARACTERS.
* R0 = ADDRESS OF STRING TO MATCH
* R1 = ADDRESS OF PATTERN TO COMPARE STRING TO
* ON EXIT, R15 = Ø IF THE STRING MATCHES THE PATTERN
* R15 = 4 IF THE STRING DOES NOT MATCH THE PATTERN
* DESTROYS REGISTERS R0, R1, R4, R5, R6, R7, R8 AND R15
* BASIC ALGORITHM:
* P = CURRENT ADDR WITHIN THE PATTERN (R2)
* PL = REMAINING LENGTH WITHIN THE PATTERN (R7)
* S = CURRENT ADDR WITHIN THE STRING (R4)
* SL = REMAINING LENGTH WITHIN THE STRING (R5)
* WHILE (PL ≠ Ø)
* IF (*P == '*')
* LOOP
* P++;
* PL--;
* IF MATCH(P, PL, S, SL) RETURN (MATCHED);
* ELSEIF (SL == Ø) RETURN (NOTMATCHED);
* ELSE
* S++;
* SL--;
* ENDIF
* ENDLOOP
* ELSEIF (*P == '+')
* IF (*S == ' ') RETURN (NOTMATCHED);
* ENDIF
* ELSE
* IF (*S ≠ *P) RETURN (NOTMATCHED);
* ENDIF
* S++;
* SL--;
* P++;
* PL--;
* ENDWHILE
* IF (SL == Ø) RETURN (MATCHED);
* ELSE RETURN (NOTMATCHED);
* ENDIF
* NOTE THAT THE ABOVE ALGORITHM MAKES A RECURSIVE CALL TO 'MATCH'
* WHEN AN ASTERISK IS FOUND IN THE PATTERN. THIS IS HANDLED BY
* USING A MATCH STACK (MATCHSTK) WHICH CONTAINS THE CURRENT PATTERN
* AND STRING POINTERS.
* THE ROUTINE INITIALIZES THE MATCH STACK, THEN LOOPS PROCESSING
* UNTIL THE MATCH STACK IS EMPTY OR UNTIL WE GET A MATCH. R6 POINTS
* TO THE TOP ENTRY ON THE STACK.

MATCH EQU *
STM R4,R8,MATCHSV1
ST RE,MATCHSV1 SAVE OUR RETURN ADDRESS
L R1,SVRC1 ..POINT TO THE USER-ID PATTERN
L R0,SVRD ..POINT TO USER-ID
LA R6,MATCHSTK POINT TO TOP OF STACK
USING MSTKDS,R6
ST R1,MSTKPADD SET UP INITIAL STACK ENTRY
ST R0,MSTKSADD
CLC =C'++++++++',0(R1)
BE MATCH05
LA R1,7(.R1) POINT TO LAST BYTE OF PATTERN
LA R0,8 INIT TO 8 BYTE PATTERN
BALR RE,0 SET REG FOR LOOP
CLI 0(R1),C' ' TRAILING BLANK IN PATTERN?
BNE MATCH01Ø ..NO: GOT PATTERN LENGTH
BCTR R1,0 ..YES: DECR POINTER TO PATTERN
BCTR R0,RE ..THEN DECR LEN & KEEP CHECKING
MATCH01Ø EQU * R0 = PATTERN LENGTH
STH R0,MSTKPLEN SET PATTERN LENGTH
L R1,MSTKSADD GET STRING ADDRESS AGAIN
LA R1,7(.R1) POINT TO LAST BYTE OF STRING
LA R0,8 INIT TO 8 BYTE STRING
BALR RE,0 SET REG FOR LOOP
CLI 0(R1),C' ' TRAILING BLANK IN STRING?
BNE MATCH02Ø ..NO: GOT STRING LENGTH
BCTR R1,0 ..YES: DECR POINTER TO STRING
BCTR R0,RE ..THEN DECR LEN & KEEP CHECKING
MATCH02Ø EQU * R0 = STRING LENGTH
STH R0,MSTKSLEN SET STRING LENGTH
L R8,MSTKPADD GET ADDR WITHIN PATTERN
LH R7,MSTKPLEN GET REMAINING PATTERN LENGTH
L R4,MSTKSADD GET ADDR WITHIN STRING
LH R5,MSTKSLEN GET REMAINING STRING LENGTH
MATCH03Ø EQU * MATCH THE PATTERN & STRING ON STACK
LA R6,MSTKLEN(,R6) UPDATE STACK POINTER FOR NEXT TIME
MATCH04Ø EQU * LOOP THROUGH THE PATTERN
LTR R7,R7 ARE WE AT THE END OF THE PATTERN?
BZ MATCH09Ø ..YES: CHECK FOR END OF STRING
CLI 0(R8),C'*' ASTERISK IN PATTERN?
BNE MATCH06Ø ..NO: SKIP ASTERISK PROCESSING
LA R8,1(,R8) SKIP OVER ASTERISK IN PATTERN
BCTR R7,0
LTR R7,R7 '*' AT END OF PATTERN?
BZ MATCH095 ..YES: RETURN 'MATCHED'
ST R8,MSTKPADD SET UP PATTERN ADDR FOR RECURSIVE
STH R7,MSTKPLEN
MATCH05Ø EQU * TRY EACH POSSIBLE PATTERN
ST R4,MSTKSADD SET UP STRING ADDR FOR RECURSIVE CAL
STH R5,MSTKSLEN B MATCH03Ø TEST POSSIBLE PATTERN
MATCH06Ø EQU * NOT AN ASTERISK
LTR R5,R5 IS THERE ANY STRING LEFT TO COMPARE?
BZ MATCH11Ø NO: RETURN 'NO MATCH'
CLI 0(R8),C'+'+ PLUS IN PATTERN?
BE MATCH08Ø ..YES: ALLOW ANY CHARACTER

CLC  β(1,R8),β(R4)  DOES PATTERN MATCH STRING SO FAR?
BNE MATCH110  ..NO: RETURN 'NOT MATCHED'

MATCH080  EQU  *  PATTERN MATCHES STRING SO FAR
LA  R4,1(,R4)  GO TO NEXT STRING CHARACTER
BCTR  R5,0
LA  R8,1(,R8)  GO TO NEXT PATTERN CHARACTER
BCTR  R7,0
B  MATCH040  KEEP TRYING TO MATCH STRING VS PATT

MATCH090  EQU  *  END OF PATTERN
LTR  R5,R5  END OF STRING AT THE SAME TIME?
BNZ  MATCH110  ..NO: STRING DOES NOT MATCH

MATCH095  EQU  *  STRING MATCHES
SLR  RF,RF
MATCH100  EQU  *  RETURN TO THE CALLER
LTR  RF,RF  WAS THERE A MATCH.
BNZ  MATCH105  NO-BRANCH TO MATCH105.
MVI  MTCSW,C'1'  INDICATE MATCH.

MATCH105  EQU  *  RETURN TO THE CALLER
L  RE,MATCHSV  RESTORE RETURN ADDRESS
LM  R4,R8,MATCHSV1
BR  RE  RETURN TO CALLER

MATCH110  EQU  *  STRING DOES NOT MATCH
LA  RF,4  INDICATE 'NO MATCH'
SH  R6,=Y(MSTKLEN)  POP RECURSIVE CALL STACK
LA  R1,MATCHSTK  GET STACK START ADDRESS
CR  R6,R1  BOTTOM OF STACK?
BNH  MATCH100  ..YES: RETURN 'NOT FOUND'
L  R8,MSTKPADD  RESTORE SAVED PATTERN POINTER
LA  R7,MSTKPLEN
L  R4,MSTKSADD  RESTORE SAVED STRING POINTER
LH  R5,MSTKSLEN
LTR  R5,R5  ANY MORE STRING LEFT?
BZ  MATCH110  ..NO: RETURN 'NOT FOUND'
LA  R4,1(,R4)  ..YES: EXTEND '*' 1 MORE BYTE
BCTR  R5,0
B  MATCH050  GO CHECK MATCH AGAIN
DROP  R6
LTORG  DISPLAY LITERALS.
DS  ØD
CCB  CCB  SYSLOG,CCW
DC  C'CCW'
DS  ØD
CCW  CCW  X'09',CNWK,X'20',L'CNWK
DC  C' '  DO NOT MOVE/REMOVE THIS STATEMENT.
CNWK  DC  CL68' '  CONSOLE WORK AREA.
MSTKDS  DSECT  MATCH STACK DSECT
MSTKPADD  DS  F  ..ADDRESS OF PATTERN
MSTKSADD  DS  F  ..ADDRESS OF STRING TO MATCH
MSTKPLEN  DS  H  ..REMAINING LENGTH OF PATTERN
MSTKSLEN  DS  H  ..REMAINING LENGTH OF STRING
MSTKLEN  EQU  *-MSTKDS  LENGTH OF ONE ENTRY IN STACK
DPCKJPE DC X'FF'
* SYSTEM COMMUNICATIONS REGION. (SYSCOM).
SYSCOM SYSCOM,
* PARTITION COMMUNICATIONS REGION. (MAPCOMR).
MAPCOMR,
* PARTITION CONTROL BLOCK. (MAPPCB).
MAPPCB,
* RECORDER FILE TABLE. (MAPRFTAB).
MAPRFTAB,
END

DPRTNC
The DPRTNC subroutine allows the caller to GET/SET the $RC return code or get the $MRC return code. One parameter must be passed consisting of two fields. The first field, the function code, is one byte and must contain one of the following:

- Reset ‘$rc’, ‘$mrc’ and all ‘on’ conditions.
- Get the last ‘$rc’ return code.
- Get the maximum ‘$mrc’ return code.
- Set the ‘$rc’ return code.

The second field is four bytes and will contain the values, in EBCDIC format (ie $RC 4 is returned as C'0004'), of either the $RC or $MRC return codes for functions ‘1’ and ‘2’, or it must contain a four-digit value that you wish the $RC to be set to.

For function ‘3’, leading zeros must be entered if the value being set is less than 1000. If the value entered isn’t numeric, a ‘6’ is returned in the first field. If function ‘0’ is selected this field is ignored.

Notes:
- If the value of the first field isn’t ‘0’ through ‘3’, a ‘7’ is returned in this field.
- This subroutine uses an SVC 4 (load) to load $IJBCJC into storage. If the load is unsuccessful the first field will contain an ‘8’.
- $IJBCJC inserts a return code into register 15 if it was unsuccessful.
If this occurs, a PDUMP is issued so register 15 can be examined. Also, to indicate this, the first field is set to ‘9’. See the VSE Messages and Codes manual (CONDJC macro return codes) for the meaning of these return codes.

- If using function code ‘0’, you must resubmit any ‘// on’ statements because, in addition to the $RC and $MRC being reset to zero, any ‘on’ statements are set to blanks.
- If using function code ‘3’, if the second field contains a value greater than 4095, VSE will force the value to 4095.
- Because of VSE restrictions, it’s suggested that the $RC return code be set to a value less than 16, or to a value of 128. If set at 16 or greater, but not 128, you’ll be unable to check it with a ‘// if’ $RC statement, because VSE will cancel any job with a return code greater than 15 but not 128. One way to get around this is to use a ‘// on’ statement, which would eliminate any idea of setting and checking any return code greater than 15 using the ‘// if’ statement.

The calling sequence is:

COBOL:

```cobol
call 'dprtnc' using fields.
```

ALC:

```alc
la 13,savearea (13 can also be r13 or rd).
call dprtnc,(fields)
.
.(mainline part of program).
.
savearea dc 18f'0'
```

RPG2:

```rpg
  call 'dprtnc' xx (xx=any unused in-
parm fields dicator)
  setof xx
```

An 18-word save area must be passed through register 13 by the user (standard COBOL linkage).
SAVE (14,12)
BALR 3,Ø
USING *,3
ST 13,SAVEAREA+4
LA 13,SAVEAREA
B DPRBEG
DC C'DPRNTC STARTS HERE. ' INSERT EYE CATCHER.

DPRBEG EQU *
XC CJCFLDS,CJCFLDS CLEAR FUNCTION/ADDRESS FIELDS.
L 4,Ø(1) GET ADDRESS OF PASSED PARAMETER.
MVC PRMFlds,Ø(4) SVE PASSED FIELDS.
MVC PRMFUNCS,PRMFUNC SVE FUNCTION CODE.
CLI PRMFUNC,C'Ø' IS FUNCTION CODE A 'Ø'. (RESET).
BNE *+12 NO-SKIP NEXT TWO (2) INST.
MVI PRMFUNC,C'6' INDICATE 'RESET' FUNCTION.
B DPRNU3 BRANCH TO DPRNU3.
CLI PRMFUNC,C'1' IS FUNCTION CODE LOWER THAN '1'.
BL DPRFER YES-BRANCH TO DPRFER.
CLI PRMFUNC,C'3' IS FUNCTION CODE HIGHER THAN '3'.
BH DPRFER YES-BRANCH TO DPRFER.
CLI PRMFUNC,C'3' IS FUNCTION CODE '3'.
BE *+10 YES-SKIP NEXT INST.
MVC PRMADDR,=C'ØØØØ' SET ADDRESS FIELD TO ZERO.
LA 11,PRMADDR LOAD ADDRESS OF PRMADDR TO REG 11.
LA 12,L'PRMADDR LOAD LENGTH OF PRMADDR TO REG 12.

DPRNU1 EQU *
CLI Ø(11),C'Ø' IS THIS POSITION LOWER THAN ZERO.
BL DPRNER YES-BRANCH TO DPRNER.
CLI Ø(11),C'9' IS THIS POSITION HIGHER THAN NINE.
BH DPRNER YES-BRANCH TO DPRNER.
LA 11,1(11) INCREMENT REG 11 BY ONE (1).
BCT 12,DPRNU1 BRANCH TO DPRNU1 UNTIL REG 12 ZERO.

DPRNU3 EQU *
STM 13,14,SV1314 SVE REGS 13 AND 14.
LA 13,SAVEAREA LOAD ADDRESS OF SAVEAREA TO REG 13.
LA 1,=CL8'$IJBCJC' LOAD ADDRESS OF $IJBCJC TO REG 1.
SR 0,Ø CLEAR REG 0.
SR 15,15 CLEAR REG 15.
SVC 4 ISSUE LOAD.
LTR 15,15 WAS LOAD SUCCESSFUL.
BNZ DPRLDE NO-BRANCH TO DPRLDE.
LR 15,1 LOAD ADDRESS TO REG 15.
ST 1,SVØ1 SVE ADDRESS OF LOAD ADDRESS.
L 1,=X'FF000000' SET ENABLE STORAGE PROT KEY.
SVC 13 GO DO IT.
PACK DOUBWORD,PRMFUNC PACK FUNCTION CODE.
CVB Ø,DOUBWORD CONVERT IT TO BINARY.
ST Ø,CJCFUNC SVE IT.
MVC CJCADDR,PRMADDR SVE PARAMETER ADDRESS FIELD.
LA 1,CJCADDR LOAD ADDRESS OF RETURN CODE FIELD TO
LA 13,SAVEAREA LOAD ADDRESS OF SAVEAREA TO REG 13.
BALR 14,15 BRANCH TO $IJBCJC.
ST 15,SV15               SVE RETURN CODE.
L  1,=X'FF00000F'       RESET ENABLE STORAGE PROT KEY.
SVC 12                   GO DO IT.
L 15,SV15                 RESTORE REG 15.
LM 13,14,SV1314           RESTORE REG 13 AND 14.
LTR 15,15                 WAS CALL SUCCESSFUL.
BNZ DPRCLE                NO-BRANCH TO DPRCLE.
MVC PRMADDR,CJCADDR       MVE SAVED PARAMETER ADDRESS FIELD.
MVC PRMFUNC,PRMFUNCS      RESTORE SAVED FUNCTION CODE.
B DPRRTN                  BRANCH TO DPRRTN.

DPRNER EQU *              D
MVI PRMFUNC,C'6'          INDICATE PRMADDR NOT NUMERIC.
B DPRRTN                  BRANCH TO DPRRTN.

DPRFER EQU *              D
MVI PRMFUNC,C'7'          INDICATE FUNCTION CODE ERROR.
B DPRRTN                  BRANCH TO DPRRTN.

DPRLDE EQU *              D
MVI PRMFUNC,C'8'          INDICATE LOAD ERROR.
B DPRCLE7                 BRANCH TO DPRCLE7.

DPRCLE EQU *              D
CH 15,=H'4'               WAS RETURN CODE '4'.
BNE DPRCLE3               NO-BRANCH TO DPRCLE3.
MVC PRMADDR,=C'0000'      SET ADDRESS FIELD TO ZERO.
B DPRRTN                  BRANCH TO DPRRTN.

DPRCLE3 EQU *              D
MVI PRMFUNC,C'9'          INDICATE CALL ERROR.

DPRCLE7 EQU *              D
PDUMP DPRTNCS,DPRTNCE     D

DPRRTN EQU *              D
MVC Ø(L'PRMFLDS,4),PRMFLDS MVE PARAMETER FIELDS.
L 13,SAVEAREA+4          RETURN (14,12)

DPRTNCS DC C'DPRTNCS STORAGE HERE.' INSERT EYE CATCHER.
PRMFLDS DS ØCL5
PRMFUNC DS C
PRMADDR DS CL4
PRMFUNCS DS C
DS ØD
CJCFLDS DS ØCL8
CJCFUNC DC F'0'
CJCADDR DC F'0'
SV1314 DC 2F'0'
SV01 DC F'0'
SV15 DC F'0'
DOUBWORD DS D
SAVEAREA DS 18F
DPRTNCE DC X'FF'

Robert Botsis
Senior Systems Programmer (USA)    © Xephon 1998
Converting macros to define statements – part 2

The latest versions of CICS do not provide macro resource definitions for defining transaction (PCT) and program (PPT) entries, and VSAM file (FCT) entries must be assembled and then migrated to the CICS System Definition (CSD) file. This month we conclude the article on creating replacement macros that process the obsolete definitions and build CSD DEFINE statements.

```plaintext
&X       SETC  '&X','.,'  
.IS1Ø  ANOP  
&X       SETC  '&X'.'NEPCLAS=&NEPCLAS'  
&NEPCLAS(&PMAX) SETC '&NEPCLAS'  
.
.NONEPCL AIF (T'&RAQ   EQ 'O').NORAQ 
   AIF (K'&X EQ Ø).IS11  
&IS     SETC 'ARE'  
&X       SETC 'R'.', ' 
.IS11  ANOP  
&X       SETC 'R'.,'RAQ=&RAQ'  
&RAQ(&PMAX) SETC '&RAQ'  
.
.NORAQ  ANOP  
&     SETC 'X'  
.
.BUILD  ANOP  
&X       SETC 'X'  
.
.NEXT   AIF (K'&X+K'&RDO(&J) LT 72).CONCAT  
PUNCH '&X'  
AGO   .BUILD  
.CONCAT ANOP  
&X        SETC 'X&RDO(&J)'  
&J       SETA &J+1  
AIF (&J LE &I).NEXT  
PUNCH (K'&X LE 6).DESCR  
AIF (K'&X EQ Ø).NOPROF  
.
&I       SETA 1  
PUNCH 'DESCRIPTION(&DESCR)'  
.
.AIF (K'&P EQ Ø).ADDPF
```
.PFLOOP AIF ('&DVSUPC(&I)' NE '&DVSUPC(&PMAX)') .NEXTP
AIF ('&PRTCMPC(&I)' NE '&PRTCMPC(&PMAX)') .NEXTP
AIF ('&RTIMOC(&I)' NE '&RTIMOC(&PMAX)') .NEXTP
AIF ('&SCRNSZC(&I)' NE '&SCRNSZC(&PMAX)') .NEXTP
AIF ('&INBFMHC(&I)' NE '&INBFMHC(&PMAX)') .NEXTP
AIF ('&JFILEIC(&I)' NE '&JFILEIC(&PMAX)') .NEXTP
AIF ('&LOGRECC(&I)' NE '&LOGRECC(&PMAX)') .NEXTP
AIF ('&MODEMNMC(&I)' NE '&MODEMNMC(&PMAX)') .NEXTP
AIF ('&MSGJRNC(&I)' NE '&MSGJRNC(&PMAX)') .NEXTP
AIF ('&NEPCLAC(&I)' NE '&NEPCLAC(&PMAX)') .NEXTP
AIF ('&RAQC(&I)' EQ '&RAQC(&PMAX)') .OLDPF
.

.ANEXT AIF (&I LT &PMAX) .ILEPMAX
MNOTE 4,'PROFILE TABLE EXCEEDED, SEARCH SUSPENDED'
AGO .ADDPF
.

.ILEPMAX AIF (&I LE &NP) .PFLOOP
.

.ADPF AIF (&I LE &NP) .ADDPF
.

&NP SETA &NP+1

&DVSUPC(&I) SCTC '&DVSUPC(&PMAX)'
&PRTCMPC(&I) SCTC '&PRTCMPC(&PMAX)'
&RTIMOC(&I) SCTC '&RTIMOC(&PMAX)'
&SCRNSZC(&I) SCTC '&SCRNSZC(&PMAX)'
&INBFMHC(&I) SCTC '&INBFMHC(&PMAX)'
&JFILEIC(&I) SCTC '&JFILEIC(&PMAX)'
&LOGRECC(&I) SCTC '&LOGRECC(&PMAX)'
&MODEMNMC(&I) SCTC '&MODEMNMC(&PMAX)'
&MSGJRNC(&I) SCTC '&MSGJRNC(&PMAX)'
&NEPCLAC(&I) SCTC '&NEPCLAC(&PMAX)'
&RAQC(&I) SCTC '&RAQC(&PMAX)'
&PFX(&I) SCTC '&PFX(&PMAX)'
.

&DVSUPC(&PMAX) SCTC ''
&PRTCMPC(&PMAX) SCTC ''
&RTIMOC(&PMAX) SCTC ''
&SCRNSZC(&PMAX) SCTC ''
&INBFMHC(&PMAX) SCTC ''
&JFILEIC(&PMAX) SCTC ''
&LOGRECC(&PMAX) SCTC ''
&MODEMNMC(&PMAX) SCTC ''
&MSGJRNC(&PMAX) SCTC ''
&NEPCLAC(&PMAX) SCTC ''
&RAQC(&PMAX) SCTC ''
.

&PFID(&I) SCTC '&GROUPC'.#####
&PFID(&I) SCTC '&PFID(&I)'(1,8-K1&I).'&I'
.

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
PUNCH '       PROFILE(&PFID(&I))' 
PUNCH '*'

.*
PUNCH 'DEFINE PROFILE(&PFID(&I)) GROUP(&GROUPC)' 
&K SETA Ø

.*
AIF ('&DVSUPC(&I)' EQ '').XDVS

K SETA &K+1
&RDO(&K) SETC 'INBFMH(INBFMH)' 

.*
XDVS AIF ('&PRTCMPC(&I)' EQ '').XPRT

K SETA &K+1
&RDO(&K) SETC 'PRINERCOMP(PRTCOMP)'

.*
XPRT AIF ('&RTIMOC(&I)' EQ '').XRTI

K SETA &K+1
&RDO(&K) SETC 'RTIMEOUT(TRIMOUT)'

.*
XRTI AIF ('&SCRNSZC(&I)' EQ '').XSCR

K SETA &K+1
&RDO(&K) SETC 'SCRNSIZE(SCRNSIZ)'

.*
XSCR AIF ('&INBFMHC(&I)' EQ '').XINB

K SETA &K+1
&RDO(&K) SETC 'INBFMH(INBFMH)'

.*
XIHB AIF ('&JFILEIC(&I)' EQ '').XJFI

K SETA &K+1
&RDO(&K) SETC 'JOURNAL(2)

.*
XJFI AIF ('&LOGRECC(&I)' EQ '').XLOG

K SETA &K+1
&RDO(&K) SETC 'LOGREC(LOGREC)'

.*
XLOG AIF ('&MODENMC(&I)' EQ '').XMOD

K SETA &K+1
&RDO(&K) SETC 'MODENAME(MODENAM)'

.*
XMOD AIF ('&MSGJRNC(&I)' EQ '').XMSG

K SETA &K+1
&RDO(&K) SETC 'MSGJRNL(MSGJRNL)'

.*
XMSG AIF ('&NEPCLAC(&I)' EQ '').XNEP

K SETA &K+1
&RDO(&K) SETC 'NEPCLASS(NEPCLAS)'

.*
.XNEP AIF ('&RAQC(&I)' EQ '').XRAQ
&_K SETA &K+1
&_RDO(&K) SETC 'RAQ(&RAQ)'
.*
.XRAQ ANOP
&_J SETA 1
.*
.BUILDP ANOP
&_X SETC '       '
.*
.NEXTPF AIF (K'&X+K'&RDO(&J) LT 72).CONCATP
PUNCH '&X'
AGO .BUILDP
.CONCATP ANOP
&_X SETC '&X&RDO(&J)'
&_J SETA &J+1
AIF (&J LE &K).NEXTPF
AIF (K'&X LE 6).DESCRP
PUNCH '&X'
.DESCRP ANOP
.*
PUNCH '       DESCRIPTION(&DESCR)'
PUNCH '*THE ABOVE PROFILE GENERATED FROM THE FOLLOWING PARAMETERS' 
.PLOOP AIF (K'&P LE 71).LE71
&_X SETC '*'. '&P'(1,71)
PUNCH '&X'
&_P SETC '&P'(72,K'&P)
AGO .PLOOP
.LE71 AIF (K'&P EQ Ø).NOPROF
PUNCH '*&P'
.*
AGO .NOPROF
.*
.OLDPF ANOP
PUNCH '       PROFILE(&PFID(&I)) '
PUNCH '*THE ABOVE PROFILE NAME CREATED BY ENTRY &PFX(&I)'
.*
.NOPROF ANOP
PUNCH '*'
.*
&_X SETC ''
.*
AIF (T'&CICS EQ 'O').NOCICS
&_X SETC '&X'. 'CICS=&CICS '
.*
.NOCICS AIF (T'&SUBSET EQ 'O').NOSUBST
&_X SETC '&X'. 'SUBSET=&SUBSET '
.*
.NOSUBST AIF (T'&COMPAT EQ 'O').NOCOMPT

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
MNOTE 4,'THE FOLLOWING PARAMETERS WERE IGNORED &X' .* 
NOMNT ANOP .END MEND GBLC &DESCR

DFHPPT MACRO

MACRO &NAME DFHPPT &TYPE=, TYPE OF ENTRY * 
&TYPE=, OBSOLETE OPTION * 
&PROGRAM=, PROGRAM IDENTIFICATION *
&MAPSET=, MAPSET IDENTIFICATION *
&PARTSET=, PARTITIONSET IDENTIFICATION *
&FN=, FUNCTIONAL GROUP *
&PGMLANG=, PROGRAM LANGUAGE *
&SUFFIX=, P-P-T SUFFIX *
&SUBSET=, NO* ALLOW FOR OLD SUBSET TABLES *
&RELOAD=, NO* PROGRAM TO BE RELOADED? *
&USAGE=, PROGRAM USAGE *
&PGMSTAT=, ENABLED* PROGRAM STATUS *
&DLI=, NO, DL/I PROGRAM *
&PAGENXD=, PAGE INDEX REQUEST *
&INDEX=, FULL INDEX OPTION *
&STARTER=, PREGENERATED TABLES ONLY *
&RSL=, RESOURCE SECURITY LEVEL *
&RES=, PROGRAM RESIDENCE INDICATOR *
&SHR=

.* ABOVE * INDICATES DEFAULT VALUE REMOVED (DEFAULT PRECEDES *)
.* THESE ARE ALSO THE CSD DEFAULTS AND WOULD ONLY CREATE REDUNDANT
.* PARAMETERS. THESE (AND OTHERS) MAY BE MODIFIED FOR INDIVIDUAL
.* PREFERENCES.
.*
GBLC &GROUPP,&SUFXP
LCLA &I,&J
LCLC &X,&RDO(50)
GBLC &IDS(500),&CMTS(500),&DESCR
GBLA &IDN
.
AIF ('&TYPE' NE 'INITIAL').NOINIT
AIF (T'&SUFFIX EQ 'O').NOINIT
&SUFXP SETC '&SUFFIX'
.
.NOINIT AIF ('&TYPE' NE 'ENTRY').END
.
&I SETA Ø
AIF (&IDN EQ Ø).FIRSTID
.
.IDLOOP ANOP
.
&I SETA &I+1
AIF ('&PROGRAM' NE '&IDS(&I)').NOTID
PUNCH '*&PROGRAM IS DUPLICATED ABOVE, SEE &CMTS(&I)'
AGO .NOMNT
.
.NOTID AIF (&I LT &IDN).IDLOOP
.
.FIRSTID ANOP
.
&I SETA Ø
&J SETA 1

© 1998. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (940) 455 7050, fax (940) 455 2492.
.*
AIF ('&GROUPP' NE '').GROUP
AIF ('&SYSPARM' EQ '').NOSPARM
&GROUPP SETC '&SYSPARM'
AGO .GROUP
.NOSPARM ANOP
&GROUPP SETC 'PCTXX&SUFXP'
.*
.GROUP AIF (T'&PROGRAM NE '0').PROGRAM
AIF (T'&MAPSET NE '0').MAPSET
AIF (T'&PARTSET NE '0').PARTSET
.*
.PROGRAM ANOP
PUNCH 'DEFINE PROGRAM(&PROGRAM) GROUP(&GROUPP)'
.*
AIF ('&DESCR' NE '').DESCRX
&DESCR SETC 'PPT GROUP=&GROUPP'
.DESCRX ANOP
.*
&IDN SETA &IDN+1
&IDS(&IDN) SETC '&PROGRAM'
&CMTS(&IDN) SETC '&DESCR'
.*
&I SETA &I+1
AIF (T'&PGMLANG EQ '0').NOLANG
AIF ('&PGMLANG' EQ 'PL/I').PLI
&RDO(&I) SETC 'LANGUAGE(&PGMLANG)'
AGO .LANG
.NOLANG ANOP
&RDO(&I) SETC 'LANGUAGE(ASSEMBLER)'
AGO .LANG
.PLI ANOP
&RDO(&I) SETC 'LANGUAGE(PLI)'
AGO .LANG
.*
.MAPSET ANOP
PUNCH 'DEFINE MAPSET(&MAPSET) GROUP(&GROUPP)'
AGO .LANG
.*
.PARTSET ANOP
PUNCH 'DEFINE PARTITIONSET(&PARTSET) GROUP(&GROUPP)'
.*
.LANG AIF (T'&PGMSTAT EQ '0').NOPSTAT
&I SETA &I+1
&RDO(&I) SETC 'STATUS(&PGMSTAT)'
.*
.NOPSTAT AIF (T'&RELOAD EQ '0').NORLOAD
&I SETA &I+1
&RDO(&I) SETC 'RELOAD(&RELOAD)'
.*

.* NORLOAD AIF (T'&RES EQ 'O').NORES
&I SETA &I+1
&RDO(&I) SETC 'RESIDENT(&RES)'
.*
.* NORES AIF (T'&RSL EQ 'O').NORSL
MNOTE 4,'THE RSL KEYWORD IS NOT VALID IN CICS 4.1'
.*
.* NORSL AIF (T'&USAGE EQ 'O').NOUSAGE
&I SETA &I+1
&RDO(&I) SETC 'USAGE(TRANSIENT)'
.*
.* NOUSAGE AIF (T'&SHR EQ 'O').BUILD
&I SETA &I+1
&RDO(&I) SETC 'USELPACOPY(&SHR)'
.*
* KEYWORDS PROCESSED, PUNCH RDO DATA
*.
.* .BUILD ANOP
&X SETC ''
.*
.* .NEXT AIF (K'&X+K'&RDO(&J) LT 72).CONCAT
PUNCH '&X'
AGO .BUILD
.CONCAT ANOP
&X SETC '&X&RDO(&J)'
&J SETA &J+1
AIF (&J LE &I).NEXT
AIF (K'&X LE 6).DESCR
PUNCH '&X'
.DESCRIPTION ANOP
.*
.* PUNCH ' DESCRIPTION(&DESCR)'
.*
&X SETC ''
.*
.* AIF (T'&CICS EQ 'O').NOCICS
&X SETC '&X'.CICS=&CICS'
.*
.* .NOCICS AIF (T'&SUBSET EQ 'O').NOSUBST
&X SETC '&X'.SUBSET=&SUBSET'
.*
.* .NOSUBST AIF (T'&DLI EQ 'O').NODLI
&X SETC '&X'.DLI=&DLI'
.*
.* .NODLI AIF (T'&PAGENXD EQ 'O').NOPAGE
&X SETC '&X'.PAGENXD=&PAGENXD'
.*
.* .NOPAGE AIF (T'&INDEX EQ 'O').NOINDEX
&X SETC '&X'.INDEX=&INDEX'
.*
SAMPLE JCL

The following JCL processes DFHPCT table entries. Replace all occurrences of PCT with FCT and PPT to process DFHFCT and DFHPPT entries, respectively.

```plaintext
//xxxxxxxx JOB ,...
_Api---+----------------------------------------------------------*/
_Api---+ CONVERT CICS MACRO DEFINITIONS TO BATCH RDO STATEMENTS
_Api---+----------------------------------------------------------*/
_Api---S EXEC ASMHC,
_Api---  // PARM.C=(NOBJECT,'XREF(SHORT)',DECK,TERM,ALIGN,
_Api---    //  'LINECOUNT(55)',SYSPARM(MP3PCT))
_Api---S SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR
_Api---S SYSPUNCH DD DSN=ceda.define.source,DISP=OLD
_Api---S SYSPRINT DD SYSOUT=*                                  
_Api---S SYSTERM  DD SYSOUT=*                                 
_Api---S SYSPIN DD DSN=dummy.cics.maclib(DFHPCT),DISP=SHR
_Api---  // DD DSN=cics.table.source(PCTDEFS1),DISP=SHR
_Api---  // DD DSN=cics.table.source(PCTDEFS2),DISP=SHR
_Api---  // DD DSN=dummy.cics.source(ENDCARD),DISP=SHR
```

The ‘DSN=dummy.cics.maclib library’ is only included if the above macros are to be included within a separate library to be concatenated with the source. Optionally, these may be included within the SYSLIB definition. The ENDCARD consists of an Assembler END statement which is only needed if the preceding table source does not contain an END statement.

---

Keith H Nicaise  
Technical Services Manager  
Touro Infirmary (USA)  © Xephon 1998
CICS news

CICS users can now benefit from a joint announcement by IBM and Intelligent Environments, based around IBM’s Network Station NCs and the software vendor’s Amazon legacy integration and development software.

Intelligent Environments believes that its software is ideal for IBM NCs in IBM shops, because it’s geared to combining existing transactions and legacy systems with new processes and logic. Incorporation and reuse extends to CICS transactions and MQSeries messages, as well as 3270 or 5250 screens.

Intelligent Environments has also announced that its Amazon Web development tool will now integrate with SilverStream Software’s Web application platform. The integrated products, called Beyond JDBC, support CICS, 3270 and 5250 terminal emulation, APPC, and MQSeries.

This will mean that developers using start-up SilverStream’s products will be able to build Web applications and integrate them with legacy systems through Amazon.

For further information contact:
Intelligent Environments, 67 Bedford Street, Burlington, MA 01776, USA.
Tel: (800) 669 8777.
Intelligent Environments, 8 Windmill Business Village, Brooklands Close, Sunbury-on-Thames, Middx, TW16 7DY, UK.
Tel: (01932) 772266.

* * *

CICS news

CICS users can now access Java Beans and COM interfaces following the announcement of Cool:Gen by Sterling Software. Cool:Gen is a development environment that will provide client access to enterprise server components through the automatic generation of Java Beans and Component Object Model interfaces. The new facility means components, delivered using Cool:Gen, can be automatically deployed on to the Internet through Active Server Page and Java Beans.

The idea is that Cool:Gen will generate COM and Java elements, or proxies, which will act as gateways between open clients and server components. While transaction throughput of application transactions will be maintained, and continue to be managed by CICS (and other TP monitors such as IMS, or Tuxedo), the means by which the components can be accessed are now completely open, says the supplier.

Both the COM and the Java proxies communicate with the generated components via TCP/IP and LU6.2, and Sterling says support for MQSeries will follow shortly.

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.