154 CICS update

September 1998

In this issue

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38030
From USA: 01144 1635 38030
E-mail: xephon@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Contributions
Articles published in *CICS Update* are paid for at the rate of £170 ($250) per 1000 words and £90 ($140) per 100 lines of code for original material. To find out more about contributing an article, without any obligation, please contact us at any of the addresses above and we will send you a copy of our *Notes for Contributors*.

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.

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.

Printed in England.

CICS resource maintenance systems

Any CICS site with more than 30 CICS regions has a CICS table management requirement. This requirement increases exponentially with the number of CICS regions, because nearly all tables have interdependencies. Most CICS shops will have a table management system of sorts.

Having maintained or developed CICS table management systems varying from elaborate to simple, in a number of countries, I have a few suggestions that may be helpful to those maintaining such systems, or those intending to set one up:

- The system should be easy to maintain and change at short notice. CICS table requirements may change as a result of new PTFs. For this reason, REXX/ISPF based systems are preferable to COBOL or CLIST-based systems. Ease of maintenance requires strict observance of naming standards for all objects in the system – variables, panels, skeletons, REXX EXECs, etc. Ideally, if time permits, a data dictionary for REXX variables is recommended. Misleading variable names are a possible source of bugs. All REXX EXECs should have the same general format.
- The system must be easy to use and be forgiving, since table maintenance is often assigned to the trainee system programmer. All delete or production changes should be prefaced by 'confirm' panels.
- In multi-MVS image systems, the system should be implemented on only one image, to prevent possible discrepancies and dual maintenance.
- The following functions are recommended, in rough order of importance. Each site must make a decision on the trade-off between the implementation effort and the benefits obtained:
  - A straightforward back-up and recovery procedure (preferably automatic) for the whole system.
  - A register of all CICS systems – CICS version, maintenance

level, image, description, development/test/production, etc.

- A register of all MVS images – maintenance level (of CICS SVCs), description, development/test/production, etc.
- For each CICS system, a register of CICS tables, with functions to edit/browse, assemble, transmit the table load module to a staging library on the target image, examine assembly output, back-up/restore of table source and load modules. Any copy books referenced by tables should also be controlled/maintained by the system.
- The CICS table register should also record the outcome of the assembly (OK, warning, failure, or abend). Transmit will be disabled for compilation failures/abends.
- The system should not allow assembly of tables with inconsistent maintenance levels for the target CICS/image. As an additional cross-check, edit macros can be written to check the actual CICS version of some tables. Assembly of tables should be consistent with standard promotion paths.
- The table display panel should record when, and by whom, a table was updated and assembled (and, optionally, transmitted).
- 'Mass assembly' and 'mass transmit' procedures so that, for example, all tables for a given CICS, given image, or given CICS version could be assembled or transmitted. These facilities would definitely require 'confirm' panels.
- A 'database' of SITs/SIT parameters to provide a facility to check for parameter consistency and standardization. The values for a given SIT parameter (eg GRPLIST) for all CICS regions could be displayed together.
- Similarly, a 'database' of PLTPIs and PLTSDs could be provided. Those sites which have implemented a 'PLT processor' system could provide a 'database' of PLT processor control statements.
- A 'database' of DCTs and JCTs.

– A log/audit trail of the more important actions, to assist in problem diagnosis.

The registers and ‘databases’ discussed above would be implemented in ISPF tables. The table design should minimize data duplication.

The system should be concurrently usable to the greatest extent possible by several system programmers. Care should be taken that any ISPF table should be opened in WRITE mode for the minimum time needed to perform a given update. For the occasional (and unavoidable) contention problem, ‘Table in use by XXXXXXX – please try later’ messages are to be preferred to REXX crashes.

- A ‘new CICS’ system could be developed concurrently with this system. The user would enter the system parameters – APPLID, SYSID, third-party products, etc – and the system would automatically generate:

  – The JCL for the CICS start-up procedure.

  – The CICS system files (DFHTEMP, etc).

  – The CICS CSD.

  – The CICS tables, including the RCT if required.

  This system would be of great value in busy sites where new CICSs must be commissioned at very short notice. It would be based upon a ‘reference’ CICS with standard CSD, CICS tables, etc. Obviously some tailoring may be required (of the CSD for example) after the set-up REXX has run, but this could be minimized with suitable design of the initial parameter input panel.

  Ideally, JCL for existing CICSs should also be maintained by the system, rather than by hand, to avoid discrepancies between regions, and to ease the task of new product/new version installs.

- In a previous article entitled *DB2 queries for CSD data under CICS 3.1*, in *CICS Update*, Issue 61, December 1990, I discussed the implementation of a DB2 database for CICS CSD objects.

This same database could be used as an active control, rather than just as a reporting system. This means that, after the initial set-up, all CSD changes would be controlled from a REXX/ISPF front-end to this database. This front-end would generate jobs containing appropriate DFHCSDUP control statements to run on the target MVS image, including JES MODIFY statements if a CEDA INSTALL is required (or setting up of a COLD start flag for automated operations systems). The benefits of such a system could be very great, for example in systems with many ISC/MRO connections. Maintaining consistency and control of (say) IOAREALEN, receive/send sizes, suffixes, etc is no easy task. Maintaining the CSD from the REXX/ISPF/DB2 front-end allows enforcement of cross-system controls and standards and should reduce errors. The benefits for reporting and CICS release change work are clear. An additional benefit from 'marrying' the CSD/DB2 database and the CICS table maintenance system is that tables which migrate to the CSD in higher CICS releases (eg the DCT) can be migrated more easily and with fewer errors.

For CICS 4.1 systems and above, EXEC CICS CREATE statements could be substituted for DFHCSDUP control statements. Files of CREATE statements would be read and executed by a program running at PLTPI time to define/delete CICS resources. However, implementation of a CREATE-based system would involve a lot of work and a whole new paradigm for looking at resource definition, and therefore the benefits over CSD definitions need to be carefully assessed. I am aware that a 'central point of control' CICS resource definition system has been hinted at for future releases of CICSplex/SM.

- If resources permit, the above system(s) should be implemented in a 'development' and 'production' mode. This would allow the system maintainer to test changes, add new features, fix bugs, etc without affecting 'production' work by other systems programmers on CICS tables/CSDs. This would require a 'mirror' set of REXX EXECs, ISPF tables, etc.
- An additional, but related, project would be to set up a system similar to that for CICS tables for system programmer-written

programs, exits, utilities and so on. Such programs present similar challenges to those discussed for tables, namely:

– Control of source code and associated copy books.

– Compilation with the correct CICS SYSLIB maintenance level for the target CICS/image.

– Recording when and by whom the program was changed/ compiled.

– Control of promotion procedures.

– Dependencies on resources defined in CICS tables or CSD.

David Roth
CICS Consultant (UK)

Maintaining statistics for non-CICS resources

The usage count of private, non-CICS resources can be maintained in a table, which can be incremented by a program every time the resource is used.

Sometimes, the logical flow of an application program results in its usage not being included in the statistics. To make such events visible, you may want to maintain a count in each CICS session.

For the duration of a CICS session, usage counts can be maintained in a CWA or GETMAINED area, and must be investigated before the end of a job, or saved through a dump, a notice on a papersheet, etc.

For a long-term log, it is necessary to automate this procedure by maintaining a private file.

A convenient medium for a long-term log would be the CICS statistics, and its program section (with program usage counts) – but how do you get non-CICS resources into this?

For special events which are to be counted, special CICS dummy

programs could be defined, coded, compiled, and linked. The usage count could be incremented by XCTLing into the program. However, this is not recommended for higher usage rates, because CPU cycles run for the XCTL.

We have maintained the CICS statistic for several months in the past, and run reports out of it. Our aim was the insertion of some of our non-CICS resources usage counts into the CICS statistic. These resources, and their usage count, are maintained in application or software in-core tables.

Because the usage counts reach millions, it is not feasible to increment the programs usage count by XCTL. Instead, the usage count has to be inserted as part of the resource name.

The variability of the usage count and the number of resources does not allow for the generation of predefined dummy programs and definitions for these.

CICS Version 4.1 offers the program auto-install, which allows you to build a variable program name within the CICS program name rules. Auto-install can take place without the need for the program to exist as a load module.

We built up the 8-byte program name from the following components:

```
Q RRRR K CC
```

where:

- 'Q' is a qualifier – a character that does not appear as the first character in other program names.
- 'RRRR' is the name of the resource.
- 'K' is the encryption key for the 'CC' value.
- 'CC' is the highest two digits of the resource usage count – the lower use count digits are lost.

Using this method, we inserted the usage count of some privately maintained non-CICS resource tables into the CICS statistics in the section programs. As mentioned, the original usage count of these programs is null.

The auto-install is performed in a PLT-SD program using an EXEC CICS LOAD PROGRAM (PROGNAME) command. Duplicate program names can be guarded against by using an EXEC CICS INQUIRE PROGRAM (PROGNAME) command in front of the load. Duplicates can be avoided by building the program name with a counter in the first digits that forces you to split the resource usage output to more than one program name.

Your auto-install program must support the type of program names that you build.

This is an unusual technique, but you don't need special databases, files, monitoring activities, or reports to maintain and visualize the usage values on non-CICS resources. For example, you can bring in the DB2 call-counts, obtained from the RCT-table.

Beginning with CICS transaction server, the auto-install method has an alternative in the EXEC CICS CREATE RESOURCE method.

Some lines from the CICS statistics program section are shown below:

```
PROGRAM
NAME             TIMES USED    COMMENT

OFIDE                    25    CICS RESOURCE PROGRAM
MPOFIDE                  5Ø    CICS RESOURCE MAP

..SOLUTION WITH 1 LINE PER RESOURCE FOR 4 BYTE RESOURCES:

XDBØ2C23                 Ø     NON-CICS RESOURCE DBØ2, USECOUNT 23Ø
XCDØ3B12                 Ø     NON-CICS RESOURCE CDØ3, USECOUNT  12
XFHØ8D14                 Ø     NON-CICS RESOURCE FHØ8, USECOUNT 14ØØ

..SOLUTION WITH 3 LINES PER RESOURCE FOR 8 BYTE RESOURCES:

XØØ1ABCD                 Ø     NON CICS RESOURCE ABCDEFGH,
XØØ2EFGH                 Ø      USECOUNT   18Ø
XØØ3ØC18                 Ø
XØØ4IJKL                 Ø     NON CICS RESOURCE IJKLMNOP,
XØØ5MNOP                 Ø      USECOUNT 43ØØØ
XØØ6ØE43                 Ø
```

Günther Hassold
INA Werk Schaeffer (Germany)

An MQSeries API-exit for CICS – part 2

This month we conclude the article using a modified CSQCAPX exit to answer questions relating to the processing of MQSeries messages within the boundaries of CICS.

```
         BGFILMSG MSGØØ8,I,EIBTRNID,,TASKNUM,,MSGID
         BAL   R6,CSQCAPX_WRITEMSG         MSGID
         BGFILMSG MSGØ1Ø,I,EIBTRNID,,TASKNUM,,MSGIDDUMP
         BAL   R6,CSQCAPX_WRITEMSG         MSGID DUMP
         BGFILMSG MSGØØ9,I,EIBTRNID,,TASKNUM,,CORID
         BAL   R6,CSQCAPX_WRITEMSG         CORRELID
         BGFILMSG MSGØ1Ø,I,EIBTRNID,,TASKNUM,,CORIDDUMP
         BAL   R6,CSQCAPX_WRITEMSG         CORRELID DUMP
         BGFILMSG MSGØØ6,I,EIBTRNID,,TASKNUM,,LENGTH
         BAL   R6,CSQCAPX_WRITEMSG         DATALENGTH
         BGFILMSG MSGØØ7,I,EIBTRNID,,TASKNUM,,BUFFER
         BAL   R6,CSQCAPX_WRITEMSG         DATA
         B     ENDPROG                     EXIT PROGRAM
         EJECT
*
* CHECK MQPUT PROCESSING
*
* MQPUT / MQPUT1 ARE THE SAME EXCEPT THAT WE HAVE AN OBJECT HANDLE
* FOR MQPUT CALL AND AN OBJECT DESCRIPTOR FOR MQPUT1 CALL ...
*
*
ISPUT    DS    ØH
         LA    RØ,MQXC_MQPUT               LOAD
         C     RØ,MQXP_EXITCOMMAND         IS IT MQPUT?
         BNE   ISPUT1                      NO .. TRY MQPUT1
         MVC   OPCODE,OP_PUT               SET CHARACTER OPCODE
         LA    R2,8                        OFFSET TO HOBJ IN PARMS
         BAL   R6,GETOBJECTHANDLE          GET OBJECT HANDLE
         MVI   MQPUT,TRUE                  SET TRUE FOR PUT
         B     JOIN_PUTPROCESSING          CONTINUE MQPUT / MQPUT1
*
* CHECK MQPUT1 PROCESSING
*
ISPUT1   DS    ØH
         LA    RØ,MQXC_MQPUT1              LOAD
         C     RØ,MQXP_EXITCOMMAND         IS IT MQPUT1?
         BNE   ISINQ                       NO .. TRY MQINQ
         MVC   OPCODE,OP_PUT1
         LA    R2,8                        OFFSET TO OBJDESC IN PARMS
         BAL   R6,GETOBJECTNAME            GET OBJECTNAME
```

```
         MVI   MQPUT,FALSE                 SET FALSE FOR PUT
         B     JOIN_PUTPROCESSING          CONTINUE MQPUT / MQPUT1
*
* HERE WE COME TOGEHTER FROM MQPUT AND MQPUT1
*
JOIN_PUTPROCESSING DS ØH
         CLI   BEFORECALL,TRUE             BEFORE MQPUT?
         BNE   ISPUT_AFTER                 NO, AFTER-MQPUT PROCESSING
*
* BEFORE MQPUT / MQPUT1
*
         CLI   MQPUT,TRUE                  DO WE COME FROM MQPUT
         BNE   BEFORE_MQPUT1               YES, MUST BE MQPUT1
* PUT OBJECT HANDLE INTO LOG MESSAGE FOR MQPUT
         BGFILMSG MSGØØ2,I,EIBTRNID,,TASKNUM,,OPCODE,,HOBJ
         B     JOIN_BEFOREPUT              AND CONTINUE
* PUT OBJECTNAME INTO LOG MESSAGE FOR MQPUT1
BEFORE_MQPUT1 DS ØH
         BGFILMSG MSGØØ2,I,EIBTRNID,,TASKNUM,,OPCODE,,OBJECTNAME
JOIN_BEFOREPUT DS ØH                       JOIN AGAIN
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
* COLLECT DATA
         LA    R2,2Ø                       OFFSET TO DATALENGTH
         BAL   R6,GETDATALENGTH            GET DATALENGTH BEFORE MQPUT
         LA    R2,24                       OFFSET TO BUFFER
         BAL   R6,GETDATA                  GET DATA BEFORE MQPUT
* WRITE LOG MESSAGES
         BGFILMSG MSGØØ6,I,EIBTRNID,,TASKNUM,,LENGTH
         BAL   R6,CSQCAPX_WRITEMSG         LENGTH MESSAGE
         BGFILMSG MSGØØ7,I,EIBTRNID,,TASKNUM,,BUFFER
         BAL   R6,CSQCAPX_WRITEMSG         DATA   MESSAGE
         B     ENDPROG                     EXIT PROGRAM
*
* AFTER  MQPUT / MQPUT1
*
ISPUT_AFTER DS ØH
         BAL   R6,GETRESULTCODES           GET COMPCODE AND REASON
         CLI   MQPUT,TRUE                  DO WE COME FROM MQPUT?
         BNE   AFTER_MQPUT1                NO, MUST BE MQPUT1
* PUT OBJECT HANDLE INTO LOG MESSAGE FOR MQPUT
         BGFILMSG MSGØØ3,I,EIBTRNID,,TASKNUM,,OPCODE,,CCC,,RCC,,HOBJ
         B     JOIN_AFTERPUT               AND CONTINUE
* PUT OBJECT NAME INTO LOG MESSAGE FOR MQPUT1
AFTER_MQPUT1 DS ØH
         BGFILMSG MSGØØ3,I,EIBTRNID,,TASKNUM,,OPCODE,,CCC,,RCC,,OBJECTN*
               AME                         SORRY FOR LINE WRAP
JOIN_AFTERPUT DS ØH                        JOIN AGAIN
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
         BAL   R7,GETCHARACTERRC           RC IN CHARACTER IF NEEDED
```

```
* CHECK IF DATA WAS WRITTEN
         CLC   CCC,=C'0000'                COMPLETIONCODE 0?
         BNE   ENDPROG                     NO, EXIT PROGRAM
* COLLECT DATA
         LA    R2,12                       OFFSET TO MSGDESCRIPTOR
         BAL   R6,GETMSGIDCORID            GET MSGID, CORID AFTER MQGET
* SEND MESSAGES
         BGFILMSG MSG008,I,EIBTRNID,,TASKNUM,,MSGID
         BAL   R6,CSQCAPX_WRITEMSG         MSGID
         BGFILMSG MSG010,I,EIBTRNID,,TASKNUM,,MSGIDDUMP
         BAL   R6,CSQCAPX_WRITEMSG         MSGID DUMP
         BGFILMSG MSG009,I,EIBTRNID,,TASKNUM,,CORID
         BAL   R6,CSQCAPX_WRITEMSG         CORRELID
         BGFILMSG MSG010,I,EIBTRNID,,TASKNUM,,CORIDDUMP
         BAL   R6,CSQCAPX_WRITEMSG         CORRELID DUMP
         B     ENDPROG                     EXIT PROGRAM
         EJECT
*
* MQINQ PROCESSING
*
ISINQ    DS    0H
         LA    R0,MQXC_MQINQ               LOAD
         C     R0,MQXP_EXITCOMMAND         IS IT MQINQ?
         BNE   ISSET                       NO .. TRY MQSET
         MVC   OPCODE,OP_INQ               SET CHARACTER OPCODE
         LA    R2,8                        OFFSET TO HOBJ IN PARMS
         BAL   R6,GETOBJECTHANDLE          GET OBJECT HANDLE
         CLI   BEFORECALL,TRUE             BEFORE MQINQ?
         BNE   ISINQ_AFTER                 NO, AFTER-MQINQ PROCESSING
*
* BEFORE MQINQ
*
         BGFILMSG MSG002,I,EIBTRNID,,TASKNUM,,OPCODE,,HOBJ
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
         B     ENDPROG                     EXIT PROGRAM
*
* AFTER  MQINQ
*
ISINQ_AFTER DS 0H
         BAL   R6,GETRESULTCODES           GET COMPCODE AND REASON
         BGFILMSG MSG003,I,EIBTRNID,,TASKNUM,,OPCODE,,CCC,,RCC,,HOBJ
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
         BAL   R7,GETCHARACTERRC           RC IN CHARACTER IF NEEDED
         B     ENDPROG                     EXIT PROGRAM
         EJECT
*
* MQSET PROCESSING
*
```

```
ISSET    DS    ØH
         LA    RØ,MQXC_MQSET               LOAD
         C     RØ,MQXP_EXITCOMMAND         IS IT MQSET?
         BNE   OP_UNKWN                    NO .. UNKNOWN OK-CODE
         MVC   OPCODE,OP_SET               SET CHARACTER OPCODE
         LA    R2,8                        OFFSET TO HOBJ IN PARMS
         BAL   R6,GETOBJECTHANDLE          GET OBJECT HANDLE
         CLI   BEFORECALL,TRUE             BEFORE MQSET?
         BNE   ISSET_AFTER                 NO, AFTER-MQSET PROCESSING
*
* BEFORE MQSET
*
         BGFILMSG MSGØØ2,I,EIBTRNID,,TASKNUM,,OPCODE,,HOBJ
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
         B     ENDPROG                     EXIT PROGRAM
*
* AFTER  MQSET
*
ISSET_AFTER DS ØH
         BAL   R6,GETRESULTCODES           GET COMPCODE AND REASON
         BGFILMSG MSGØØ3,I,EIBTRNID,,TASKNUM,,OPCODE,,CCC,,RCC,,HOBJ
         BAL   R6,CSQCAPX_WRITEMSG         WRITE LOG MESSAGE
         BAL   R7,GETCHARACTERRC           RC IN CHARACTER IF NEEDED
         B     ENDPROG                     EXIT PROGRAM
         EJECT
*
* UNKNOWN OP-CODE
*
OP_UNKWN DS    ØH
         L     RØ,MQXP_EXITCOMMAND         LOAD EXIT COMMAND
         CVD   RØ,WRKDWORD                 CONVERT TO PACKED DECIMAL
         UNPK  WORK1(8),WRKDWORD+4(4)      CONVERT TO ZONED DECIMAL
         MVZ   WORK1+7(1),WORK1+6          MAKE IT DISPLAYABLE
         BGFILMSG MSGØØ4,C,EIBTRNID,,TASKNUM,,WORK1
         BAL   R6,CSQCAPX_WRITEMSG         SEND MESSAGE
         B     ENDPROG                     EXIT PROGRAM
         EJECT
*
* RETURN TO CALLING PROGRAM
*
ENDPROG  DS    ØH
         EXEC  CICS RETURN
         EJECT
*********************************************************************
* SUBROUTINES
*
* PLEASE NOTICE : OFFSET TO PARM IS PARMNUMBER * 4 AND HAS TO BE
*                 PASSED IN REGISTER 2 BY CALLER
*
```

```
*********************************************************************
*
* GET COMPCODE AND RESULTCODE AFTER MQ CALL
*
GETRESULTCODES DS ØH
* GET COMPCODE FROM PARAMETER LIST (NUMBER OF PARMS-1)
         L     R2,MQXP_EXITPARMCOUNT       LOAD NUMBER OF PARAMETERS
         BCTR  R2,Ø                        REDUCE COUNT BY 1
         L     R3,COMPTR                   LOAD START OF CALL PARMLIST
         SLL   R2,2                        MULTIPLY PARMS-1 BY 4
         L     R4,Ø(R2,R3)                 TO GET OFFSET OF COMPCODE
         L     RØ,Ø(Ø,R4)                  LOAD COMPCODE
         CVD   RØ,WRKDWORD                 CONVERT TO PACKED DECIMAL
         UNPK  WORKFLD1,WRKDWORD+4(4)      CONVERT TO ZONED DECIMAL
         MVZ   WORKFLD1+7(1),WORKFLD1+6    MAKE IT DISPLAYABLE
         MVC   CCC(4),WORKFLD1+4           SAVE VALUE FOR MESSAGE
* GET REASON FROM PARAMETER LIST (LAST PARM)
         L     R2,MQXP_EXITPARMCOUNT       LOAD NUMBER OF PARAMETERS
         L     R3,COMPTR                   LOAD START OF CALL PARMLIST
         SLL   R2,2                        MULTIPLY PARMS BY 4
         L     R4,Ø(R2,R3)                 TO GET OFFSET OF REASON
         L     RØ,Ø(Ø,R4)                  LOAD REASON
         CVD   RØ,WRKDWORD                 CONVERT TO PACKED DECIMAL
         UNPK  WORKFLD1,WRKDWORD+4(4)      CONVERT TO ZONED DECIMAL
         MVZ   WORKFLD1+7(1),WORKFLD1+6    MAKE IT DISPLAYABLE
         MVC   RCC(4),WORKFLD1+4           SAVE VALUE FOR MESSAGE
         BR    R6                          RETURN TO CALLER
         EJECT
*
* GET RESULTCODE TEXT FROM RC_TABLE (IF RC NOT ØØØØ)
*
GETCHARACTERRC  DS ØH
         CLC   RCC,=CL4'ØØØØ'              RC ZERO?
         BER   R7                          YES, RETURN TO CALLER
*
         LA    R6,RC_TABLE                 ADDRESS RC TABLE
GETCRC_LOOP DS ØH
         CLC   28(4,R6),=CL4'FFFF'         END OF TABLE?
         BE    GETCRC_MOVE_TEXT            YES, MOVE TEXT
         CLC   28(4,R6),RCC                IS THIS OUR RC?
         BE    GETCRC_MOVE_TEXT            YES, MOVE TEXT
         LA    R6,32(R6)                   POINT TO NEXT TABLE ENTRY
         B     GETCRC_LOOP                 AND TRY AGAIN
*
GETCRC_MOVE_TEXT DS ØH                     BUILD LOG MSG WITH RC TEXT
         BGFILMSG MSGØ11,I,EIBTRNID,,TASKNUM,,Ø(R6),28
         BAL   R6,CSQCAPX_WRITEMSG         AND WRITE LOG MESSAGE
         BR    R7                          RETURN TO CALLER
         EJECT
```

```
*
* GET OBJECTNAME FROM PARAMETER LIST (MQOPEN, MQPUT1)
*
GETOBJECTNAME DS ØH
         L     R3,COMPTR                  LOAD START OF CALL PARMLIST
         L     R4,Ø(R2,R3)                OFFSET TO OBJDESCR
         LA    R4,12(R4)                  OBJECT NAME IS AT OFFSET 12
         MVC   OBJECTNAME(48),Ø(R4)       MOVE OBJECT NAME
         BR    R6                         RETURN TO CALLER
         EJECT
*
* GET OBJECT HANDLE FROM CALL...
*
GETOBJECTHANDLE DS ØH
         L     R3,COMPTR                  LOAD START OF CALL PARMLIST
         L     R4,Ø(R2,R3)                OFFSET TO HOBJ
         MVC   WORKFLD1(4),Ø(R4)          MOVE HANDLE
         UNPK  HOBJ(9),WORKFLD1(5)        UNPACK FOR DUMP FORMAT
         TR    HOBJ(8),HEXTAB             CONVERT TO DUMP FORMAT
         BR    R6                         RETURN TO CALLER
         EJECT
*
* GET MSGID, CORID FROM CALL
*
GETMSGIDCORID DS ØH
         L     R3,COMPTR                  LOAD START OF CALL PARMLIST
         L     R4,Ø(R2,R3)                OFFSET TO MSGDESC
         LA    R4,48(R4)                  POINT TO MESSAGE-ID
         MVC   MSGID(24),Ø(R4)            SAVE MESSAGE-ID
         LA    R4,24(R4)                  POINT TO CORREL-ID
         MVC   CORID(24),Ø(R4)            SAVE CORREL-ID
         UNPK  MSGIDDUMP(9),MSGID(5)      UNPACK FOR DUMP FORMAT
         UNPK  MSGIDDUMP+8(9),MSGID+4(5)
         UNPK  MSGIDDUMP+16(9),MSGID+8(5)
         UNPK  MSGIDDUMP+24(9),MSGID+12(5)
         UNPK  MSGIDDUMP+32(9),MSGID+16(5)
         UNPK  MSGIDDUMP+4Ø(9),MSGID+2Ø(5)
         TR    MSGIDDUMP(48),HEXTAB       CONVERT TO DUMP FORMAT
         UNPK  CORIDDUMP(9),CORID(5)      UNPACK FOR DUMP FORMAT
         UNPK  CORIDDUMP+8(9),CORID+4(5)
         UNPK  CORIDDUMP+16(9),CORID+8(5)
         UNPK  CORIDDUMP+24(9),CORID+12(5)
         UNPK  CORIDDUMP+32(9),CORID+16(5)
         UNPK  CORIDDUMP+4Ø(9),CORID+2Ø(5)
         TR    CORIDDUMP(48),HEXTAB       CONVERT TO DUMP FORMAT
         BR    R6                         RETURN TO CALLER
         EJECT
*
* GET LENGTH OF DATA, EITHER MQGET OR MQPUT/PUT1 CALLS
```

```
*
GETDATALENGTH DS ØH
         L     R3,COMPTR                  LOAD START OF CALL PARMLIST
         L     R4,Ø(R2,R3)                OFFSET TO LENGTH
         L     R2,Ø(R4)                   R2 CONTAINS LENGTH NOW
         ST    R2,LENSAVE                 SAVE FOR DATA MOVE
* MAKE IT DISPLAYABLE
         CVD   R2,WRKDWORD                CONVERT TO PACKED DECIMAL
         UNPK  WORKFLD1,WRKDWORD+4(4)     CONVERT TO ZONED DECIMAL
         MVZ   WORKFLD1+7(1),WORKFLD1+6   MAKE IT DISPLAYABLE
         MVC   LENGTH,WORKFLD1            SAVE FOR MESSAGE
         BR    R6                         RETURN TO CALLER
         EJECT
*
* GET DATA BEFORE MQPUT/PUT1 OR AFTER MQGET CALL
*
GETDATA DS ØH
         L     R3,COMPTR                  LOAD START OF CALL PARMLIST
         L     R4,Ø(R2,R3)                OFFSET TO BUFFER
         L     R2,LENSAVE                 GET SAVED LENGTH
         C     R2,=F'6Ø'                   > 6Ø ?
         BH    GETBIGGER6Ø                YES, MOVE 6Ø BYTES
* LENGTH IS SMALLER THAN 6Ø, USE LENGTH TO MOVE.
         MVI   BUFFER,C' '                CLEAR BUFFER
         MVC   BUFFER+1(L'BUFFER-1),BUFFER
         LR    R3,R2                      LENGTH TO R3
         LA    R2,BUFFER                  ADDRESS RECEIVING STORAGE
         LR    R5,R3                      LENGTH TO R5
         MVCL  R2,R4                      MOVE DATA
         BR    R6                         RETURN TO CALLER
* LENGTH IS BIGGER THAN 6Ø, MOVE 6Ø BYTES
GETBIGGER6Ø DS ØH
         MVC   BUFFER(6Ø),Ø(R4)           MOVE 6Ø BYTES
         BR    R6                         RETURN TO CALLER
         EJECT
*
* WRITE LOG MESSAGE TO CICS QUEUE
*
CSQCAPX_WRITEMSG  DS  ØH
*
         EXEC  CICS WRITEQ TD QUEUE(TDQNAME)                          *
                    FROM(BM_MSG)                                      *
                    LENGTH(TDQLEN) NOHANDLE
         BR    R6                         RETURN TO CALLER
         EJECT
*-----------------------------------*
* CONSTANTS, EQUATES & MESSAGES                                  *
*-----------------------------------*
* MESSAGES FOR USE WITH BGFILMSG MACRO
```

```
MSGØØ1 DC C'ØØ1%... %....... - NO COMMAREA PASSED TO EXIT'
MSGØØ2 DC C'ØØ2%... %....... BEFORE %....... % '
MSGØØ3 DC C'ØØ3%... %....... AFTER  %....... CC: %... RC: %... HOBJ: %'
MSGØØ4 DC C'ØØ4%... %....... - INVALID EXITCOMMAND %........'
MSGØØ5 DC C'ØØ5%... %....... - INVALID EXITREASON  %........'
MSGØØ6 DC C'ØØ6%... %....... LEN  : %....... '
MSGØØ7 DC C'ØØ7%... %....... DATA : %6ØBYTES '
MSGØØ8 DC C'ØØ8%... %....... MSGID: %24BYTES'
MSGØØ9 DC C'ØØ9%... %....... CORID: %24BYTES'
MSGØ1Ø DC C'Ø1Ø%... %.......      : %48BYTES'
MSGØ11 DC C'Ø11%... %....... RC IS: %28BYTES'
* CHARACTER COMMAND CODES
OP_OPEN  DC  CLØ8'MQOPEN '                MQI CALL
OP_CLOSE DC  CLØ8'MQCLOSE'                MQI CALL
OP_GET   DC  CLØ8'MQGET  '                MQI CALL
OP_PUT   DC  CLØ8'MQPUT  '                MQI CALL
OP_PUT1  DC  CLØ8'MQPUT1 '                MQI CALL
OP_INQ   DC  CLØ8'MQINQ  '                MQI CALL
OP_SET   DC  CLØ8'MQSET  '                MQI CALL
         DS  ØF
BM_TRTAB DC  XL256'ØØ'                    TRANSLATE TABLE FOR BGFILMSG
         ORG BM_TRTAB+C'%'
         DC  C'%'
         ORG
HEXTAB   EQU *-C'Ø'                       TRANSLATE TABLE FOR DUMP
         DC  C'Ø123456789ABCDEF'           (NOT WITHIN FIRST 24Ø BYTES
         DS  ØF                             OF SECTION)
TDQLEN   DC  H'128'                       LENGTH FOR WRITEQ TD
TDQNAME  DC  CL4'MØØ1'                    TD QUEUENAME FOR MESSAGES
TRUE     EQU C'1'                         BETTER TO READ....
FALSE    EQU C'Ø'
*
         CMQA LIST=NO                     MQI CONSTANTS
*
* TABLE WITH MQ REASON CODES, TAKEN FROM CMQA IN SCSQMACS
*
RC_TABLE DS ØF
*
 DC CL28'ALIAS_BASE_Q_TYPE_ERROR     ',CL4'2ØØ1'
 DC CL28'ALREADY_CONNECTED           ',CL4'2ØØ2'
 DC CL28'BACKED_OUT                  ',CL4'2ØØ3'
 DC CL28'BUFFER_ERROR                ',CL4'2ØØ4'
 DC CL28'BUFFER_LENGTH_ERROR         ',CL4'2ØØ5'
 DC CL28'CHAR_ATTR_LENGTH_ERROR      ',CL4'2ØØ6'
 DC CL28'CHAR_ATTRS_ERROR            ',CL4'2ØØ7'
 DC CL28'CHAR_ATTRS_TOO_SHORT        ',CL4'2ØØ8'
 DC CL28'CONNECTION_BROKEN           ',CL4'2ØØ9'
 DC CL28'DATA_LENGTH_ERROR           ',CL4'2Ø1Ø'
 DC CL28'DYNAMIC_Q_NAME_ERROR        ',CL4'2Ø11'
```

```
DC CL28'ENVIRONMENT_ERROR           ',CL4'2012'
DC CL28'EXPIRY_ERROR                ',CL4'2013'
DC CL28'FEEDBACK_ERROR              ',CL4'2014'
DC CL28'GET_INHIBITED               ',CL4'2016'
DC CL28'HANDLE_NOT_AVAILABLE        ',CL4'2017'
DC CL28'HCONN_ERROR                 ',CL4'2018'
DC CL28'HOBJ_ERROR                  ',CL4'2019'
DC CL28'INHIBIT_VALUE_ERROR         ',CL4'2020'
DC CL28'INT_ATTRCCOUNT_ERROR        ',CL4'2021'
DC CL28'INT_ATTRCCOUNT_TOO_SMALL    ',CL4'2022'
DC CL28'INT_ATTRS_ARRAY_ERROR       ',CL4'2023'
DC CL28'SYNCPOINT_LIMIT_REACHED     ',CL4'2024'
DC CL28'MAX_CONNS_LIMIT_REACHED     ',CL4'2025'
DC CL28'MD_ERROR                    ',CL4'2026'
DC CL28'MISSING_REPLY_TO_Q          ',CL4'2027'
DC CL28'MSG_TYPE_ERROR              ',CL4'2029'
DC CL28'MSG_TOO_BIG_FOR_Q           ',CL4'2030'
DC CL28'MSG_TOO_BIG_FOR_Q_MGR       ',CL4'2031'
DC CL28'NO_MSG_AVAILABLE            ',CL4'2033'
DC CL28'NO_MSG_UNDERCCURSOR         ',CL4'2034'
DC CL28'NOT_AUTHORIZED              ',CL4'2035'
DC CL28'NOT_OPEN_FOR_BROWSE         ',CL4'2036'
DC CL28'NOT_OPEN_FOR_INPUT          ',CL4'2037'
DC CL28'NOT_OPEN_FOR_INQUIRE        ',CL4'2038'
DC CL28'NOT_OPEN_FOR_OUTPUT         ',CL4'2039'
DC CL28'NOT_OPEN_FOR_SET            ',CL4'2040'
DC CL28'OBJECT_CHANGED              ',CL4'2041'
DC CL28'OBJECT_IN_USE               ',CL4'2042'
DC CL28'OBJECT_TYPE_ERROR           ',CL4'2043'
DC CL28'OD_ERROR                    ',CL4'2044'
DC CL28'OPTION_NOT_VALID_FOR_TYPE   ',CL4'2045'
DC CL28'OPTIONS_ERROR               ',CL4'2046'
DC CL28'PERSISTENCE_ERROR           ',CL4'2047'
DC CL28'PERSISTENT_NOT_ALLOWED      ',CL4'2048'
DC CL28'PRIORITY_EXCEEDS_MAXIMUM    ',CL4'2049'
DC CL28'PRIORITY_ERROR              ',CL4'2050'
DC CL28'PUT_INHIBITED               ',CL4'2051'
DC CL28'Q_DELETED                   ',CL4'2052'
DC CL28'Q_FULL                      ',CL4'2053'
DC CL28'Q_NOT_EMPTY                 ',CL4'2055'
DC CL28'Q_SPACE_NOT_AVAILABLE       ',CL4'2056'
DC CL28'Q_TYPE_ERROR                ',CL4'2057'
DC CL28'Q_MGR_NAME_ERROR            ',CL4'2058'
DC CL28'Q_MGR_NOT_AVAILABLE         ',CL4'2059'
DC CL28'REPORT_OPTIONS_ERROR        ',CL4'2061'
DC CL28'SECOND_MARK_NOT_ALLOWED     ',CL4'2062'
DC CL28'SECURITY_ERROR              ',CL4'2063'
DC CL28'SELECTORCCOUNT_ERROR        ',CL4'2065'
DC CL28'SELECTOR_LIMIT_EXCEEDED     ',CL4'2066'
```

```
DC CL28'SELECTOR_ERROR              ',CL4'2067'
DC CL28'SELECTOR_NOT_FOR_TYPE       ',CL4'2068'
DC CL28'SIGNAL_OUTSTANDING          ',CL4'2069'
DC CL28'SIGNAL_REQUEST_ACCEPTED     ',CL4'2070'
DC CL28'STORAGE_NOT_AVAILABLE       ',CL4'2071'
DC CL28'SYNCPOINT_NOT_AVAILABLE     ',CL4'2072'
DC CL28'TRIGGERCCONTROL_ERROR       ',CL4'2075'
DC CL28'TRIGGER_DEPTH_ERROR         ',CL4'2076'
DC CL28'TRIGGER_MSG_PRIORITY_ERR    ',CL4'2077'
DC CL28'TRIGGER_TYPE_ERROR          ',CL4'2078'
DC CL28'TRUNCATED_MSG_ACCEPTED      ',CL4'2079'
DC CL28'TRUNCATED_MSG_FAILED        ',CL4'2080'
DC CL28'UNKNOWN_ALIAS_BASE_Q        ',CL4'2082'
DC CL28'UNKNOWN_OBJECT_NAME         ',CL4'2085'
DC CL28'UNKNOWN_OBJECT_Q_MGR        ',CL4'2086'
DC CL28'UNKNOWN_REMOTE_Q_MGR        ',CL4'2087'
DC CL28'WAIT_INTERVAL_ERROR         ',CL4'2090'
DC CL28'XMIT_Q_TYPE_ERROR           ',CL4'2091'
DC CL28'XMIT_Q_USAGE_ERROR          ',CL4'2092'
DC CL28'NOT_OPEN_FOR_PASS_ALL       ',CL4'2093'
DC CL28'NOT_OPEN_FOR_PASS_IDENT     ',CL4'2094'
DC CL28'NOT_OPEN_FOR_SET_ALL        ',CL4'2095'
DC CL28'NOT_OPEN_FOR_SET_IDENT      ',CL4'2096'
DC CL28'CONTEXT_HANDLE_ERROR        ',CL4'2097'
DC CL28'CONTEXT_NOT_AVAILABLE       ',CL4'2098'
DC CL28'SIGNAL1_ERROR               ',CL4'2099'
DC CL28'OBJECT_ALREADY_EXISTS       ',CL4'2100'
DC CL28'OBJECT_DAMAGED              ',CL4'2101'
DC CL28'RESOURCE_PROBLEM            ',CL4'2102'
DC CL28'ANOTHER_Q_MGRCCONNECTED     ',CL4'2103'
DC CL28'UNKNOWN_REPORT_OPTION       ',CL4'2104'
DC CL28'STORAGE_CLASS_ERROR         ',CL4'2105'
DC CL28'COD_NOT_VALID_FOR_XCF_Q     ',CL4'2106'
DC CL28'XWAIT_CANCELED              ',CL4'2107'
DC CL28'XWAIT_ERROR                 ',CL4'2108'
DC CL28'SUPPRESSED_BY_EXIT          ',CL4'2109'
DC CL28'FORMAT_ERROR                ',CL4'2110'
DC CL28'SOURCE_CCSID_ERROR          ',CL4'2111'
DC CL28'SOURCE_INTEGER_ENC_ERROR    ',CL4'2112'
DC CL28'SOURCE_DECIMAL_ENC_ERROR    ',CL4'2113'
DC CL28'SOURCE_FLOAT_ENC_ERROR      ',CL4'2114'
DC CL28'TARGET_CCSID_ERROR          ',CL4'2115'
DC CL28'TARGET_INTEGER_ENC_ERROR    ',CL4'2116'
DC CL28'TARGET_DECIMAL_ENC_ERROR    ',CL4'2117'
DC CL28'TARGET_FLOAT_ENC_ERROR      ',CL4'2118'
DC CL28'NOT_CONVERTED               ',CL4'2119'
DC CL28'CONVERTED_MSG_TOO_BIG       ',CL4'2120'
DC CL28'BRIDGE_STARTED              ',CL4'2125'
DC CL28'BRIDGE_STOPPED              ',CL4'2126'
```

```
DC CL28'ADAPTER_STORAGE_SHORTAGE    ',CL4'2127'
DC CL28'ADAPTERCCONN_LOAD_ERROR     ',CL4'2129'
DC CL28'ADAPTER_SERV_LOAD_ERROR     ',CL4'213Ø'
DC CL28'ADAPTER_DEFS_ERROR          ',CL4'2131'
DC CL28'ADAPTER_DEFS_LOAD_ERROR     ',CL4'2132'
DC CL28'ADAPTERCCONV_LOAD_ERROR     ',CL4'2133'
DC CL28'ADAPTER_DISC_LOAD_ERROR     ',CL4'2138'
DC CL28'CICS_WAIT_FAILED            ',CL4'214Ø'
DC CL28'SOURCE_LENGTH_ERROR         ',CL4'2143'
DC CL28'TARGET_LENGTH_ERROR         ',CL4'2144'
DC CL28'SOURCE_BUFFER_ERROR         ',CL4'2145'
DC CL28'TARGET_BUFFER_ERROR         ',CL4'2146'
DC CL28'DBCS_ERROR                  ',CL4'215Ø'
DC CL28'TRUNCATED                   ',CL4'2151'
DC CL28'ASID_MISMATCH               ',CL4'2157'
DC CL28'CONN_ID_IN_USE              ',CL4'216Ø'
DC CL28'Q_MGR_QUIESCING             ',CL4'2161'
DC CL28'Q_MGR_STOPPING              ',CL4'2162'
DC CL28'DUPLICATE_RECOV_COORD       ',CL4'2163'
DC CL28'PMO_ERROR                   ',CL4'2173'
DC CL28'API_EXIT_NOT_FOUND          ',CL4'2182'
DC CL28'API_EXIT_LOAD_ERROR         ',CL4'2183'
DC CL28'REMOTE_Q_NAME_ERROR         ',CL4'2184'
DC CL28'GMO_ERROR                   ',CL4'2186'
DC CL28'PAGESET_FULL                ',CL4'2192'
DC CL28'PAGESET_ERROR               ',CL4'2193'
DC CL28'NAME_NOT_VALID_FOR_TYPE     ',CL4'2194'
DC CL28'UNEXPECTED_ERROR            ',CL4'2195'
DC CL28'UNKNOWN_XMIT_Q              ',CL4'2196'
DC CL28'UNKNOWN_DEF_XMIT_Q          ',CL4'2197'
DC CL28'DEF_XMIT_Q_TYPE_ERROR       ',CL4'2198'
DC CL28'DEF_XMIT_Q_USAGE_ERROR      ',CL4'2199'
DC CL28'NAME_IN_USE                 ',CL4'22Ø1'
DC CL28'CONNECTION_QUIESCING        ',CL4'22Ø2'
DC CL28'CONNECTION_STOPPING         ',CL4'22Ø3'
DC CL28'ADAPTER_NOT_AVAILABLE       ',CL4'22Ø4'
DC CL28'MSG_ID_ERROR                ',CL4'22Ø6'
DC CL28'CORREL_ID_ERROR             ',CL4'22Ø7'
DC CL28'FILE_SYSTEM_ERROR           ',CL4'22Ø8'
DC CL28'NO_MSG_LOCKED               ',CL4'22Ø9'
DC CL28'FILE_NOT_AUDITED            ',CL4'2216'
DC CL28'CONNECTION_NOT_AUTHORIZED   ',CL4'2217'
DC CL28'MSG_TOO_BIG_FORCCHANNEL     ',CL4'2218'
DC CL28'CALL_IN_PROGRESS            ',CL4'2219'
DC CL28'Q_MGR_ACTIVE                ',CL4'2222'
DC CL28'Q_MGR_NOT_ACTIVE            ',CL4'2223'
DC CL28'Q_DEPTH_HIGH                ',CL4'2224'
DC CL28'Q_DEPTH_LOW                 ',CL4'2225'
```

```
 DC CL28'Q_SERVICE_INTERVAL_HIGH     ',CL4'2226'
 DC CL28'Q_SERVICE_INTERVAL_OK       ',CL4'2227'
 DC CL28'HCONFIG_ERROR               ',CL4'228Ø'
 DC CL28'FUNCTION_ERROR              ',CL4'2281'
 DC CL28'CHANNEL_STARTED             ',CL4'2282'
 DC CL28'CHANNEL_STOPPED             ',CL4'2283'
 DC CL28'CHANNEL_CONV_ERROR          ',CL4'2284'
 DC CL28'SERVICE_NOT_AVAILABLE       ',CL4'2285'
 DC CL28'INITIALIZATION_FAILED       ',CL4'2286'
 DC CL28'TERMINATION_FAILED          ',CL4'2287'
 DC CL28'UNKNOWN_Q_NAME              ',CL4'2288'
 DC CL28'SERVICE_ERROR               ',CL4'2289'
 DC CL28'Q_ALREADY_EXISTS            ',CL4'229Ø'
 DC CL28'USER_ID_NOT_AVAILABLE       ',CL4'2291'
 DC CL28'UNKNOWN_ENTITY              ',CL4'2292'
 DC CL28'UNKNOWN_AUTH_ENTITY         ',CL4'2293'
 DC CL28'UNKNOWN_REF_OBJECT          ',CL4'2294'
 DC CL28'CHANNEL_ACTIVATED           ',CL4'2295'
 DC CL28'CHANNEL_NOT_ACTIVATED       ',CL4'2296'
 DC CL28'?? PLEASE CHECK MSGCODES MAN',CL4'FFFF'       END OF TABLE
*
         LTORG
         END  CSQCAPX
```

Stefan Raabe
Systems Programmer
Braun AG (Germany)

Call for papers

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*?

More on macros to define statements

INTRODUCTION

This article is a continuation of *Converting macros to define statements*, published in *CICS Update* Issue 147, February 1998 and *CICS Update* Issue 148, March 1998. It provides an additional macro which eliminates all VSAM entries from an FCT.

It also contains a program that merges CSD define statements.

FCT MACRO

Note: because it processes DFHFCT macros, the FCT macro is named DFHFCT and could conflict with the DFHFCT macro from the above article. Therefore, care must be taken to save this macro with a different name from that used previously. Usage is the same as previously described except that the JCL is changed to include the current macro.

MACRO SOURCE

```
         MACRO
.*
.*  THIS MACRO PUNCHES FCT TABLES FROM EXISTING TABLES.  THE PUNCHED
.*  TABLE WILL NOT CONTAIN ANY 'ACCMETH=VSAM' ENTRIES.  THE KEYWORDS
.*  WILL BE ARRANGED IN THE FOLLOWING SEQUENCE.
.*
&NAME    DFHFCT  &TYPE=,           TYPE OF MACRO                       *
               &ACCMETH=,          ACCESS METHOD IDENTIFICATION        *
               &BASE=,             BASE SYMBOL FOR BSTRNO TABULATION   +
               &BLKKEYL=,          PHYSICAL KEY LENGTH (DEFAULT = Ø)   *
               &BUFNI=,            VSAM INDEX BUFFER NUMBER            *
               &BUFND=,            VSAM DATA BUFFER NUMBER             *
               &BLKSIZE=,          BLOCK SIZE                          *
               &BUFFERS=,          BUFFERS FOR VSAM POOL               *
               &BUFSP=,            VSAM BUFFER SPACE                   *
               &DATASET=,          NAME OF CICS FILE (SAME AS DDNAME)  *
               &FILE=,             NAME OF CICS FILE (SAME AS DDNAME)  *
               &FILSTAT=,          FILE STATUS                         *
               &GROUP=,            RDO GROUP NAME                      *
               &EXTENT=,           NUMBER OF DISK EXTENTS              *
```

```
                &LRECL=,            LOGICAL RECORD LENGTH               *
                &RKP=,              RELATIVE KEY POSITION               *
                &KEYLEN=,           KEY LENGTH OF LOGICAL RECORD        *
                &RELTYPE=,          TYPE OF RELATIVE RECORD ADDR        *
                &VERIFY=,           WRITE VERIFY OPTION                 *
                &SRCHM=,            MULTIPLE TRACK SEARCH - KEY         *
                &JID=,              JOURNAL IDENTIFICATION              *
                &JREQ=,             JOURNAL REQUESTS                    *
                &LOG=,              SYSTEM LOG INDICATOR                *
                &MIGRATE=,          RESOURCE DEFINITION ONLINE CALL     *
                &OPEN=,             OLD DEFERRED OPEN OPTION            *
                &PASSWD=,           VSAM PASSWORD                       *
                &RECFORM=,          RECORD FORMAT                       *
                &RMTNAME=,          DATASET NAME ON REMOTE SYSTEM       *
                &RSCLMT=,           RESOURCE PERCENT FOR VSAM POOL      *
                &RSL=,              RESOURCE LEVEL SECURITY             *
                &SIZE=,             DATA TABLE SIZE                     *
                &STRNO=,            VSAM MAXIMUM STRINGS                *
                &STRNOG=,           CICS 'GET ONLY' STRINGS  (OS ONLY)  *
                &SERVREQ=,          SERVICE REQUEST IDENTIFICATION      *
                &LSRPOOL=,          VSAM RESOURCE-SHARING SPECIFICATION +
                &SUFFIX=,           FILE CONTROL TABLE NAME SUFFIX      *
                &STARTER=,          PREGENERATED TABLES ONLY            *
                &SYSIDNT=,          REMOTE SYSTEM IDENTIFIER            *
                &DSNAME=,           DATA SET NAME FOR DYNAMIC ALLOCATION*
                &DSNSHR=,           DOES DSN-SHARING AFFECT R/O ACCESS? +
                &DISP=,             DISPOSITION FOR DATASET             +
                &DUMMY4=,                                               *
                &DUMMY3=,                                               *
                &DUMMY2=,                                               *
                &DUMMY1=,                                               *
                &DUMMY=             PROTOTYPE DUMMY PARAMETER@15553 @LBC
.*
        LCLA  &I,&J,&K
        LCLC  &X,&P(5Ø)
        GBLA  &FIRST
.*
        AIF   (&FIRST NE Ø).NOT1ST
        PUNCH '***  THE FOLLOWING FCT WAS PRODUCED BY ELIMINATING ALL -
              VSAM FILES'
&FIRST  SETA  1
.*
.NOT1ST AIF   ('&ACCMETH' EQ 'VSAM' OR
              '&ACCMETH' EQ '(VSAM)').END
.*
&X      SETC  '&NAME'.'         '
&X      SETC  '&X'(1,9).'DFHFCT '
.*
        AIF   (T'&TYPE EQ 'O').TYPE
&I      SETA  &I+1
```

```
&P(&I)   SETC  'TYPE=&TYPE'
.*
.TYPE    AIF   (T'&BASE EQ 'O').BASE
&I       SETA  &I+1
&P(&I)   SETC  'BASE=&BASE'
.*
.BASE    AIF   (T'&BLKKEYL EQ 'O').BLKKEYL
&I       SETA  &I+1
&P(&I)   SETC  'BLKKEYL=&BLKKEYL'
.*
.BLKKEYL AIF   (T'&BUFNI EQ 'O').BUFNI
&I       SETA  &I+1
&P(&I)   SETC  'BUFNI=&BUFNI'
.*
.BUFNI   AIF   (T'&BUFND EQ 'O').BUFND
&I       SETA  &I+1
&P(&I)   SETC  'BUFND=&BUFND'
.*
.BUFND   AIF   (T'&BLKSIZE EQ 'O').BLKSIZE
&I       SETA  &I+1
&P(&I)   SETC  'BLKSIZE=&BLKSIZE'
.*
.BLKSIZE AIF   (T'&BUFFERS EQ 'O').BUFFERS
&I       SETA  &I+1
&P(&I)   SETC  'BUFFERS=&BUFFERS'
.*
.BUFFERS AIF   (T'&BUFSP EQ 'O').BUFSP
&I       SETA  &I+1
&P(&I)   SETC  'BUFSP=&BUFSP'
.*
.BUFSP   AIF   (T'&DATASET EQ 'O').DATASET
&I       SETA  &I+1
&P(&I)   SETC  'DATASET=&DATASET'
.*
.DATASET AIF   (T'&FILE EQ 'O').FILE
&I       SETA  &I+1
&P(&I)   SETC  'FILE=&FILE'
.*
.FILE    AIF   (T'&FILSTAT EQ 'O').FILSTAT
&I       SETA  &I+1
&P(&I)   SETC  'FILSTAT=&FILSTAT'
.*
.FILSTAT AIF   (T'&GROUP EQ 'O').GROUP
&I       SETA  &I+1
&P(&I)   SETC  'GROUP=&GROUP'
.*
.GROUP   AIF   (T'&EXTENT EQ 'O').EXTENT
&I       SETA  &I+1
&P(&I)   SETC  'EXTENT=&EXTENT'
.*
```

```
.EXTENT  AIF   (T'&LRECL EQ 'O').LRECL
&I       SETA  &I+1
&P(&I)   SETC  'LRECL=&LRECL'
.*
.LRECL   AIF   (T'&RKP EQ 'O').RKP
&I       SETA  &I+1
&P(&I)   SETC  'RKP=&RKP'
.*
.RKP     AIF   (T'&KEYLEN EQ 'O').KEYLEN
&I       SETA  &I+1
&P(&I)   SETC  'KEYLEN=&KEYLEN'
.*
.KEYLEN  AIF   (T'&RELTYPE EQ 'O').RELTYPE
&I       SETA  &I+1
&P(&I)   SETC  'RELTYPE=&RELTYPE'
.*
.RELTYPE AIF   (T'&VERIFY EQ 'O').VERIFY
&I       SETA  &I+1
&P(&I)   SETC  'VERIFY=&VERIFY'
.*
.VERIFY  AIF   (T'&SRCHM EQ 'O').SRCHM
&I       SETA  &I+1
&P(&I)   SETC  'SRCHM=&SRCHM'
.*
.SRCHM   AIF   (T'&JID EQ 'O').JID
&I       SETA  &I+1
&P(&I)   SETC  'JID=&JID'
.*
.JID     AIF   (T'&JREQ EQ 'O').JREQ
&I       SETA  &I+1
&P(&I)   SETC  'JREQ=&JREQ'
.*
.JREQ    AIF   (T'&LOG EQ 'O').LOG
&I       SETA  &I+1
&P(&I)   SETC  'LOG=&LOG'
.*
.LOG     AIF   (T'&MIGRATE EQ 'O').MIGRATE
&I       SETA  &I+1
&P(&I)   SETC  'MIGRATE=&MIGRATE'
.*
.MIGRATE AIF   (T'&OPEN EQ 'O').OPEN
&I       SETA  &I+1
&P(&I)   SETC  'OPEN=&OPEN'
.*
.OPEN    AIF   (T'&PASSWD EQ 'O').PASSWD
&I       SETA  &I+1
&P(&I)   SETC  'PASSWD=&PASSWD'
.*
.PASSWD  AIF   (T'&RECFORM EQ 'O').RECFORM
&I       SETA  &I+1
```

```
&P(&I)   SETC  'RECFORM=&RECFORM'
.*
.RECFORM AIF   (T'&RMTNAME EQ 'O').RMTNAME
&I       SETA  &I+1
&P(&I)   SETC  'RMTNAME=&RMTNAME'
.*
.RMTNAME AIF   (T'&RSCLMT EQ 'O').RSCLMT
&I       SETA  &I+1
&P(&I)   SETC  'RSCLMT=&RSCLMT'
.*
.RSCLMT  AIF   (T'&RSL EQ 'O').RSL
&I       SETA  &I+1
&P(&I)   SETC  'RSL=&RSL'
.*
.RSL     AIF   (T'&SIZE EQ 'O').SIZE
&I       SETA  &I+1
&P(&I)   SETC  'SIZE=&SIZE'
.*
.SIZE    AIF   (T'&STRNO EQ 'O').STRNO
&I       SETA  &I+1
&P(&I)   SETC  'STRNO=&STRNO'
.*
.STRNO   AIF   (T'&STRNOG EQ 'O').STRNOG
&I       SETA  &I+1
&P(&I)   SETC  'STRNOG=&STRNOG'
.*
.STRNOG  AIF   (T'&SERVREQ EQ 'O').SERVREQ
&I       SETA  &I+1
&P(&I)   SETC  'SERVREQ=&SERVREQ'
.*
.SERVREQ AIF   (T'&LSRPOOL EQ 'O').LSRPOOL
&I       SETA  &I+1
&P(&I)   SETC  'LSRPOOL=&LSRPOOL'
.*
.LSRPOOL AIF   (T'&SUFFIX EQ 'O').SUFFIX
&I       SETA  &I+1
&P(&I)   SETC  'SUFFIX=&SUFFIX'
.*
.SUFFIX  AIF   (T'&STARTER EQ 'O').STARTER
&I       SETA  &I+1
&P(&I)   SETC  'STARTER=&STARTER'
.*
.STARTER AIF   (T'&SYSIDNT EQ 'O').SYSIDNT
&I       SETA  &I+1
&P(&I)   SETC  'SYSIDNT=&SYSIDNT'
.*
.SYSIDNT AIF   (T'&DSNAME EQ 'O').DSNAME
&I       SETA  &I+1
&P(&I)   SETC  'DSNAME=&DSNAME'
```

```
.*
.DSNAME  AIF   (T'&DSNSHR EQ 'O').DSNSHR
&I       SETA  &I+1
&P(&I)   SETC  'DSNSHR=&DSNSHR'
.*
.DSNSHR  AIF   (T'&DISP EQ 'O').DISP
&I       SETA  &I+1
&P(&I)   SETC  'DISP=&DISP'
.*
.DISP    AIF   (T'&DUMMY4 EQ 'O').DUMMY4
&I       SETA  &I+1
&P(&I)   SETC  'DUMMY4=&DUMMY4'
.*
.DUMMY4  AIF   (T'&DUMMY3 EQ 'O').DUMMY3
&I       SETA  &I+1
&P(&I)   SETC  'DUMMY3=&DUMMY3'
.*
.DUMMY3  AIF   (T'&DUMMY2 EQ 'O').DUMMY2
&I       SETA  &I+1
&P(&I)   SETC  'DUMMY2=&DUMMY2'
.*
.DUMMY2  AIF   (T'&DUMMY1 EQ 'O').DUMMY1
&I       SETA  &I+1
&P(&I)   SETC  'DUMMY1=&DUMMY1'
.*
.DUMMY1  AIF   (T'&DUMMY EQ 'O').DUMMY
&I       SETA  &I+1
&P(&I)   SETC  'DUMMY=&DUMMY'
.*
.DUMMY   AIF   (&I LE 1).LAST
.*
.LOOP    ANOP
&J       SETA  &J+1
.*
         AIF   (K'&X+K'&P(&J) LE 7Ø).WILLFIT
.*
&X       SETC  '&X'.(72)' '
&X       SETC  '&X'(1,71).'X'
         PUNCH '&X'
&X       SETC  (15)' '
.*
.WILLFIT AIF   (&J GE &I).LAST
.*
&X       SETC  '&X'.'&P(&J)'.','
         AGO   .LOOP
.*
.LAST    ANOP
&X       SETC  '&X'.'&P(&I)'
         PUNCH '&X'
```

```
.*
         PUNCH '*'
.*
.END     MEND
```

CSD MERGE PROGRAM

RDOMERGE merges new RDO/CSD control statements from file INPUT1 and the old statements from file INPUT2 to the OUTPUT file.

The actual processing steps are as follows:

- File INPUT2 is read until the first record that contains either ‘DEFINE’ or ‘* PROCESSED BY RDOMERGE ’ beginning in the first position of the record. This is to copy any JCL statements to the OUTPUT file. This last statement is not copied until the remainder of the file has been copied. To ensure that any comments that precede the first DEFINE statement are retained with that statement, it might be desirable to insert the above comment record prior to such comments.
- A record is added to the file to indicate when this merge occurred. Its content is ‘* PROCESSED BY RDOMERGE mm/dd/yy hh:mm:ss.th’.
- File INPUT1 is read and the ‘DEFINE’ statements are counted. This count is needed to obtain a main storage area to retain the names of entries. The file is then closed.
- INPUT1 is reopened and copied to the OUTPUT file. The type and entry name of each DEFINE statement is saved in the above main storage area.
- A record is added to the file to indicate the end of this merge. Its content is ‘* END PROCESSING BY RDOMERGE mm/dd/yy hh:mm:ss.th’.
- The remainder of INPUT2 is copied to OUTPUT. Each record is examined to determine whether it has been redefined by the newer records. If so, each of its records is shifted right one position and an asterisk (‘*’) is inserted in the first position. A

record is added to indicate which of the new statements caused these statements to be replaced by comments. This record also contains the current date and time to indicate which merge resulted in this action.

- The name is extracted from each GROUP(name) parameter and, if it does not exist, is retained in the table GROUPS. These names will be used to create ADD statements, which are inserted at the end of the OUTPUT file. These statements are also listed. The format of the ADD statements is:

  ```
  ADD GROUP(name) LIST(INITLIST)
  ```

  The list name INITLIST may be changed by altering the source at label LISTNAME.

- A summary of the above activity is produced on file PRINTER.

SAMPLE RDOMERGE JCL

```
//SYST00#2L JOB ...
//*-------------------------------------*//
//*   MERGE EXISTING RDO STATEMENTS WITH NEW RDO STATEMENTS
//*-------------------------------------*//
//S1       EXEC PGM=RDOMERGE
//STEPLIB  DD  DSN=MPAC2.MTST.LOADLIB,DISP=SHR
//SYSUDUMP DD  SYSOUT=*
//PRINTER  DD  SYSOUT=*
//INPUT2   DD  DSN=RDO.MAINT.FILE(NEWRDO),DISP=SHR
//INPUT1   DD  DSN=RDO.MAINT.FILE(OLDRDO),DISP=SHR
//OUTPUT   DD  DSN=RDO.MERGED.FILE,DISP=OLD
```

RDOMERGE PROGRAM

```
         LCLC  &MYNAME
*
&MYNAME  SETC  'RDOMERGE'               CSECT NAME
RBASE    EQU   12                       BASE REGISTER FOR CSECT
RBAL     EQU   10                       BAL REGISTER
*
         TITLE '&MYNAME'                LISTING TITLE
*********************************************************************
***                                                               ***
***   THIS PROGRAM READS TWO INPUT FILES (INPUT1 AND INPUT2).     ***
***                                                               ***
***   FIRST INPUT1 IS COPIED TO OUTPUT AND THE TYPE(ENTRY) OF     ***
```

```
***   RDO 'DEFINE TYPE(ENTRY) ...' STATEMENTS ARE SAVED.           ***
***                                                                ***
***   NEXT INPUT2 IS READ, EDITED, AND COPIED TO OUTPUT.  IF A     ***
***   DUPLICATE RDO DEFINITION TYPE IS FOUND, IT IS FLAGGED WITH   ***
***   AN ASTERISK IN COLUMN 1 (COMMENTED OUT) AND A MESSAGE IS     ***
***   INSERTED TO INDICATE IT IS REMOVED.                          ***
***                                                                ***
********************************************************************
         EJECT
********************************************************************
***                                                                ***
***      LINKAGE CONVENTIONS ENTERING PROGRAM                      ***
***                                                                ***
********************************************************************
&MYNAME  CSECT ,
         STM   R14,R12,12(R13)          SAVE REGS TO CALLER S.A.
         B     (BEGIN-&MYNAME)(R15)     BRANCH AROUND EYECATCHER
         DC    A(L'NAME)                LENGTH OF CSECT NAME
NAME     DC    C'&MYNAME'               CSECT NAME
         DC    C' &SYSDATE &SYSTIME '   ASSEMBLY DATE/TIME STAMP
BEGIN    LR    RBASE,R15                LOAD BASE REGISTER
         USING &MYNAME,RBASE              ADDRESSABILITY
         PRINT NOGEN
         GETMAIN R,LV=WORKDLEN          GET SAVE/WORK AREA
         ST    R1,8(Ø,R13)              MY S.A. ADDR INTO CALLER S.A.
         ST    R13,4(Ø,R1)              CALLER S.A. ADDR INTO MY S.A.
         LR    R13,R1                   R13 POINTS TO MY S.A.
         USING WORKD,R13                ADDRESSABILITY OF SAVE AREA
         L     R1,4(Ø,R13)              R1 POINTS TO CALLER S.A.
         LM    R15,R1,16(R1)            R15 RØ AND R1 ARE RESTORED
*
         EJECT
********************************************************************
***                                                                ***
***      MAINLINE ROUTINE                                          ***
***                                                                ***
********************************************************************
MAIN     EQU   *                        BEGIN MAINLINE ROUTINE
         ST    R1,R1SAVE                SAVE INITIAL R1
         XC    COMPCODE,COMPCODE        CLEAR COMPLETION CODE
*
         MVC   JGMOTBL(13*L'JGMOTBL),JGMOTBLD  COPY JULGREG DAYS/MONTH
*
* BEGIN DCB INITIALIZATION
*
         MVC   PRINTER(PRINTERL),PRINTERD  INITIALIZE DCB
*
         MVC   INPUT1(INPUT1L),INPUT1D   INITIALIZE INPUT1 DCB
*
         MVC   INPUT2(INPUT2L),INPUT2D   INITIALIZE INPUT2 DCB
```

```
*
         MVC   OUTPUT(OUTPUTL),OUTPUTD   INITIALIZE OUTPUT DCB
*
* END DCB INITIALIZATION
*
*
* BEGIN DCB OPENS
*
         MVC   PROPENL(PROPENLN),OPEND INITIALIZE SET PRINTER OPEN LIST
         OPEN  (PRINTER,(OUTPUT)),MF=(E,PROPENL)  OPEN PRINTER
*
         MVC   I1OPENL(I1OPENLN),OPEND   SET INPUT1 OPEN LIST
         OPEN  (INPUT1,(INPUT)),MF=(E,I1OPENL)   OPEN INPUT1
*
         MVC   I2OPENL(I2OPENLN),OPEND   SET INPUT2 OPEN LIST
         OPEN  (INPUT2,(INPUT)),MF=(E,I2OPENL)   OPEN INPUT2
*
         MVC   OPOPENL(OPOPENLN),OPEND   SET OUTPUT OPEN LIST
         OPEN  (OUTPUT,(OUTPUT)),MF=(E,OPOPENL)  OPEN OUTPUT
*
* END DCB OPENS
*
         XC    TRTAB2,TRTAB2       CLEAR ALL BYTES
         MVI   TRTAB2+C' ',X'FF'   TURN ON BLANK POSITION
*
         MVI   TRTAB1,X'FF'        SET NONZERO
         MVC   TRTAB1+1(L'TRTAB1-1),TRTAB1 SET ALL NONZERO
         MVI   TRTAB1+C' ',Ø       TURN OFF BLANK POSITION
*
         LA    R2,GROUPS-L'GROUPS  ADDERSS OF ENTRY(-1)
         ST    R2,GROUPLOC         SAVE INITIAL LOC OF GROUP TABLE END
*
         MVI   LINE,C' '           SET SEED
         MVC   LINE+1(L'LINE-1),LINE CLEAR TO BLANKS
         MVC   OUTAREA,LINE         "
*
         MVC   DDNAME,IN1DDN       SET DDNAME
         BAL   RBAL,GETNAMES       GET JOB NAME AN INPUT1 DSN
         MVC   IN1DSN,HEADDSN      SAVE
*
         MVC   DDNAME,IN2DDN       SET DDNAME
         BAL   RBAL,GETNAMES       GET JOB NAME AND INPUT2 DSN
         MVC   IN2DSN,HEADDSN      SAVE
*
         MVC   HEADER(L'HEAD),HEAD INITIALIZE HEADER
         MVC   HEADER+L'HEAD(L'HEADER-L'HEAD),HEADER+L'HEAD-1 CLEAR
         MVC   PAGENO-4(4),=C'PAGE' SET PAGE NUMBER ID
         ZAP   PAGES,=P'1'         INITIALIZE PAGE COUNT
         TIME
         ST    RØ,TIME             SAVE HH:MM:SS.TH
```

```
         ST    R1,JGYYDDD          SAVE JULIAN DATE
         BAL   RBAL,JULGREG        CONVERT JULIAN DATE TO GREGDATE
         MVC   HEADTIME,TIMEPAT    SET EDIT PATTERN
         ED    HEADTIME,TIME       FORMAT HH:MM:SS.TH
*
         MVC   DDNAME,OUTDDN       SET DDNAME
         BAL   RBAL,GETNAMES       GET JOB NAME AND OUTPUT DSN
*
         MVC   HEADDATE,JGMMDDYY   MOVE MM/DD/YY TO HEADER
         BAL   RBAL,HEADPAGE       PRINT PAGE HEADER
*
         MVI   IN2FLAG,C'*'        SET DUPLICATE FLAG
         MVI   DUPFLAG,Ø           SET TO INDICATE NON-DUPLICATE
         MVI   PASSFLAG,Ø          CLEAR PASS SWITCHES
*
         MVC   LINE+1(16),=C'INPUT FILES ARE:'
         MVC   LINE+19(L'IN1DSN),IN1DSN   SET INPUT1 DSN IN PRINT LINE
         BAL   RBAL,PRINT          GO PRINT INPUT1 DSN
*
         MVC   LINE+13(4),=C'AND:'
         MVC   LINE+19(L'IN2DSN),IN2DSN   SET INPUT2 DSN IN PRINT LINE
         BAL   RBAL,PRINT          GO PRINT INPUT2 DSN
*
         MVI   LINE,C'Ø'           SET TO DOUBLE SAPCE
         BAL   RBAL,DOUBLESP       ALLOW FOR DOUBLE SPACE
*
         ZAP   COUNT1,=P'Ø'        INITIALIZE INPUT1 RECORD COUNT
         ZAP   COUNT2,=P'Ø'        INITIALIZE INPUT2 RECORD COUNT
         ZAP   DUPS,=P'Ø'          INITIALIZE INPUT2 RECORD COUNT
*
         BAL   RBAL,COPYJCL        COPY INPUT2 TO FIRST DEFINE STATEMNT
*
         BAL   RBAL,LOGOUT         WRITE RUN INFORMATION TO OUTPUT
*
         BAL   RBAL,COUNTREC       GO READ/DOUNT DEFINES FROM INPUT1
*
         BAL   RBAL,COPYIN1        COPY INPUT TO OUTPUT
*
         BAL   RBAL,COPYIN2        PROCESS INPUT2
*
         BAL   RBAL,DOTOTALS       WRITE TOTALS
*
* BEGIN DCB CLOSE
*
         MVC   PRCLOSL(PRCLOSLN),CLOSED  INITIALIZE CLOSE LIST
         CLOSE (PRINTER),MF=(E,PRCLOSL) CLOSE IT
*
         MVC   I1CLOSL(I1CLOSLN),CLOSED  SET INPUT1 CLOSE LIST
         CLOSE (INPUT1),MF=(E,I1CLOSL)  CLOSE INPUT1
*
```

```
         MVC   I2CLOSL(I2CLOSLN),CLOSED  SET INPUT2 CLOSE LIST
         CLOSE (INPUT2),MF=(E,I2CLOSL)  CLOSE INPUT2
*
         MVC   OPCLOSL(OPCLOSLN),CLOSED  SET OUTPUT CLOSE LIST
         CLOSE (OUTPUT),MF=(E,OPCLOSL)  CLOSE OUTPUT
*
* END DCB CLOSE
*
END00    LA    R15,0                    SET COMPLETION CODE 00
         ST    R15,COMPCODE               INTO STORAGE
         B     ENDING                   GO TO ENDING
*
         EJECT
*********************************************************************
***                                                               ***
***      LINKAGE CONVENTIONS EXITING PROGRAM                      ***
***                                                               ***
*********************************************************************
ENDING   L     R14,COMPCODE             R14 SAVES COMP CODE
         LR    R1,R13                   R1 SAVES ADDR OF MY S.A.
         L     R13,4(0,R1)              R13 RESTORED, PTR CALLER S.A.
         FREEMAIN R,LV=WORKDLEN,A=(R1)  FREE MY SAVE/WORK AREA
         LR    R15,R14                  R15 SET TO COMP CODE
         LM    R0,R12,20(R13)           R0-R12 RESTORED
         L     R14,12(0,R13)            R14 RESTORED
         MVI   12(R13),X'FF'            SET COMPLETION SIGNAL
         BR    R14                      RETURN TO CALLER
*
*
* BEGIN STUB DEFINE
*
*
         EJECT
*********************************************************************
***                                                               ***
***   CONVERT JULIAN DATE TO GREGORGIAN DATE                      ***
***                                                               ***
*********************************************************************
*
JULGREG  ST    RBAL,SAVJGBAL       SAVE LINKAGE REGISTER
*
         CLI   JGYYDDD,1           IS ACTUAL CENTURY PRESENT?
         BH    JGACTUAL             YES
         TR    JGYYDDD(1),=X'1920' CENTURY=0 ==> 19XX, 1==>20XX
JGACTUAL ZAP   JGDAYS,JGYYDDD+2(2) SAVE DAYS FROM BEGINNING OF YEAR
         ZAP   JGMONTHS,=P'1'    INITIALIZE MONTH
*
         LA    R15,JANUARY       LOAD ADDRESS OF DAYS/MONTH TABLE
         LA    R0,L'JANUARY       ... WIDTH OF TABLE
         LA    R1,DECEMBER        ... END OF TABLE
```

```
*
         ZAP   FEBRUARY,=P'28'     SET NON-LEAP YEAR DAYS
         CLC   =X'2000',JGYYDDD    YEAR 2000?
         BE    JGYR2000             YES
*
JG20THCN TM    JGYYDDD+1,1         LEAP YEAR?
         BO    JGLOOP               NO
         TM    JGYYDDD+1,X'12'
         BM    JGLOOP               NO
JGYR2000 AP    FEBRUARY,=P'1'      ADJUST
*
JGLOOP   CP    JGDAYS,0(L'JANUARY,R15) CURRENT MONTH?
         BNH   JGFOUND              YES
         AP    JGMONTHS,=P'1'      INCREMENT MONTH
         SP    JGDAYS,0(L'JANUARY,R15) DECREMENT DAYS PER CURRENT MONTH
         BXLE  R15,R0,JGLOOP       CONTINUE
*
JGFOUND  UNPK  JGMMDDYY(2),JGMONTHS  UNPACK MONTH
         UNPK  JGMMDDYY+3(2),JGDAYS  UNPACK DAY
         UNPK  JGMMDDYY+6(3),JGYYDDD+1(2) UNPACK YEAR
         MVI   JGMMDDYY+2,C'/'     SEPARATE MONTH AND DAY
         MVI   JGMMDDYY+5,C'/'     SEPARATE DAY AND YEAR
         OI    JGMMDDYY+1,C'0'     FORCE MONTH NUMERIC
         OI    JGMMDDYY+4,C'0'     FORCE DAY NUMERIC
         OI    JGMMDDYY+7,C'0'     FORCE YEAR NUMERIC
*
JGRETURN L     RBAL,SAVJGBAL       LOAD LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE GETS CURRENT JOB NAME AND DSN FOR DDNAME.      ***
***                                                               ***
*********************************************************************
*
GETNAMES ST    RBAL,SAVGNBAL       SAVE LINKAGE REGISTER
*
         XR    R15,R15             ADDRESS OF PSA
         USING PSA,R15             ESTABLISH ADDRESSABILITY
         L     R14,FLCCVT          ADDRESS OF CVT
         DROP  R15                 DROP ADDRESSABILITY TO PSA
         USING CVTMAP,R14          ESTABLISH ADDRESSABILITY TO CVT
         L     R15,CVTTCBP         ADDRESS OF NEXT TCB POINTER
         L     R15,4(0,R15)        ADDRESS OF CURRENT TCB
         DROP  R14                 DROP ADDRESSABILITY TO CVT
         USING TCB,R15             ESTABLISH ADDRESSABILITY CURRENT TCB
         L     R14,TCBTIO          ADDRESS OF TIOT
         USING TIOT,R14            ESTABLISH ADDRESSABILITY TO TIOT
         MVC   HEADJOBN,TIOCNJOB   MOVE JOB NAME TO HEADER
```

```
         MVC   HEADJOBN-4(4),=C'JOB='        SET JOBNAME ID
*
         DROP  R15                 DROP ADDRESSABILITY TO TCB
         LA    R15,TIOELNGH        ADDRESS OF FIRST TIOT ENTRY
         DROP  R14                 DROP ADDRESSABILITY (HLASM OBJECTS)
         USING TIOENTRY,R15        ESTABLISH ADDRESSABILITY TO TIOT
*
GNTIOTLP CLI   TIOELNGH,X'ØØ'      END OF TIOT CHAIN?
         BE    GNRETURN             YES (SHOULDN'T HAPPEN)
         CLC   TIOEDDNM(8),DDNAME  PDS NAME FOUND?
         BE    GNDSN                YES
         XR    RØ,RØ               CLEAR REGISTER
         IC    RØ,TIOELNGH         INSERT ENTRY LENGTH
         AR    R15,RØ              POINT TO NEXT ENTRY
         B     GNTIOTLP            CONTINUE
*
GNDSN    XR    R1,R1               CLEAR REGISTER
         ICM   R1,7,TIOEJFCB       ADDRESS OF JFCB
         USING JFCB,R1             ESTABLISH ADDRESSABILITY TO JFCB
         MVC   HEADDSN,JFCBDSNM    MOVE DSNAME TO HEADER
         MVC   HEADDSN-4(4),=C'DSN=' SET DSN ID IN HEADER
*
         MVC   HEADDSN+L'HEADDSN(1Ø),LINE+1 CLEAR TO BLANKS
*
         TM    JFCBIND1,JFCPDS     PARTITIONED DATA SET?
         BZ    GNRETURN             NO
*
         MVC   MEMBER,JFCBELNM     MOVE MEMBER NAME TO SAVEAREA
*
         DROP  R1,R15              DROP ADDRESSING TO JFCB,TIOT,ENTRY
*
         LA    R1,HEADDSN+L'HEADDSN SET FOR NO BLANKS (SHOULDN'T)
         TRT   HEADDSN,TRTAB2      FIND FIRST BLANK
         LR    R2,R1               SAVE ADDRESS
         MVI   Ø(R2),C'('          SEPARATE DSN/MEMBER NAME
*
         MVC   1(8,R1),MEMBER      MOVE MEMBER NAME
*
         LA    R1,9(R1)            SET FOR SCAN FAIL
         TRT   1(8,R2),TRTAB2      FIND FIRST BLANK
         MVI   Ø(R1),C')'          CLOSE MEMBER NAME
*
GNRETURN L     RBAL,SAVGNBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE COPIES RUN IDENTIFICATION TO OUTPUT FILE.      ***
***                                                               ***
```

```
*********************************************************************
*
LOGOUT   ST    RBAL,SAVLOBAL       SAVE LINKAGE REGISTER
*
         MVC   OUTAREA,LINE+1      CLEAR TO BLANKS
         MVC   OUTAREA(L'PROCESSD),PROCESSD  '* PROCESSED BY RDOMERGE'
         MVC   OUTAREA+L'PROCESSD+1(L'HEADDATE+L'HEADTIME),HEADDATE
         BAL   RBAL,WRITEREC       LINK TO WRITEREC
*
         MVC   OUTAREA+2(11),=C'INPUT1 DSN='
         MVC   OUTAREA+13(L'IN1DSN),IN1DSN   SET INPUT1 DSN
         BAL   RBAL,WRITEREC       LINK TO WRITEREC
*
         MVI   OUTAREA+7,C'2'      CHANGE TO INPUT2
         MVC   OUTAREA+13(L'IN2DSN),IN2DSN   SET INPUT2 DSN
         BAL   RBAL,WRITEREC       LINK TO WRITEREC
*
         L     RBAL,SAVLOBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE COUNTS THE DEFINE STATEMENTS IN THE INPUT1 FILE, ***
***   OBTAINS SUFFICIENT STORAGE TO SAVE THEM, CLOSES/OPENS THE    ***
***   FILE FOR FURTHER PROCESSING.                                ***
***                                                               ***
*********************************************************************
*
COUNTREC ST    RBAL,SAVCRBAL       SAVE LINKAGE REGISTER
*
         XR    R4,R4               CLEAR REGISTER
*
CRLOOP   GET   INPUT1,IN1AREA      READ RECORD
*
         CLC   =C'DEFINE ',IN1AREA DEFINE STATEMENT?
         BNE   CRLOOP               NO
*
         LA    R4,1(R4)            COUNT DEFINE STATEMENT
         B     CRLOOP              GO CONTINUE
*
I1EOF    TM    PASSFLAG,1          FIRST PASS?
         BO    C1FINISH             NO
*
         MH    R4,=AL2(L'DEFINE)   SIZE OF SAVE ENTRY
         GETMAIN R,LV=(R4)         GET WORK AREA FOR INPUT BLOCKS
         ST    R1,ADEFSAVE         SAVE ADDRESS
         LA    R4,L'DEFINE         SIZE OF SAVE ENTRY
         ST    R4,ADEFSAVE+4       SAVE SIZE FOR SEARCH BXLE
         SR    R1,R4               INITIALIZE CURRENT POSITION
```

```
         ST    R1,LDEFSAVE         SAVE ADDRESS
*
         MVC   I1CLOSL(I1CLOSLN),CLOSED  SET INPUT1 CLOSE LIST
         CLOSE (INPUT1),MF=(E,I1CLOSL)  CLOSE INPUT1
         OPEN  (INPUT1,(INPUT)),MF=(E,I1OPENL)   RE-OPEN INPUT1
         OI    PASSFLAG,1          FLAG FOR LAST PASS
*
         L     RBAL,SAVCRBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE READS INPUT1, COPIES ALL INPUT TO OUTPUT, AND  ***
***   SAVES DEFINE ENTRY INFORMATION FOR DUPLICATE TEST.          ***
***                                                               ***
*********************************************************************
*
COPYIN1  ST    RBAL,SAVC1BAL       SAVE LINKAGE REGISTER
*
         OI    PASSFLAG,1          FLAG AS SECOND (COPY) PASS
*
C1LOOP   GET   INPUT1,IN1AREA      READ INPUT RECORD
*
         AP    COUNT1,=P'1'        COUNT NEW RECORDS
*
         MVC   OUTAREA,IN1AREA     MOVE TO OUTPUT AREA
         BAL   RBAL,WRITEREC       COPY RECORD
*
         BAL   RBAL,DOGROUP        CHECK FOR GROUP
*
         CLC   =C'DEFINE ',IN1AREA IS THIS A DEFINE STATEMENT?
         BE    C1DEFINE             YES
*
         TM    PASSFLAG,X'8Ø'      FIRST DEFINE REACHED?
         BO    C1LOOP               YES
*
         MVC   LINE+1(L'IN1AREA),IN1AREA MOVE BEGINNING LINE FOR PRINT
         BAL   RBAL,PRINT          PRINT COMMENTS, ETC.
         B     C1LOOP              GO READ NEXT RECORD
*
C1DEFINE TRT   IN1AREA+6(65),TRTAB1 SEARCH FOR NON-BLANK
         BZ    C1LOOP              NONE FOUND
*
         OI    PASSFLAG,X'8Ø'      INDICATE THAT A DEFINE HAS OCCURRED
*
         L     R2,LDEFSAVE         GET PREVIOUS SAVE ADDRESS
         LA    R2,L'DEFINE(R2)     POINT TO NEXT AVAILABLE AREA
         ST    R2,LDEFSAVE         SAVE CURRENT POSITION
         MVC   Ø(L'DEFINE,R2),Ø(R1) SAVE DEFINE ID
```

```
         B     C1LOOP              GO CONTINUE COPY
*
C1FINISH MVC   OUTAREA,LINE+1      CLEAR TO BLANKS
         MVC   OUTAREA(28),=C'* END PROCESSING BY RDOMERGE'
         MVC   OUTAREA+29(L'HEADDATE+L'HEADTIME),HEADDATE SET DATE/TIME
         BAL   RBAL,WRITEREC       LINK TO WRITEREC
*
C1RETURN L     RBAL,SAVC1BAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE COPIES INPUT2 TO OUTPUT UNTIL EITHER A DEFINE  ***
***   STATEMENT OR END-OF-FILE IS REACHED.                        ***
***                                                               ***
*********************************************************************
*
COPYJCL  ST    RBAL,SAVCJBAL       SAVE LINKAGE REGISTER
*
CJLOOP   GET   INPUT2,IN2AREA      READ RECORD
*
         AP    COUNT2,=P'1'        COUNT RECORD
*
         CLC   =C'DEFINE ',IN2AREA CSD DEFINE STATEMENT?
         BE    CJRETURN             YES
*
         CLC   PROCESSD,IN2AREA    PREVIOUS RDOMERGE BEGIN
         BE    CJRETURN             YES
*
         MVC   OUTAREA,IN2AREA     MOVE RECORD
         BAL   RBAL,WRITEREC       COPY RECORD TO OUTPUT FILE
*
         B     CJLOOP              CONTINUE SEARCH
*
I2EOF    TM    PASSFLAG,2          WAS COPYIN2 BEGUN?
         BO    C2FINISH             YES
*
         OI    PASSFLAG,2          FLAG NO DEFINE STATEMENTS
*
CJRETURN L     RBAL,SAVCJBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*********************************************************************
***                                                               ***
***   THIS ROUTINE READS THE FILE FROM INPUT2, IF IT IS THE       ***
***   BEGINNING OF A DEFINE STATEMENT IT SEARCHES THE 'DEFINE'    ***
***   TABLE FOR DUPLICATES AND SETS 'DUPFLAG' EITHER ON OR OFF.   ***
***   IF THE DUPFLAG IS ON THE COMMENT IS WRITTEN AND THE         ***
```

```
***   STATEMENT IS COMMENTED OUT.                                    ***
***                                                                  ***
**********************************************************************
*
COPYIN2  ST    RBAL,SAVC2BAL       SAVE LINKAGE REGISTER
*
         MVI   DUPFLAG,Ø           INITIALLY TURN OFF FLAG
*
         TM    PASSFLAG,2          NULL 'OLD' FILE OR DEFINE?
         BO    C2RETURN             YES
*
         OI    PASSFLAG,2          FLAG FOR EXIT ON END-OF-FILE
         B     C2RESUME            GO PROCESS RECORD READ BY COPYJCL
*
C2LOOP   GET   INPUT2,IN2AREA      READ INPUT2
*
         AP    COUNT2,=P'1'        COUNT RECORD
*
         CLC   =C'/*',IN2AREA      E-O-J (OTHER JCL HAS BEEN BYPASSED)
         BE    C2SS                 NO
*
         CLC   =C'//',IN2AREA      E-O-J (OTHER JCL HAS BEEN BYPASSED)
         BNE   C2NOTSS              NO
*
C2SS     BAL   RBAL,PUTGROUP       LINK TO PUTGROUP
*
C2NOTSS  TRT   IN2AREA,TRTAB1      SEARCH FOR FIRST NON-BLANK
         BZ    C2ADD               BRANCH IF NOT FOUND
*
         CLC   =C'ADD ',Ø(R1)      ADD STATEMENT?
         BNE   C2NOTADD             NO
*
         MVI   DUPFLAG,Ø           TURN OFF DUPLICATE FLAG
*
C2ADD    MVC   OUTAREA,IN2AREA     MOVE IMAGE
         B     C2COPY              GO COPY ADD STATEMENT
*
C2NOTADD CLI   IN2AREA,C' '        DEFINE CONTINUATION?
         BE    C2CONT               YES
*
C2RESUME CLC   =C'DEFINE ',IN2AREA BEGINNING OF DEFINE STATEMENT?
         BNE   C2CONT               NO
*
         TRT   IN2AREA+7(65),TRTAB1 SEARCH FOR NON-BLANK
         BZ    C2CONT              OUT IF NOT FOUND
*
         LR    R3,R1               SAVE ADDRESS OF NON-BLANK
         LA    R4,IN2AREA+72       POINT PAST LAST POSSIBLE LOCATION
         LR    R1,R4               SAVE FOR POSSIBLE SEARCH FAILURE
         SR    R4,R3               MAXIMUM LENGTH
         EX    R4,C2TRT            SEARCH FOR FIRST BLANK
```

```
         LR    R2,R1               ADDRESS OF FIRST BLANK
         SR    R2,R3               LENGTH OF TYPE/ENTRY
         BNP   C2CONT              EXIT IF NOT POSITIVE
         BCTR  R2,Ø                LENGTH-1
         LR    R1,R3               SAVE STARTING POSITION
*
         MVI   DUPFLAG,Ø           INITIALLY TURN OFF FLAG
         LM    R3,R5,ADEFSAVE      LOAD REGISTERS
*
C2LOOP2  EX    R2,C2CLC            MATCH FOUND?
         BE    C2MATCH              YES
         BXLE  R3,R4,C2LOOP2       CONTINUE SEARCH
         B     C2CONT              GO COPY STATEMENT
*
C2MATCH  AP    DUPS,=P'1'          COUNT DUPLICATE
         MVI   DUPFLAG,X'FF'       SET FLAG
*
         MVC   OUTAREA,LINE+1      SET TO BLANKS
         MVC   OUTAREA(43),=C'*DUPLICATE DEFINE COMMENTED OUT BY RDOMER-
               GE'
         MVC   OUTAREA+44(L'HEADDATE+L'HEADTIME),HEADDATE SET DATE/DATE
         BAL   RBAL,WRITEREC       COPY COMMENT TO OUTPUT
*
         MVC   LINE+1(LDUPPAT),DUPPAT SET EDIT PATTERN
         ED    LINE+L'DUPPAT(6),COUNT2 FORMAT RECORD COUNT
         MVC   LINE+1+LDUPPAT+2(L'IN2AREA),IN2AREA COPY RECORD
         BAL   RBAL,PRINT          PRINT DUPLICATE DELETED MESSAGE
*
C2CONT   MVC   OUTAREA,IN2AREA     ASSUME NOT DUPLICATE
*
         CLI   DUPFLAG,Ø           ASSUMPTION CORRECT?
         BE    C2COPY               YES
*
         CLI   IN2AREA,C'*'        ALREADY COMMENT?
         BE    C2COPY               YES
*
         CLI   IN2AREA,C'/'        JCL STATEMENT?
         BE    C2COPY               YES
*
         MVC   OUTAREA,IN2AREA-1   MOVE FLAGGED RECORD
         MVC   OUTAREA+71(8),IN2AREA+71 ADJUST COLUMNS 72-8Ø
*
C2COPY   BAL   RBAL,WRITEREC       GO WRITE RECORD TO OUTPUT FILE
*
         BAL   RBAL,DOGROUP        GO CHECK FOR POSSIBLE GROUP NAME
*
         B     C2LOOP              GO PROCESS NEXT RECORD
*
C2FINISH BAL   RBAL,PUTGROUP       GO WRITE ADD GROUPS
*
C2RETURN L     RBAL,SAVC2BAL       RESTORE LINKAGE REGISTER
```

```
         BR    RBAL                RETURN
*
C2TRT    TRT   Ø(*-*,R3),TRTAB2
C2CLC    CLC   Ø(*-*,R1),Ø(R3)
*
         EJECT
*************************************************************************
***                                                                   ***
***   COPY RECORD TO 'OUTPUT'                                         ***
***                                                                   ***
*************************************************************************
*
WRITEREC ST    RBAL,SAVWRBAL       SAVE LINKAGE REGISTER
*
         PUT   OUTPUT,OUTAREA      WRITE RECORD
*
         L     RBAL,SAVWRBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
*************************************************************************
***                                                                   ***
***   THIS ROUTINE PRINTS FINAL TOTALS                                ***
***                                                                   ***
*************************************************************************
*
DOTOTALS ST    RBAL,SAVDTBAL       SAVE LINKAGE REGISTER
*
         MVC   LINE(LPAT1),PAT1  SET EDIT PATTERN
         ED    LINE+L'PAT1(6),COUNT1 FORMAT INPUT1 RECORDS COUNT
         BAL   RBAL,PRINT          PRINT INPUT1 COUNT
         BAL   RBAL,DOUBLESP       ALLOW FOR DOUBLE SPACE
*
         MVC   LINE(LPAT2),PAT2  SET EDIT PATTERN
         ED    LINE+L'PAT2(6),COUNT2 FORMAT INPUT1 RECORDS COUNT
         BAL   RBAL,PRINT          PRINT INPUT2 COUNT
*
         MVC   LINE+1(LPATD),PATD  SET EDIT PATTERN
         ED    LINE+L'PATD(6),DUPS FORMAT DUPLICATE COUNT
         BAL   RBAL,PRINT          PRINT INPUT1 COUNT
*
         L     RBAL,SAVDTBAL       RESTORE LINKAGE REGISTER
         BR    RBAL                RETURN
*
         EJECT
```

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

Keith H Nicaise
Technical Services Manager
Touro Infirmary (USA)

Flows and SYNCPOINTs in DPL

Function shipping was introduced into CICS in 1977, and has been widely emulated in the LAN and WAN marketplace. The CICS resources that could be accessed transparently were keyed files, queues, both temporary storage and transient data, and the initiation of remote asynchronous transactions (ie EXEC CICS START).

When CICS is executing an EXEC CICS command that includes a CICS resource, a check is made against the resource definition entry to see whether the SYSID field is blank, and if so the command is executed locally. The command is also executed locally if the value is equal to the CICS that is executing the command. If neither of these two conditions is true then the command will be shipped to the system with the SYSID specified where the CICS mirror will execute (mirror) the command for you. The second SYSID check is very important! This allows the same resource definitions in the CICS System Definition File (CSD) to be used on multiple CICS systems and the command will execute correctly no matter whether the resource is local or remote.

DISTRIBUTED PROGRAM LINK

In 1990, CICS OS/2 extended the remote resources to include programs, so giving it a Transactional Remote Procedure Call (TRPC) capability. Other members of the CICS family, including CICS/ESA, have now implemented this powerful function, which was named Distributed Program Link (DPL).

A simple remote EXEC CICS LINK request and its associated flows are shown in Figure 1. The abbreviations used are explained at the end of the article. They have an SNA flavour, but some CICS implementations flow over other protocols, for example TCP/IP.

Note: the invoked program(PROGRAMB) *must not* issue SYNCPOINTs because the caller (the left hand side) is in charge of the conversation. If the called program does issue an EXEC CICS SYNCPOINT, then an ADPL abend will result with an 'EIBRESP2 = 200'. On the other hand, EXEC CICS SYNCPOINT ROLLBACK is permissible and will negate the updates to all local and connected

remote protected resources. Care must be taken to make sure that the CICS ABEND happens – in C or C++, where the default action is NOHANDLE, the HANDLE option will have to be added to the EXEC CICS SYNCPOINT command. You should also check whether there is an EXEC CICS HANDLE ABEND active because this will catch the ABEND and, unless the ABEND is re-issued, you will have to explicitly code to tell the other end that you wish to ROLLBACK.

In the documentation there is a list of EXEC CICS commands that cannot be invoked from within a linked-to program. From the explanation above, it should be simple to see that any command that does not use an MRO/ISC entry as its principal facility will, by definition, be invalid. These checks were not implemented on CICS/MVS, so if programs have used this loophole they will have to be updated for a Year 2000 release.

The flows shown in Figure 1 are equivalent to the optimized last agent flows of a two-phase commit process.

Also note how the linked program runs under transaction-id LOCL – this is important for PLAN authorization with DB2.

CICS COMMAREAs are defined to be the same length when sent as when returned; however, most programming models either send a small amount of data and receive lots (equivalent to an inquiry) or send lots and receive a little (equivalent of an update), so this could result in a lot of non-useful data being transmitted through a network. All CICS implementations contain an optimization to try to reduce the amount of data flowing around the network by 'not sending' trailing null characters, X'00's, and reconstituting the total COMMAREA at the receiving end.

An equivalent CICS API function is performed by the use of the DATALENGTH option on an EXEC CICS LINK on the invoking side. Unfortunately, there is no equivalent on an EXEC CICS RETURN, so you have to fill the COMMAREA by hand, from the end to the last significant byte, with null characters (LOW-VALUES to COBOL programmers).

Many user networks are unable to sustain two round trips through the network per transaction, so the SYNCONRETURN option was added to the EXEC CICS LINK command. If needed and applicable, the addition and use of this option will cut the network flows between one

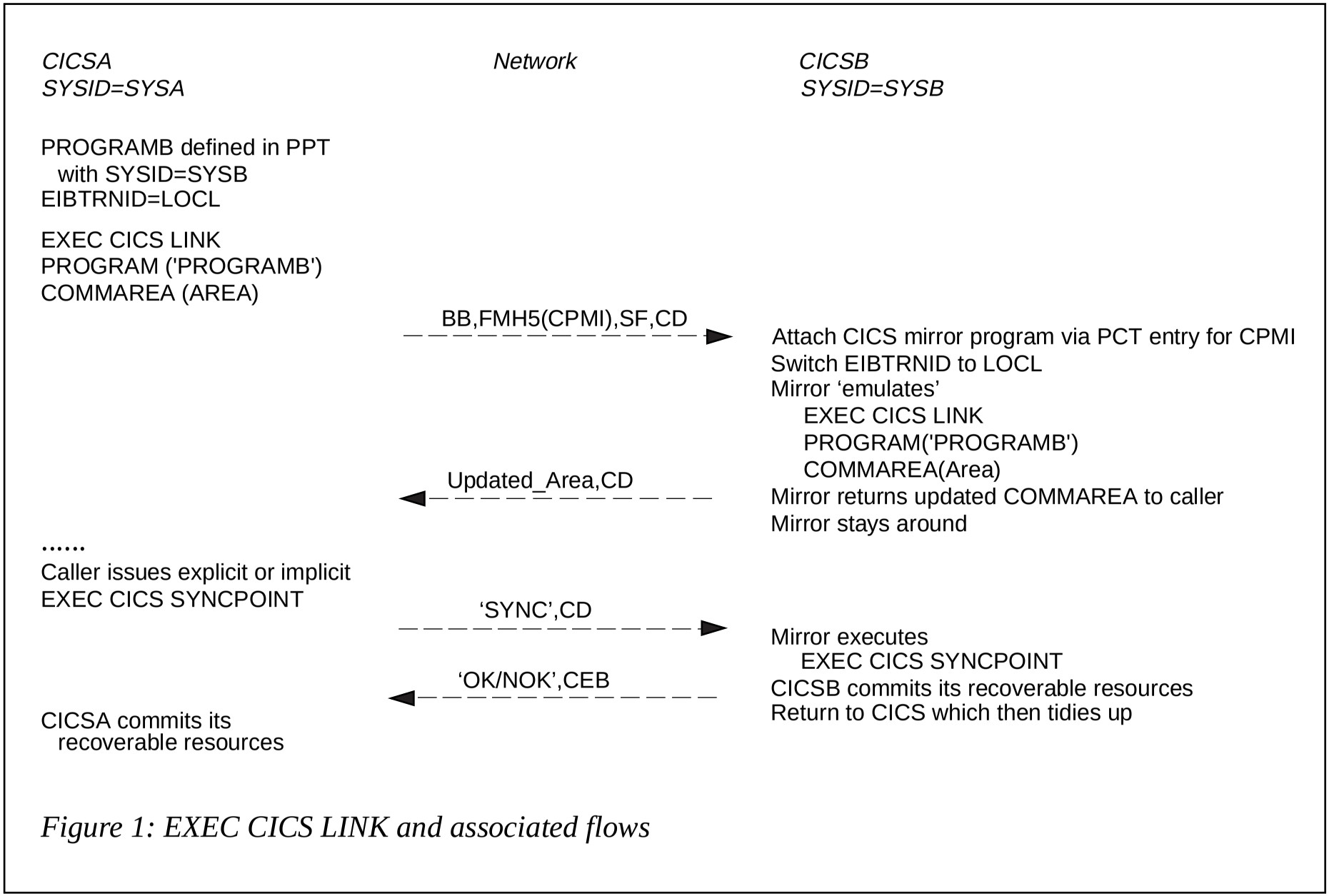

Figure 1: EXEC CICS LINK and associated flows

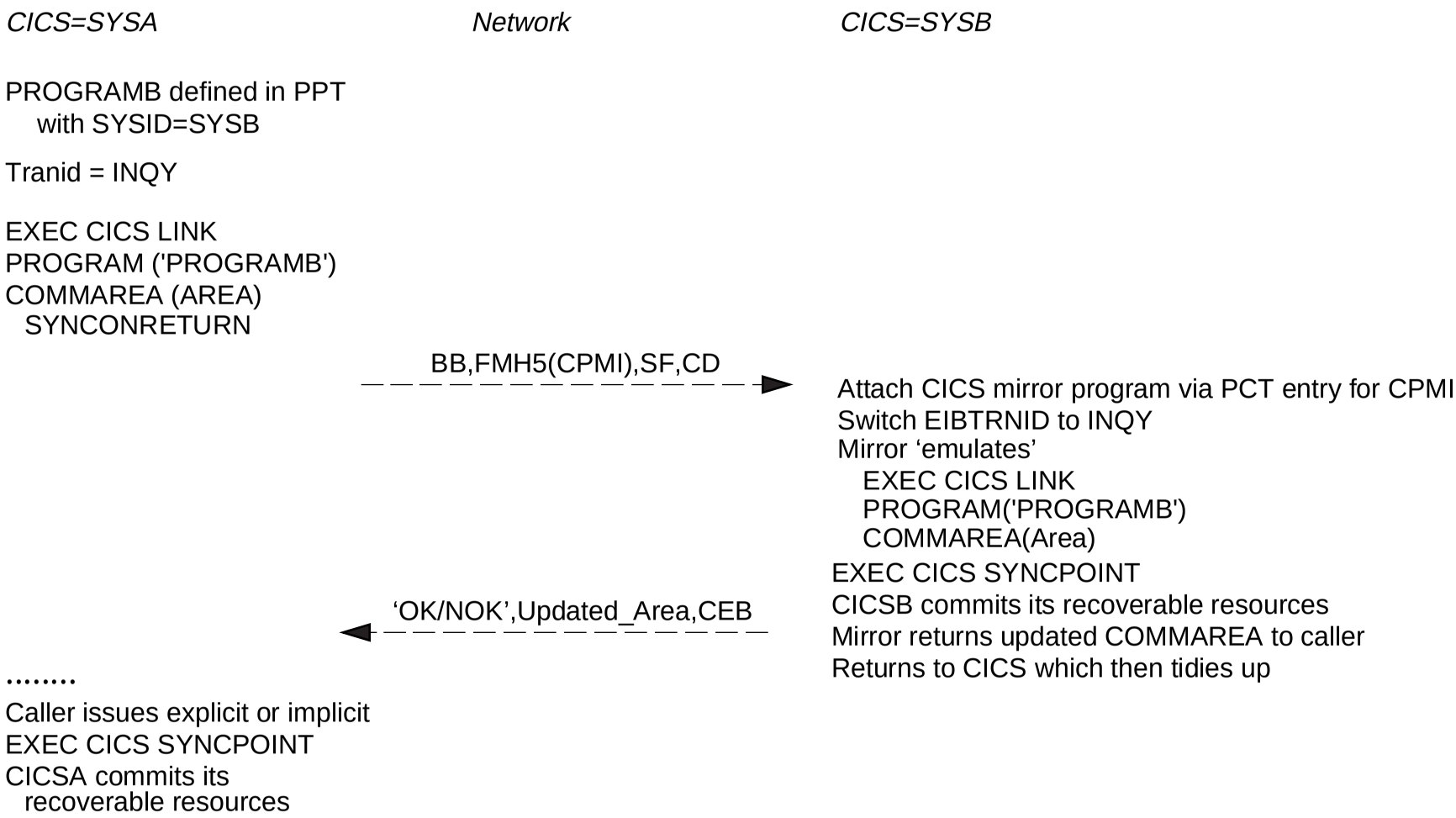

Figure 2: Addition of SYNCONRETURN option

CICS and another by half. This is shown in Figure 2.

The EXEC CICS SYNCPOINT in Figure 2 can be issued in two ways – implicitly, by the CICS mirror terminating, or explicitly, if invoked in PROGRAMB.

In the explicit case, the challenge is, as always, to see whether the commit call has worked or not. Provided you do not have an active EXEC CICS HANDLE ABEND, then the abend in the user program will be propagated up to the CICS mirror and a ROLLEDBACK response will be reflected at the calling end. However, if you are handling abends yourself, either with an EXEC CICS HANDLE ABEND or you have coded the RESP keyword on the EXEC CICS SYNCPOINT, then it is up to you to tell the caller that the called program has not SYNCPOINTed correctly.

Notice how different the recovery scopes are – in the first case the caller, CICSA, was in charge of committing the recoverable resources on CICSB, whereas in the second case (SYNCONRETURN) the called program, or the CICS mirror, commits the resources on CICSB and returns to the caller with an updated COMMAREA and an 'OK/NOK' indicator.

For security or billing reasons, some users wanted to run the remote program under a different transaction-id. To meet this requirement a TRANSID operand was added (see Figure 3). The transaction definition

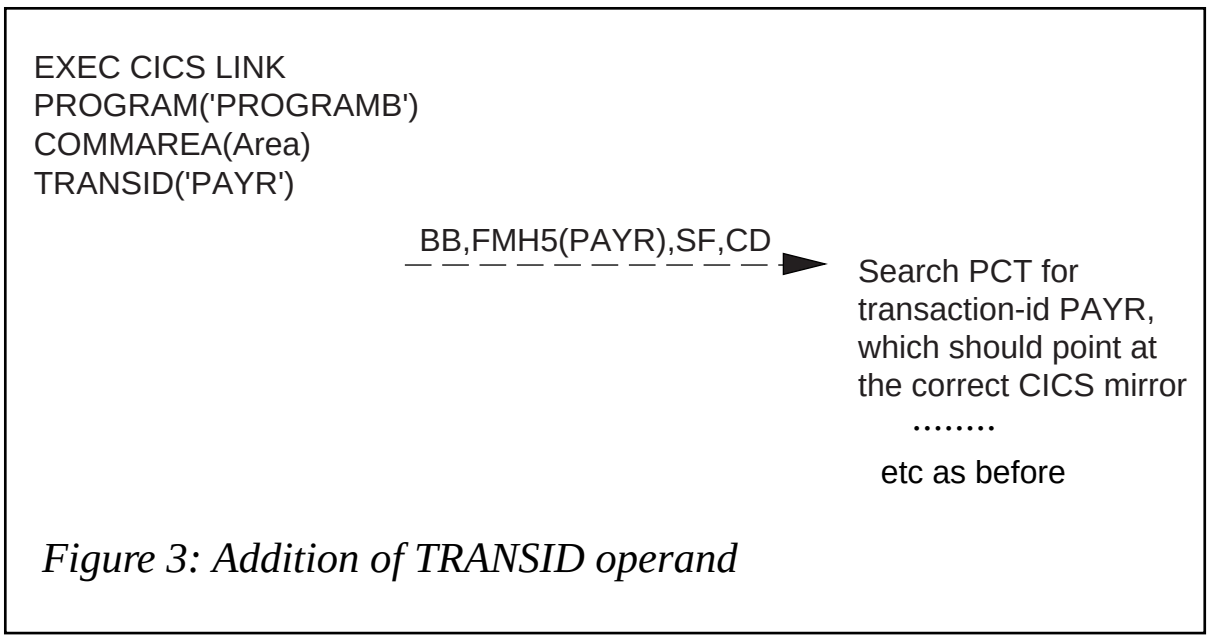

Figure 3: Addition of TRANSID operand

has to point to the relevant CICS mirror program with its appropriate profile.

Two very common system programming errors can occur:

- Defining PAYR to point at the correct CICS mirror program in the PCT – but forgetting to install the definition!
- Incorrectly pointing PAYR directly at PROGRAMB in the PCT. From the above description, you should see that the CICS mirror program is vital in decoding and encoding the structured fields and looking after the commit scopes.

ABBREVIATIONS

The following abbreviations have been used in this article:

- BB – begin bracket, an indicator to tell the receiver this data is the beginning of a transaction/conversation.
- CD – change direction, an indicator to tell the receiver this data is the completion of what is being sent and the receiver owns the flow and should reply.
- FMH – function management header, a set of indicators to explain the protocol and capability of the sender to request the execution of the transaction contained within it.
- FMH5 – architected function management header for an LU6.2 conversation which contains the transaction-id.
- SF – a self-describing structured set of fields containing all the parameters that need to be passed and can be decoded at the receiving end.
- FMH43 – CICS architected function management header that is a structured field containing an encoding of the requested CICS function to be executed on the remote system.
- CEB – conditional end bracket ie an indicator to tell the receiver this data is the end of a transaction.

Andy M Krasun
IBM (UK)

MacKinney has announced its Macro Level Interpreter (MLI) for shifting CICS macro code to CICS/ESA and CICS Transaction Server. MLI translates macro-level applications to command-level without the need for the original source code, and eliminates the need to maintain multiple versions and unsupported versions of CICS. It supports Assembler, COBOL, and PL/I languages, and command-level applications using the restricted EXEC CICS ADDRESS CSA command in CICS/ESA Version 3 or above. The software also supports vendor applications written with CICS macro code.

An optional 31-bit feature allows applications to execute above the 16MB line, while an optional macro feature, MLIMAC, eliminates the need for CICS 2.1.2 software by providing compile libraries for Assembler, COBOL, and PL/I languages.

Another optional feature, Macro Level Detector, audits applications and determines which programs must be translated by MLI. Storage Protection in CICS/ESA 3.3 and above is supported with no CPU overhead, plus Dynamic Attach, mixed mode programs, and all standard DFH calls. It also supports ISAM compatibility or unblocked files under CICS/ESA, and it co-exists with debugging tools like XPEDITER from Compuware and INTERTEST from Computer Associates.

For further information contact:
MacKinney Systems, 2740 South Glenstone, Suite 103, Springfield, MO 65804, USA.
Tel: (417) 882 8012.
URL: http://www.mackinney.com.

* * *

Data 21 has announced the VSE version of IpServer for CICS, which runs natively within CICS enabling it to take advantage of the capabilities of the System/390 CICS environment. A CICS Web Server is complemented by a native CICS CGI interface that simplifies Web-enabling CICS applications.

The native CICS CGI interface enables users to leverage existing hardware, software, and programming skills to create enterprise class e-business applications. The CGI interface is fully multi-threaded and allows programmers to write CGIs in familiar CICS command-level languages.

For further information contact:
Data 21, 18093-H South Prairie Avenue, Torrance, CA 90504-3700, USA.
Tel: (702) 832 2191.
URL: http://www.data21.com.

* * *

IBM has announced Expedite/CICS Version 4.4, which provides communications and user interfaces to the EDI Services mailbox component of IBM Information Exchange.

Enhancements to Version 4.4 include: intersystem addressing for UN/EDIFACT and UN/TDI; do-not-stop processing when encountering an invalid ISA; split option for 'other' file types; automate VSE batch receive; program names added to trace and log files; and enhanced capability to handle duplicate control records.

For further information contact your local IBM representative.

* * *