169 MVS update

October 2000

In this issue

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 33598
From USA: 01144 1635 33598
E-mail: Jaimek@xephon.com

North American office
Xephon/QNA
PO Box 350100,
Westminster, CO 80035-0100
USA
Telephone: (303) 410 9344
Fax: (303) 438 0290

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

Editor
Jaime Kaminski

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, EXECs, and other contents of this journal before making any use of it.

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

Subscriptions and back-issues
A year's subscription to *MVS Update*, comprising twelve monthly issues, costs £340.00 in the UK; $505.00 in the USA and Canada; £346.00 in Europe; £352.00 in Australasia and Japan; and £350.00 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1992 issue, are available separately to subscribers for £29.00 ($43.00) each including postage.

Printed in England.

Modifying catalogs using SELCOPY

The following is a quick SELCOPY tip to deal with possible catalog problems. It stems from an error that happened on one of our test systems where a user catalog became damaged by a duplicate key. Because catalogs are basically just KSDSs, SELCOPY is a good tool to use because it is excellent at manipulating VSAM files. It does however need a tweak to modify catalogs in that it needs to be in an APF library as AC(1). To avoid this potentially being a problem, I have included a short program to load SELCOPY and use an SVC to dynamically assign that attribute. If you do not have such an SVC, then there have been several such programs documented in *MVS Update*.

To correct the duplicate key, all that was required was first to run SELCOPY to find where the problem existed, at which point SELCOPY terminates (or alternatively run an IDCAMS DIAGNOSE to identify the key). Next run another SELCOPY to delete that record. To see how easy that is, please review the following statements.

```
//your job card
//A EXEC PGM=SELAUTH
//STEPLIB DD DSN=steplib.with.selauth.in,DISP=SHR
//SYSPRINT DD SYSOUT=*
//IFILE DD DSN=CATALOG.TEST.MASTER,DISP=SHR
//SYSIN DD *
READ IFILE VSAM UPD
IF POS ANY EXACT='damaged.file.name'
                                    - Ensure this is a unique data search
THEN DEL IFILE KSDS
THEN EOJ
```

SELAUTH

```
//your job card
//STEPA EXEC ASMFCL
//ASM.SYSIN DD *
SELAUTH  TITLE 'AUTHORIZE SELCOPY'
SELAUTH  AMODE 24
SELAUTH  CSECT
         BAKR  14,Ø
         LR    12,15
         USING SELAUTH,12
         SVC   242
                                        - Insert your APF on SVC number
```

```
         LOAD  EP=SELCOPY
         LR    15,Ø
         EREG  Ø,13
         BASR  14,15
         SVC   243
                          - Insert your APF off SVC number (optional)
         PR
         END
/*
//LKED.SYSLMOD DD DSN=your.load.library,DISP=SHR
//LKED.SYSIN DD *
         ENTRY SELAUTH
         SETCODE AC(1)
         NAME  SELAUTH(R)
```

SELCOPY (once authorized) can exploit all its VSAM manipulation abilities on catalogs. It is therefore an excellent, if often overlooked, tool for fixing catalogs. Over the years I have used it for rebuilding aliases, and correcting volume errors and duplicate keys.

Trapping an auxiliary storage shortage

Most MVS systems programmers would over-define page datasets on their systems to cater for emergencies. Despite this precaution, at times something out of the ordinary happens that results in a massive overflow of auxiliary storage. This could be a DB2 or IMS system that suddenly does something out of the ordinary, or even a system address space like DUMPSRV. Whatever the cause of it, most MVS sysprogs have had the experience where they had to add page datasets in a hurry. This in itself is not a problem nowadays (provided you have a spare one readily available) because the 'Page Add' command allows for this to be done on the fly. So, in a multi-LPAR environment we can simply keep a single page volume as 'spare' and add it to whichever system may need it.

The only problem with this scenario is that it is not easy to find out afterwards what the cause of the problem was. The program we have here can be used to trap the event by writing out to the log a list of all major auxiliary storage users. The way to do this is to set up the

program to run as an MPF exit. This is done by coding the following statement in you MPFLST*xx* member:

```
IRA200E,SUP(NO),USEREXIT(SHOWAUXU)
```

With this statement in place and the program in a LNKLSTed library, you can concentrate on fixing the problem, knowing that you will afterwards have a report on the system log that shows which address space caused the problem in the first place. This is the format of the output on the log:

```
JOB=JES2     VIO=000000,NonVIO=001252 slots
JOB=CATALOG  VIO=000000,NonVIO=000702 slots
etc
```

SOURCE

```
*********************************************************************
*  This routine will display all users of more than SLOTNUM slots of
*  auxilliary storage.
*  The value of SLOTNUM is currently set at 5000, this can be changed
*  and would require the program to be re-assembled.
*
SLOTNUM  EQU   500
SHOWAUXU CSECT
SHOWAUXU AMODE 31
SHOWAUXU RMODE ANY
         BAKR  R14,0               .Save caller's status
         LR    R12,R15
         USING SHOWAUXU,12
*********************************************************************
*        Main driver routine
*********************************************************************
         LA    R3,STORSIZE         .Our requirement
STORAGE  STORAGE OBTAIN,LENGTH=(3),LOC=ANY,SP=229
         LR    R2,R1               .Point to getmained area
         LA    R3,STORSIZE         .Length of area to clear
         XR    R9,R9
         MVCL  R2,R8               .Propagate binary zeros
*
         USING STORAREA,R1         .Addressability to getmained area
         ST    R13,SAVEAREA+4      .Backchain
         LR    R13,R1
         DROP  R1
         USING STORAREA,R13        .Addressability to getmained area
         BAS   R14,SCANQUE         .Display wanted information
         LA    R3,STORSIZE         .Size of area to free
         LR    R2,R13              .Address of area to free
STORAGE RELEASE,LENGTH=(R3),ADDR=(R2),SP=229
```

```
         XR    R15,R15             .Clear return code
TOCALLER PR    ,                   .Return to caller
         DS    ØD                  .Align
*********************************************************************
*        This routine displays the wanted information
*********************************************************************
SCANQUE  BAKR  R14,Ø               .Preserve caller's status
         L     R7,16               .CVT address
         USING CVTMAP,R7           .Addressability to CVT
         L     R3,CVTASVT          .CVT -> ASVT
         USING ASVT,R3             .ASVT DSECT
         XR    R8,R8               .Offset pointer
         LA    R9,R1               .Number of entries inspected
         MVC   WTLBUFER(WTLMSGLN),WTLMSG Prepare WTL message
*
ASCBLOOP CL    R9,ASVTMAXU         .Have all entries been inspected?
         BH    SCANEXIT            .Yes,get out
*
         L     R7,ASVTENTY(8)      .No, inspect next one
         LA    R9,1(R9)
         LA    R8,4(R8)            .Point to next entry
         DROP  R7
         USING ASCB,R7             .Addressability to ASCB
*
USECHECK CLM   R7,B'1ØØØ',=X'8Ø'   .ASID being used?
         BE    ASCBLOOP            .No, skip
*
INUSE    LA    R6,ASCBJBNI         .Zero if uninitialized
         CLM   R6,B'Ø111',=XL3'ØØ'
         BE    ASCBLOOP            .Not in use, skip
*
ACTIVE   EQU   *                   .The address space is active
MOVENAME L     6,ASCBJBNS          .Jobname for MOUNT/LOGON/STC
         NI    SHOWIT,NO           .Set show-it flag off
         MVC   JOBNAME,Ø(6)        .Move JobName
SHOWTHEM L     R4,ASCBASSB
         USING ASSB,R4
         L     R5,ASSBVSC          .Number of VIO slots in use
         LR    R1,R5
*
CVTVIO   CVD   R5,DOUBLE
         UNPK  WTLBUFER+21(6),DOUBLE+5(3)
         OI    WTLBUFER+26,X'FØ'
         L     R5,ASSBNVSC         .Number of NONVIO slots in use
         AR    R1,R5
         CH    R1,=AL2(SLOTNUM)    .At least minimum pages in use?
         BL    CVTNVIO             .No, less than minimum pages VIO
         OI    SHOWIT,YES          .Yes, we have to show this one
*
CVTNVIO  CVD   5,DOUBLE
         UNPK  WTLBUFER+35(6),DOUBLE+5(3)
```

```
         OI    WTLBUFER+40,X'F0'
         CLC   JOBNAME(4),=C'INIT' .Is the jobname INIT?
         BNE   MOVEJOBN            .No, accept as is
*
         L     6,ASCBJBNI
         LA    6,0(6)
         SH    6,=H'8'
         USING CHAIN,R6
         MVC   JOBNAME,CHKEY       .Pick up the real jobname
*
MOVEJOBN CLI   JOBNAME,X'04'
         BE    ASCBLOOP

         LA    1,WTLBUFER
         MVC   WTLBUFER+8(8),JOBNAME
         TM    SHOWIT,YES          .Does job have large # slots in use?
         BNO   ASCBLOOP            .No, skip this one
*
         WTL   MF=(E,(1))          .Yes, do WTL
         B     ASCBLOOP            .Do for each ASCB
*
SCANEXIT PR
*********************************************************************
*        Constants follow
*********************************************************************
         LTORG
WTLMSG   WTL  'JOB=XXXXXXXX VIO=000000,NonVIO=000000 slots',          *
               MF=L
WTLMSGLN EQU   *-WTLMSG
*********************************************************************
*        DSECTS follow
*********************************************************************
STORAREA DSECT
SAVEAREA DS    18F                .General savearea
DOUBLE   DS    D                  .Double word for conversions
JOBNAME  DS    CL8
SHOWIT   DS    C                  .Significance flag
WTLBUFER DS    CL(WTLMSGLN)       .Workarea for WTL
STORSIZE EQU   *-STORAREA
*
R0       EQU   0
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R10      EQU   10
```

```
R11      EQU   11
R12      EQU   12
R13      EQU   13
R14      EQU   14
R15      EQU   15
         CVT   DSECT=YES
         IHAASCB
         IHAASSB
         IHAASVT
         IEZVX1Ø1                  .DSECT for command exit fields
         IEECHAIN
NO       EQU   X'ØØ'
YES      EQU   X'Ø1'
         END
```

Thomas Afbeen
Brandenburg Consulting (South Africa)

Issuing WTORs in SELCOPY

The following piece of code is a 'quick tip' that resulted from a request from our schedulers. They wanted a way to issue WTORs to pause a job between steps, and to allow the operator to state whether the job should complete or not. Rather than do any Assembler, if you have SELCOPY, do the following:

```
//your job card
//B EXEC PGM=SELCOPY
//SYSPRINT DD SYSOUT=*
//SYSOUT DD SYSOUT=A
//SYSIN DD *
READ CARD W 2ØØ
WTO   'REPLY END OR JUST PRESS ENTER' REPLY 3 AT 1ØØ
IF POS 1ØØ EQ 'END'
   THEN RETCODE=8
ELSE RETCODE=4
END
DUMMY CARD
```

This will issue a WTOR to the console and will set a return code through the use of a RETCODE that can be used in JCL IF or COND statements. In this case a return code of 8 results if the operator enters 'END' otherwise a code of 4 is set.

Systems Programmer (UK)

MVS free space information on the Web

INTRODUCTION

With more and more emphasis on utilizing Web browsers for the presentation of information, we developed this very simple routine for taking DASD free space data from a DCOLLECT file and also produced an outfile file in HTML format. This HTML file can then be served from a OS/390 Web services environment, or it can be sent to the Web server of choice in your environment. This is a very simple routine, and the HTML produced is very basic, but it provides a basis from which many other features can be implemented if so desired.

This program has been tested and implemented under MVS 5.2.2 with DFSMS 1.3, and has also been utilized under OS/390 with DFSMS 1.5. The $ESAPRO, $ESAEPI, and $ESASTG have been included and the end of the progam.

STORGHTM

```
         TITLE 'STORGHTM - SMS FREE SPACE INFORMATION IN HTML FORM'
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* CSECT   : STORGHTM                                                   *
* MODULE  : STORGHTM                                                   *
* DESC    : STORGHTM IS A UTILITY PROGRAM WHICH ACCEPTS DCOLLECT       *
*           OUTPUT AS INPUT DATA, AND GENERATES A HTML OUTPUT FILE.    *
* MACROS  : $ESAPRO $ESAEPI $ESASTG OPEN CLOSE DCB DCBE                *
*           GET PUT                                                    *
* DSECTS  :                                                            *
* INPUT   : DCOLLECT - STANDARD OUTPUT FROM IDCAMS DCOLLECT            *
* OUTPUT  : HTMLFILE - FILE CONTAINING FREE SPACE INFORMATION          *
*                      IN HTML FORMAT                                  *
* PLIST   : INFORMATION FOR THE HTML TITLE LINE                        *
* CALLS   : NONE                                                       *
* NOTES   : 31-BIT ADDRESSING USED FOR ALL FILES.                      *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
STORGHTM $ESAPRO R12,R11,RM=24,AM=31
         SPACE 1
         L     R2,Ø(R1)               PICK UP THE PLIST POINTER
         ST    R2,@PLIST              SAVE IT FOR LATER
         LA    RØ,$S_GRP              GET @(FIRST GROUP OF DATA)
         ST    RØ,FIRST@              SAVE IT FOR BXLE USE
```

```
         LR    R1Ø,RØ                  SAVE IT FOR LATER USE
         ST    RØ,LAST@                SAVE IT FOR BXLE USE
         LA    RØ,$LEN                 GET THE INCREMENT SIZE
         ST    RØ,INC_V                SAVE IT FOR BXLE USE
         OPEN  (DCOLLECT,(INPUT))
         SPACE 1
R_LOOP   DS    ØH
         SPACE 1
         GET   DCOLLECT,V_REC
         CLC   RECTYPE,V_TYPE          Q. V RECORD TYPE
         BNE   R_LOOP                  A. NO, GET NEXT RECORD
         CLC   BLANK15,S_GRP           Q. IS THE STOR GROUP BLANK
         BNE   NOTBLANK                A. NO, PROCESS IT
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* LOCAL CONVENTION FOR MVS SYSTEM VOLUMES - YOURS MAY VARY            *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
         CLC   LIT_MVS,VOLUME          Q. FIRST THREE = MVS
         BNE   NOT_ML1                 A. NO, PROCESS IT
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* LOCAL CONVENTION FOR HSM ML1 VOLUMES - YOURS MAY VARY               *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
         CLC   VOLUME+4(L'LIT_H),LIT_H Q. HSM ML1 VOLUME
         BNE   NOT_ML1                 A. NO, PROCESS IT
         MVC   S_GRP(L'LIT_HSM),LIT_HSM  MAKE IT HSM ML1 GROUP
         B     NOTBLANK
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* HANDLE OTHER VOLUMES WHICH ARE NOT SMS MANAGED - YOURS MAY VARY     *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
NOT_ML1  DS    ØH
         SPACE 1
         LM    R7,R9,SHR_CTL           PRIME REGISTERS FOR BXLE LOOP
NOT_ML11 DS    ØH
         L     R14,Ø(R7)               GET LENGTH OF THE LITERAL
         B     *+1Ø                    BRANCH AROUND THE CLC
         CLC   4(*-*,R7),VOLUME        THE ACTUAL COMPARE
         EX    R14,*-6                 EXECUTE THE PREVIOUS INSTRUCTION
         BE    ITS_SHAR                IT'S A SHARED VOLUME
         BXLE  R7,R8,NOT_ML11          GO HIT THE NEXT ENTRY
         B     NOTBLANK                NON-BLANK, NON-SHARED
         SPACE 1
ITS_SHAR DS    ØH
         SPACE 1
         MVC   S_GRP(L'LIT_SHAR),LIT_SHAR  MAKE IT SHARED VOL
         SPACE 1
NOTBLANK DS    ØH
         SPACE 1
         LM    R7,R9,FIRST@            SET UP FOR BXLE LOOP
```

```
         USING $S_GRP,R7               SO WE CAN SAVE DATA IN STRUCTURE
         SPACE 1
BXLE_L1  DS    ØH
         SPACE 1
         CLC   $S_GRP,S_GRP            Q. IS THIS THE GROUP
         BNE   BXLE_L2                 A. NO, BXLE TO NEXT ENTRY
         ICM   R5,B'1111',$V_FREE      GET THE TABLE VALUE
         ICM   R6,B'1111',V_FREE       GET THE RECORD VALUE
         AR    R5,R6                   ADD THEM TOGETHER
         STCM  R5,B'1111',$V_FREE      SAVE THEM IN THE TABLE
         ICM   R5,B'1111',$V_ALOC      GET THE TABLE VALUE
         ICM   R6,B'1111',V_ALOC       GET THE RECORD VALUE
         AR    R5,R6                   ADD THEM TOGETHER
         STCM  R5,B'1111',$V_ALOC      SAVE THEM IN THE TABLE
         ICM   R5,B'1111',$V_TOTL      GET THE TABLE VALUE
         ICM   R6,B'1111',V_TOTL       GET THE RECORD VALUE
         AR    R5,R6                   ADD THEM TOGETHER
         STCM  R5,B'1111',$V_TOTL      SAVE THEM IN THE TABLE
         B     R_LOOP
         DROP  R7
         SPACE 1
BXLE_L2  DS    ØH
         SPACE 1
         BXLE  R7,R8,BXLE_L1           Q. IS THERE ANOTHER ENTRY
         USING $S_GRP,R1Ø              LET ASSEMBLER KNOW
         MVC   $S_GRP,S_GRP            PUT NEW GROUP IN STRUCTURE
         MVC   $V_FREE,V_FREE          SAVE THE FREE SPACE
         MVC   $V_ALOC,V_ALOC          SAVE THE ALLOCATED SPACE
         MVC   $V_TOTL,V_TOTL          SAVE THE TOTAL SPACE
         ST    R1Ø,LAST@               SAVE LAST ACTIVE ENTRY ADDR.
         A     R1Ø,INC_V               BUMP TO NEXT INSERTION POINT
         B     R_LOOP                  GO GET A NEW DCOLLECT RECORD
         DROP  R1Ø
         SPACE 1
EOF_DCOL DS    ØH
         SPACE 1
         CLOSE (DCOLLECT)
         LM    R3,R5,FIRST@            SET UP FOR OUTER LOOP
         SPACE 1
BXLE_L22 DS    ØH
         SPACE 1
         CLC   =CL3Ø' ',Ø(R3)          Q. STORAGE GROUP = BLANKS
         BNE   BXLE_L33                A. NO, HIT THE BXLE
         MVC   Ø(L'NON_SMS,R3),NON_SMS MAKE IT NON SMS
         SPACE 1
BXLE_L33 DS    ØH
         SPACE 1
         BXLE  R3,R4,BXLE_L22
         LM    R3,R5,FIRST@            SET UP FOR OUTER LOOP
         S     R5,=AL4($LEN)           BUMP DOWN LAST ENTRY
         SPACE 1
BXLE_L3  DS    ØH
```

```
         SPACE 1
         LM    R8,R9,INC_V             SET UP FOR INNER LOOP
         LA    R7,$LEN(,R3)            MAKE R7 1 ENTRY > R3
         SPACE 1
BXLE_L4  DS    ØH
         SPACE 1
         CLC   Ø($LEN,R3),Ø(R7)        Q. COMPARE THE ENTRIES
         BNH   BXLE_L5                 A. R3 ENTRY < R7 ENTRY
         MVC   $TEMP($LEN),Ø(R3)       MOVE R3 ENTRY TO TEMP STORAGE
         MVC   Ø($LEN,R3),Ø(R7)        MOVE R7 ENTRY TO R3 LOCATION
         MVC   Ø($LEN,R7),$TEMP        MOVE TEMP STORAGE TO R7 LOCATION
         SPACE 1
BXLE_L5  DS    ØH
         SPACE 1
         BXLE  R7,R8,BXLE_L4           BUMP THE INNER LOOP
         BXLE  R3,R4,BXLE_L3           BUMP THE OUTER LOOP
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* OPEN AND WRITE THE HTML TEXT FILE.                                   *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
OPENHTML DS    ØH
         SPACE 1
         OPEN  (HTMLFILE,(OUTPUT))
         LA    R7,HEAD_ØØ1
         LA    R8,L'HEAD_ØØ1
         LA    R9,HEAD_ØØ8
         SPACE 1
BXLE_L6  DS    ØH
         SPACE 1
         PUT   HTMLFILE,Ø(R7)
         BXLE  R7,R8,BXLE_L6
         MVC   VT_ØØ1,HEAD_ØØ9
         LA    R4,VT_ØØ1+HEAD_Ø99
         L     R2,@PLIST
         LH    R3,Ø(R2)
         BCTR  R3,Ø
         EX    R3,MOV_PARM
         LA    R4,1(R3,R4)
         MVC   Ø(L'HEAD_999,R4),HEAD_999
         PUT   HTMLFILE,VT_ØØ1
         LA    R7,HEAD_Ø1Ø
         LA    R8,L'HEAD_Ø1Ø
         LA    R9,HEAD_Ø24
         SPACE 1
BXLE_L66 DS    ØH
         SPACE 1
         PUT   HTMLFILE,Ø(R7)
         BXLE  R7,R8,BXLE_L66
         MVC   VT_ØØ2,DT_ØØ2
         LM    R7,R9,FIRST@
         SPACE 1
```

```
BXLE_L7  DS    ØH
         SPACE 1
         USING $S_GRP,R7               SET A BASE FOR THE ASSEMBLER
         MVC   VT_ØØ1,DT_ØØ1
         MVC   VT_ØØ1+DT_Ø11(L'S_GRP),$S_GRP MOVE IN THE STG GROUP NAME
         PUT   HTMLFILE,VT_ØØ1
         ICM   R4,B'1111',$V_FREE      GET THE FREE KILOBYTES
         CVD   R4,DUBLWORK             CONVERT IT TO DECIMAL
         MVC   VT_ØØ2+DT_Ø22(L'EP_1),EP_1 MOVE THE EDIT PATTERN
         ED    VT_ØØ2+DT_Ø22(L'EP_1),DUBLWORK+2 UNPACK INTO PATTERN
         PUT   HTMLFILE,VT_ØØ2
         ICM   R4,B'1111',$V_ALOC      GET THE ALLOCATED KILOBYTES
         CVD   R4,DUBLWORK             CONVERT IT TO DECIMAL
         MVC   VT_ØØ2+DT_Ø22(L'EP_1),EP_1 MOVE THE EDIT PATTERN
         ED    VT_ØØ2+DT_Ø22(L'EP_1),DUBLWORK+2 UNPACK INTO PATTERN
         PUT   HTMLFILE,VT_ØØ2
         ICM   R4,B'1111',$V_TOTL     GET THE TOTAL KILOBYTES
         CVD   R4,DUBLWORK            CONVERT IT TO DECIMAL
         MVC   VT_ØØ2+DT_Ø22(L'EP_1),EP_1 MOVE THE EDIT PATTERN
         ED    VT_ØØ2+DT_Ø22(L'EP_1),DUBLWORK+2 UNPACK INTO PATTERN
         PUT   HTMLFILE,VT_ØØ2
         ICM   R5,B'1111',$V_ALOC     GET THE ALLOCATED AMOUNT
         XR    R4,R4                  CLEAR R4 FOR MULTIPLY
         LA    R3,1ØØ                 PUT 1ØØ IN R3
         MR    R4,R3                  DO THE MULTIPLY
         ICM   R3,B'1111',$V_TOTL     GET THE TOTAL AMOUNT
         DR    R4,R3                  DIVIDE
         LA    R3,1ØØ                 PUT 1ØØ IN R3
         SR    R3,R5                  DIFFERENCE IS THE FREE %
         CVD   R3,DUBLWORK            CONVERT IT TO DECIMAL
         MVC   VT_ØØ1,DT_ØØ3
         MVC   VT_ØØ1+DT_Ø33(L'EP_2),EP_2 MOVE THE EDIT PATTERN
         ED    VT_ØØ1+DT_Ø33(L'EP_2),DUBLWORK+6 UNPACK INTO PATTERN
         PUT   HTMLFILE,VT_ØØ1
         BXLE  R7,R8,BXLE_L7
         PUT   HTMLFILE,TAIL_ØØ3
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* GET THE DATE AND TIME FOR LAST LINE OF HTML TABLE, THEN CLOSE UP    *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
CLOSHTML DS    ØH
         SPACE 1
         MVC   VT_ØØ1,TAIL_ØØ2
         TIME  DEC,TIMEDATE,ZONE=LT,DATETYPE=MMDDYYYY,LINKAGE=SYSTEM
         XR    R4,R4
         ICM   R4,B'ØØ11',TIMEDATE
         STC   R4,VT_ØØ1+TAIL_222+4
         SRL   R4,4
         STC   R4,VT_ØØ1+TAIL_222+3
         SRL   R4,4
         STC   R4,VT_ØØ1+TAIL_222+1
```

```
         SRL   R4,4
         STC   R4,VT_001+TAIL_222+0
         NC    VT_001+TAIL_222(5),=XL5'0F0F0F0F0F'
         TR    VT_001+TAIL_222(5),=CL10'0123456789'
         MVI   VT_001+TAIL_222+2,C':'
         ICM   R4,B'1111',TIMEDATE+8
         STC   R4,VT_001+TAIL_022+9
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+8
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+7
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+6
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+4
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+3
         SRL   R4,4
         STC   R4,VT_001+TAIL_022+1
         SRL   R4,4
         STC   R4,VT_001+TAIL_022
         NC    VT_001+TAIL_022(10),=XL10'0F0F0F0F0F0F0F0F0F0F'
         TR    VT_001+TAIL_022(10),=CL10'0123456789'
         MVI   VT_001+TAIL_022+2,C'/'
         MVI   VT_001+TAIL_022+5,C'/'
         PUT   HTMLFILE,HEAD_007
         PUT   HTMLFILE,VT_001
         PUT   HTMLFILE,TAIL_003
         PUT   HTMLFILE,TAIL_004
         PUT   HTMLFILE,TAIL_005
         PUT   HTMLFILE,TAIL_006
         CLOSE (HTMLFILE)
         SPACE 1
EXIT_PGM DS    0H
         SPACE 1
         $ESAEPI RET_CODE
         TITLE 'STORGHTM - CONSTANTS AND LITERALS'
V_TYPE   DC    CL1'V'
NON_SMS  DC    CL7'NON-SMS'
EP_1     DC    XL15'4020206B2020206B2020206B202020'
EP_2     DC    XL04'40202021'
BLANK15  DC    CL15' '
LIT_MVS  DC    CL3'MVS'
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* FOLLOWING TABLE CONTAINS SHARED VOLUME NAMES OR MASKS               *
* NUMBER AND CONTENT OF THE MASKS WILL VARY BY INSTALLATION           *
* CONTENT WILL NEED TO BE CUSTOMIZED FOR YOUR ENVIRONMENT             *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
SHR_CTL  DC    A(LIT_SHRF)
         DC    A(12)
         DC    A(LIT_SHRE)
LIT_SHRF DC    F'5',CL7'MVSRS7'
```

```
         DC    F'4',CL8'RESS9'
         DC    F'4',CL8'RESM5'
         DC    F'4',CL8'MVS9S'
         DC    F'4',CL8'MVS3S'
         DC    F'4',CL8'MVS2S'
LIT_SHRE DC    F'4',CL8'MVS1S'
LIT_H    DC    CL1'H'
LIT_HSM  DC    CL7'HSM ML1'
LIT_SHAR DC    CL6'SHARED'
*
HEAD_ØØ1 DC    CL8Ø'<HTML>'
HEAD_ØØ2 DC    CL8Ø'<HEAD>'
HEAD_ØØ3 DC    CL8Ø'<TITLE>FREE SPACE REPORT</TITLE>'
HEAD_ØØ4 DC    CL8Ø'<BODY BACKGROUND="../IMAGES/BACKGRD.GIF">'
HEAD_ØØ5 DC    CL8Ø'<CENTER>'
HEAD_ØØ6 DC    CL8Ø'<TABLE BORDER=5 CELLPADDING=4 WIDTH="85%">'
HEAD_ØØ7 DC    CL8Ø'<TR>'
HEAD_ØØ8 DC    CL8Ø'<TD COLSPAN=5 ALIGN=CENTER><H2>'
*
HEAD_ØØ9 DC    C'<I>'
HEAD_Ø99 EQU   *-HEAD_ØØ9
         DC    (8Ø-(*-HEAD_ØØ9))CL1' '
HEAD_999 DC    CL21' FREESPACE REPORT</I>'
*
HEAD_Ø1Ø DC    CL8Ø'</TR>'
HEAD_Ø11 DC    CL8Ø'<TR>'
HEAD_Ø12 DC    CL8Ø'<TH ALIGN=CENTER>POOL</TH>'
HEAD_Ø13 DC    CL8Ø'<TH ALIGN=CENTER>TOTAL</TH>'
HEAD_Ø14 DC    CL8Ø'<TH ALIGN=CENTER>TOTAL</TH>'
HEAD_Ø15 DC    CL8Ø'<TH ALIGN=CENTER>TOTAL</TH>'
HEAD_Ø16 DC    CL8Ø'<TH ALIGN=CENTER>PERCENT</TH>'
HEAD_Ø17 DC    CL8Ø'</TR>'
HEAD_Ø18 DC    CL8Ø'<TR>'
HEAD_Ø19 DC    CL8Ø'<TH ALIGN=CENTER>NAME</TH>'
HEAD_Ø2Ø DC    CL8Ø'<TH ALIGN=CENTER>AVAILABLE(KB)</TH>'
HEAD_Ø21 DC    CL8Ø'<TH ALIGN=CENTER>ALLOCATED(KB)</TH>'
HEAD_Ø22 DC    CL8Ø'<TH ALIGN=CENTER>CAPACITY(KB)</TH>'
HEAD_Ø23 DC    CL8Ø'<TH ALIGN=CENTER>FREE</TH>'
HEAD_Ø24 DC    CL8Ø'</TR>'
TAIL_ØØ3 DC    CL8Ø'</TR>'
TAIL_ØØ4 DC    CL8Ø'</TABLE>'
TAIL_ØØ5 DC    CL8Ø'</BODY>'
TAIL_ØØ6 DC    CL8Ø'</HTML>'
TAIL_ØØ2 DC    C'<TD COLSPAN=5 ALIGN=CENTER><I>DATA IS CURRENT AS OF '
TAIL_Ø22 EQU   *-TAIL_ØØ2
         DC    C'MM/DD/YYYY'
         DC    C' AT '
TAIL_222 EQU   *-TAIL_ØØ2
         DC    CL5'HH:MM'
         DC    C'</I></TD>'
         DC    (8Ø-(*-TAIL_ØØ2))CL1' '
DT_ØØ1   DC    C'<TR><TD>'
```

```
DT_Ø11   EQU   *-DT_ØØ1
         DC    CL3Ø'                                '
DT_111   DC    C'</TD>',(8Ø-(*-DT_ØØ1))CL1' '
*
DT_ØØ2   DC    C'<TD ALIGN=RIGHT>'
DT_Ø22   EQU   *-DT_ØØ2
         DC    CL15'123456789Ø12345'
DT_222   DC    C'</TD>',(8Ø-(*-DT_ØØ2))CL1' '
*
DT_ØØ3   DC    C'<TD ALIGN=RIGHT>'
DT_Ø33   EQU   *-DT_ØØ3
         DC    (L'EP_2)CL1' '
DT_333   DC    C'</TD>',(8Ø-(*-DT_ØØ3))CL1' '
*
         DS    ØH
MOV_PARM MVC   Ø(*-*,R4),2(R2)
         SPACE 1
DCO_DCBE DCBE  RMODE31=BUFF,EODAD=EOF_DCOL
         SPACE 1
DCO_HTML DCBE  RMODE31=BUFF
         SPACE 1
DCOLLECT DCB DDNAME=DCOLLECT,DSORG=PS,RECFM=VB,MACRF=(GM),
               DCBE=DCO_DCBE
         SPACE 1
HTMLFILE DCB DDNAME=HTMLFILE,DSORG=PS,RECFM=FB,MACRF=(PM),
               LRECL=8Ø,DCBE=DCO_HTML
         $ESASTG
DUBLWORK DS    D                      WORK AREA FOR NUMBER CONVERSION
RET_CODE DS    F                      RETURN CODE
@PLIST   DS    F                      PLIST ADDRESS
*
TIMEDATE DS    XL16                   TIME AND DATE RETURNED HERE
*
VT_ØØ1   DS    XL8Ø                   OUTPUT LINE 1
VT_ØØ2   DS    XL8Ø                   OUTPUT LINE 2
         SPACE 1
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
* SIMPLE RECORD LAYOUT FOR THE DCOLLECT INFORMATION                    *
*----+----+----+----+----+----+----+----+----+----+----+----+----+----*
         SPACE 1
V_REC    DS    ØF
         DS    XL4                    SPACE FOR THE RDW
         DS    XL4                    FILLER
RECTYPE  DS    XL1                    RECORD TYPE
         DS    XL19                   FILLER
VOLUME   DS    XL6
V_STAT   DS    XL1                    VOLUME STATUS
         DS    XL5                    FILLER
V_FREE   DS    XL4                    FREE SPACE
V_ALOC   DS    XL4                    ALLOCATED SPACE
V_TOTL   DS    XL4                    TOTAL SPACE
         DS    XL34                   FILLER
```

```
S_GRP    DS    XL3Ø                    STORAGE GROUP NAME
         DS    (264-(*-V_REC))XL1      FILL OUT THE RECORD
         SPACE 1
NUM_ENT  EQU   5Ø                      CURRENTLY SET TO 5Ø STORAGE GRPS
FIRST@   DS    F                       ADDRESS OF FIRST ENTRY
INC_V    DS    F                       SIZE OF EACH ENTRY
LAST@    DS    F                       ADDRESS OF THE LAST ENTRY
$S_GRP   DS    XL3Ø                    STORAGE GROUP
$V_FREE  DS    XL4                     FREE STORAGE IN KILOBYTES
$V_ALOC  DS    XL4                     ALLOCATED STORAGE IN KILOBYTES
$V_TOTL  DS    XL4                     TOTAL STORAGE IN KILOBYTES
$LEN     EQU   *-$S_GRP                LET ASSEMBLER CALC. ENTRY SIZE
         DS    ((NUM_ENT-1)*$LEN)XL1   LET ASSEMBLER BUILD REMAINDER
$TEMP    DS    ($LEN)XL1               LET ASSEMBLER BUILD THE ENTRY
         END   STORGHTM
```

$ESAPRO MACRO

```
         MACRO
&LABEL   $ESAPRO &AM=31,&RM=ANY,&MODE=P
.*************************************************************************
.*       THIS MACRO WILL PROVIDE ENTRY LINKAGE AND OPTIONALLY
.*       MULTIPLE BASE REGISTERS.  TO USE THIS MACRO, YOU NEED TO
.*       ALSO USE THE $ESASTG MACRO.  THE $ESASTG DEFINES THE SYMBOL
.*       QLENGTH WHICH OCCURS IN THE CODE THAT &ESAPRO GENERATES.
.*       IF YOU DO NOT CODE ANY OPERANDS, THEN REGISTER 12 WILL BE
.*       USED AS THE BASE.  IF YOU CODE MULTIPLE SYMBOLS, THEN THEY
.*       WILL BE USED AS THE BASE REGISTERS.
.*
.*       EXAMPLES:
.*
.*              SECTNAME $ESAPRO          = REG 12 BASE
.*              SECTNAME $ESAPRO 5        = REG 5 BASE
.*              SECTNAME $ESAPRO R1Ø,R11  = REGS 1Ø AND 11 ARE BASES
.*************************************************************************
         LCLA  &AA,&AB,&AC
*
RØ       EQU   Ø
R1       EQU   1
R2       EQU   2
R3       EQU   3
R4       EQU   4
R5       EQU   5
R6       EQU   6
R7       EQU   7
R8       EQU   8
R9       EQU   9
R1Ø      EQU   1Ø
RA       EQU   1Ø
R11      EQU   11
RB       EQU   11
```

```
R12      EQU   12
RC       EQU   12
R13      EQU   13
RD       EQU   13
R14      EQU   14
RE       EQU   14
R15      EQU   15
RF       EQU   15
*
FPRØ     EQU   Ø
FPR2     EQU   2
FPR4     EQU   4
FPR6     EQU   6
*
&LABEL   CSECT
&LABEL   AMODE &AM
&LABEL   RMODE &RM
*
         SYSSTATE ASCENV=&MODE         SET THE ENVIRONMENT
*
         B     $$$$EYEC-*(R15)         BRANCH AROUND EYECATCHER
         DC    AL1(($$$$EYEC-*)-1)     EYECATCHER LENGTH
         DC    CL8'&LABEL'             MODULE ID
         DC    CL3' - '
         DC    CL8'&SYSDATE'           ASSEMBLY DATE
         DC    CL3' - '
         DC    CL8'&SYSTIME'           ASSEMBLY TIME
         DC    CL3'   '                FILLER
*
$$$$F1SA DC    CL4'F1SA'               USED FOR STACK OPERATIONS
$$$$4Ø96 DC    F'4Ø96'                 USED TO ADJUST BASE REGS
*
$$$$EYEC DS    ØH
*
         BAKR  R14,Ø                   SAVE GPRS AND ARS ON THE STACK
         AIF   (N'&SYSLIST EQ Ø).USER12
         LAE   &SYSLIST(1),Ø(R15,Ø)    LOAD OUR BASE REG
         USING &LABEL,&SYSLIST(1)      LET THE ASSEMBLER KNOW
         AGO   .GNBASE
.USER12  ANOP
         MNOTE *,'NO BASE REG SPECIFIED, REGISTER 12 USED'
         LAE   R12,Ø(R15,Ø)            LOAD OUR BASE REG
         USING &LABEL,R12              LET THE ASSEMBLER KNOW
         AGO   .STGOB
.GNBASE  ANOP
         AIF   (N'&SYSLIST LE 1).STGOB
&AA      SETA  2
&AC      SETA  4Ø96
.GNBASE1 ANOP
*
         AIF   (&AA GT N'&SYSLIST).STGOB
&AB      SETA  &AA-1
```

```
         LR    &SYSLIST(&AA),&SYSLIST(&AB) GET INITIAL BASE
         A     &SYSLIST(&AA),$$$$4Ø96      ADJUST NEXT BASE
         USING &LABEL+&AC,&SYSLIST(&AA)    LET THE ASSEMBLER KNOW
&AA      SETA  &AA+1
&AC      SETA  &AC+4Ø96
         AGO   .GNBASE1
.STGOB   ANOP
*
         L     RØ,QLENGTH              GET THE DSECT LENGTH
*
         STORAGE OBTAIN,LENGTH=(RØ),LOC=(RES,ANY)
*
         LR    R15,R1                  GET @(OBTAINED AREA)
         L     R13,QDSECT              GET DISPLACEMENT INTO AREA
         LA    R13,Ø(R13,R15)          GET @(OBTAINED AREA)
         LR    RØ,R13                  SET REG Ø = REG 13
         L     R1,QLENGTH              GET THE LENGTH OF THE AREA
         XR    R15,R15                 CLEAR REG 5
         MVCL  RØ,R14                  INTIALIZE THE AREA
         MVC   4(4,R13),$$$$F1SA       INDICATE STACK USAGE
         USING DSECT,R13               INFORM ASSEMBLER OF BASE
.MEND    ANOP
*
         EREG  R1,R1                   RESTORE REGISTER 1
         MEND
```

$ESAEPI MACRO

```
         MACRO
         $ESAEPI
.*************************************************************************
.*       THIS MACRO WILL PROVIDE EXIT LINKAGE. IT WILL FREE THE
.*       STORAGE AREA THAT WAS ACQUIRED BY THE $ESAPRO MACRO.  YOU
.*       CAN OPTIONALLY PASS IT A RETURN CODE VALUE.  THIS VALUE IS
.*       EITHER THE LABEL OF A FULL WORD IN STORAGE, OR IT IS A REG-
.*       ISTER. AS WITH THE $ESAPRO MACRO, YOU NEED TO USE THE $ESASTG
.*       MACRO.  THE SYMBOL QLENGTH WHICH OCCURS IN THE CODE THAT IS
.*       GENERATED BY THIS MACRO IS DEFINED BY $ESASTG
.*
.*       EXAMPLES:
.*
.*              $ESAEPI          = NO RETURN CODE SPECIFIED
.*              $ESAEPI (R5)     = RETURN CODE IS IN REG 5
.*              $ESAEPI RETCODE  = RETURN CODE IS IN THE FULLWORD AT
.*                                 RETCODE
.*************************************************************************
         AIF   (N'&SYSLIST EQ Ø).STGFRE
*
         AIF   ('&SYSLIST(1)'(1,1) EQ '(').REGRC
         L     R2,&SYSLIST(1)          GET RETURN CODE VALUE
```

```
         AGO   .STGFRE
.REGRC   ANOP
         LR    R2,&SYSLIST(1,1)        GET RETURN CODE VALUE
.STGFRE  ANOP
         L     RØ,QLENGTH              GET THE DSECT LENGTH
         STORAGE RELEASE,LENGTH=(RØ),ADDR=(R13)
         AIF   (N'&SYSLIST NE Ø).SETRC
         XR    R15,R15                 CLEAR THE RETURN CODE
         AGO   .MEND
.SETRC   ANOP
         LR    R15,R2                  SET THE RETURN CODE
.MEND    ANOP
         PR                            RETURN TO CALLER
* FOR ADDRESSABILITY PURPOSES
         LTORG
         MEND
```

$ESASTG MACRO

```
         MACRO
         $ESASTG
.*************************************************************************
.*       THIS MACRO IS USED IN CONJUNCTION WITH THE $ESAEPI AND $ESAPRO
.*       MACROS.  IT PROVIDES A Q TYPE ADDRESS CONSTANT WHICH WILL CON-
.*       THE LENGTH OF THE DSECT.  A REGISTER SAVE AREA ID PROVIDED AS
.*       WELL.
.*
.*       EXAMPLES:
.*              $ESASTG
.*      XXX     DC    F          = DEFINE ADDITIONAL STORAGE AREA
.*      YYY     DC    XL255
.*       .       .      .
.*       .       .      .
.*       .       .      .
.*************************************************************************
RC0000   DC    F'Ø'                    USED TO SET RETURN CODES
RC0004   DC    F'4'                    USED TO SET RETURN CODES
RC0008   DC    F'8'                    USED TO SET RETURN CODES
RC000C   DC    F'12'                   USED TO SET RETURN CODES
RC0010   DC    F'16'                   USED TO SET RETURN CODES
QDSECT   DC    Q(DSECT)                DEFINE A QCON
QLENGTH  CXD                           LET ASM CALCULATE THE LENGTH
DSECT    DSECT
         DS    18F                     SET ASIDE REGISTER SAVE AREA
         MEND
```

Automated and interactive library updates

INTRODUCTION

There is often a need to change strings in a library in an MVS environment. However, when it comes to changing a specific string in libraries with many members, it would be a very hard task to do without using an edit macro.

For example, let us assume that we have several JCL libraries related to a product, and members of these libraries are referring to a back-level product qualifier, which is BBIOA.V514 and all these occurrences have to be substituted with BBIOA.V611, which is the new version qualifier.

I have developed a small utility which consists of REXX programs, edit macros**,** several panels, and a message library member to automate this sort of task, in a panel-driven and interactive way.

WHAT IS THE ALGORITHM?

The edit macro used in this article is run against all members of a PDS. A loop in the REXX program uses a LISTDS command output to obtain the PDS members' names. For each member, an 'ISPEXEC EDIT' command is issued with the initial 'MACRO' keyword (Ispexec Edit Dataset(...) Macro(...) command).

When the main REXX CHGREXX is called, it will display the panel PANEL0 to get all 'Change' command parameters from the user, including the PO dataset to be updated (It is called as SOURCE dataset), 'from' string, 'to' string, etc. Then, a back-up copy of the SOURCE dataset is created for backout purposes. (It is called as BKPCPY dataset.)

Later on, CHGREXX reads all members of the SOURCE dataset and calls the edit macro CHGMAC within an 'ISPEXEC EDIT DATASET' command for each member read. If that member has at least one occurrence of the 'from' string, then those strings are changed and a 'Save' operation is performed. If the member is saved successfully,

ISPF statistics for that member are changed as well. This is because the 'Save' command in the edit macro sets the ID field (the user ID that last modified the data) as the TSO userid. But the REXX CHGREXX sets the UID field as CHGMAC, and in this way at a later time the user can easily distinguish which members were changed by the edit macro while editing the dataset.

Then the result of the edit macro execution are written into an ISPF table, member by member, which will be displayed in Panel 3 in the next step.

Then CHGREXX compares the SOURCE dataset, which may have been updated, with the BKPCPY dataset, which is the original version of the SOURCE dataset, using the SuperC utility, and displays the differences found to the user. It shows user differences between the initial and the final contents of the SOURCE dataset. Finally Panel 2 is displayed and the user is asked if she/he accepts the changes just made by the utility. Depending on the response, the BKPCPY dataset is kept or deleted. Panel 1 displays an explanatory message to the user during the execution of the utility. There is also a help panel, which is Panel H.

HOW TO RUN THE REXX CHGREXX

To make the use of this application easy, I recommend putting all the REXX code, panels, the editmacros, and the message member into a single library. In this article, all of them are members of the dataset EXP.CTM.REXX. Then it is sufficient to call, CHGREXX in the following way:

```
TSO EX 'EXP.CTM.REXX(CHGREXX)'
```

The REXX CHGMAC will do all necessary allocations for ISPF message, panels, the REXX, and the macro by using a series of ALTLIB and LIBDEF commands.

Another way, of course, is to put all panel definition members into an ISPPLIB, the REXX and the edit macro into a SYSEXEC, and the message member into an ISPMLIB concatenation library. Note that, if this method is chosen, all ALTLIB and LIBDEF commands, which are not necessary, have to be removed from the REXX CHGREXX. Then you can call the REXX simply by its name:


```
TSO %CHGREXX
```

But for easy maintenance, I would recommend using the first approach.

A SHORT OVERVIEW ON EDIT MACROS

You can use edit macros, which look like ordinary editor commands, to extend and customize the editor. You create an edit macro by placing a series of commands in a dataset or member of a partitioned dataset. Then you can run those commands as a single macro by typing the defined name in the 'command' line in an edit session or you call an edit macro from a REXX. The second method not only provides us with the capability of running an edit macro many times in a loop, but also save us time by not having to edit each member to be able to run the edit macro against that member.

Edit macros can be either CLISTs or REXX EXECs written in the CLIST or REXX command language, or program macros written in a programming language (such as FORTRAN, PL/I, or COBOL).

Edit macros have access to the dialog manager and system services. Because edit macros are CLISTs, REXX EXECs, or programs, they have unlimited possibilities. Edit macros can be used to:

- Perform repeated tasks
- Simplify complex tasks
- Pass parameters
- Retrieve and return information.

REXX edit macros must include a REXX comment line (ie /* REXX */) as the first line of each edit macro to distinguish them from CLIST edit macros. This comment line can contain other words or characters if necessary, but it must include the string REXX.

REXX edit macros must be in partitioned datasets. REXX edit macros can exist in the following concatenations: SYSUEXEC, ALTLIB (for datasets activated as EXECs), and SYSEXEC. Datasets in these concatenations can contain only REXX EXECs.A REXX edit macro is made up of REXX statements. Each statement falls into one of the following categories:

- Edit macro commands
- CLIST or REXX command procedure statements and comments
- ISPF and PDF dialog service requests
- TSO commands.

You can run PDF edit macros in batch by submitting JCL which allocates all of the necessary ISPF libraries and runs a command that calls the EDIT service with an initial macro. This initial macro can do anything that can be done by an initial macro in an interactive session. However, in batch, the macro should end with an ISREDIT END or ISREDIT CANCEL statement. These statements ensure that no attempt is made to display the edit screen in batch.

We can pass parameters to an edit macro. A parameter can be either a simple string or a quoted string. It can be passed by using the standard method of putting variables into shared and profile pools (use VPUT in dialogs and VGET in initial macros and *vice versa*). This method is best suited to parameters passed from one dialog to another, as in an edit macro. The edit macro and the REXX used in this utility use the VGET/VPUT commands to communicate with one another. For more information on edit macros see the IBM book *ISPF Edit and Edit Macros*.

SOME NOTES ON THE UTILITY

A 'From' string and a 'To' string have to be specified in Panel 1 of the application. If the string is a simple or delimited string (a string that begins and ends with apostrophes or quotes), the characters are treated as being both upper and lowercase even if caps mode is off. For example, this command:

```
CHANGE ALL 'CONDITION NO. 1' '........'
```

successfully changes the following:

- CONDITION NO. 1
- Condition No. 1
- condition no. 1
- coNDitION nO. 1

Also, all of the following commands have the same effect:

```
CHANGE 'Edit Commands' '........'
CHANGE 'EDIT COMMANDS' '........'
CHANGE 'edit commands' '........'
```

For this reason, if you want the change command to be satisfied by an exact character-by-character match, lowercase alphabetic characters match only with lowercase alphabetic characters, and uppercase alphabetic characters match only with uppercase, a character string must be used. For example, to change ‘aBc’ to ‘AbC’ you have to use the command:

```
Change C'aBc' 'AbC' ALL
```

Besides character strings, you can use picture strings to change a particular kind of character without regard for the specific character involved. For example to change any lower case character to upper case, use the command:

```
CHG p'<' p'>'
```

To change a string of hexadecimal digits, you can use a hex string, for example, CHG 'c1c2'x 'a1a2'x.

To change a character string regardless of whether alphabetic characters are upper or lower case, use the a text string. For example, the following command changes the text ‘spf’ to caps:

```
CHG t'spf' SPF
```

To limit the strings that are found, you can use qualifying parameters. This way, you will specify additional characteristics of string-1 (from string) by using the operands PREFIX, SUFFIX, CHARS, and WORD.

CHARS – Locates string-1 anywhere the characters match. This is the default.

PREFIX – Locates string-1 at the beginning of a word.

SUFFIX – Locates string-1 at the end of a word.

WORD – String-1 is delimited on both sides by blanks or other non-alphanumeric characters.

The col-1 and col-2 operands allow the user to search only a portion of each line, rather than the entire line. These operands, which are numbers separated by a comma or by at least one blank, show the starting and ending columns for the search. The following rules apply:

- If you specify neither col-1 nor col-2, the search continues across all columns within the current boundary columns (BOUNDS line).
- If you specify col-1, the editor finds the string only if the string starts in the specified column.
- If you specify both col-1 and col-2, the editor finds the string only if it is entirely within the specified columns.
- If the second column specified is larger than the record size, this is an error condition and the edit macro of the utility (CHGMAC) will substitute the second column with the record length of the dataset being changed.

ADDITIONAL NOTES

The edit-macro and REXX program can be used in the migration processes. For example, it is helpful to make multiple changes in the libraries when a new version of a product is implemented. Also you can make the edit macro a little more complex, by using more edit primary commands such as ‘change’ or ‘exclude’. For this, Panel 1 has to accommodate the changes made to the edit macro since they work together. For example, to incorporate one more ‘change’ command, you would have to prepare a continuation panel to Panel 1.

Another advantage of using this utility is that it permits us to change strings of up to 255 characters in length. Remember that an ordinary change command on the ISPF edit command line is not able to do that because of the limitation of the edit command line.

If it seems that the PO dataset has too many members and most of them are candidates to have updates when the REXX CHGREXX runs, special care must be taken toward S37 abends. The REXX CHGREXX compresses the SOURCE dataset at the beginning of the utility as a precaution. For this reason, it’s recommended that you enlarge the PO dataset big enough to tolerate possible growth resulting for many

possible ‘Isredit Save’ commands in the edit macro (CHGMAC). However, if you see an S37 abend condition on the result panel Panel 3 at the end of the execution, you can compress the source dataset in another split-screen and re-execute the utility.

If it is not necessary to work interactively, you can make changes in the PO libraries without using a panel as well. On the other hand, you can make changes in several PO datasets at the same time. The REXX program BATCHREX and the edit mcro BATCHMAC are given as examples with abridged code, and do not contain any error recovery controls. To execute, use the command:

```
TSO EX 'EXP.CTM.REXX(BATCHREX)'
```

REXX : CHGREXX

```
/* REXX */
/*———————————————————————————————————————————————————————————*/
/* AUTOMATED AND INTERACTIVE LIBRARY UPDATE BY USING EDIT MACRO      */
/* REXX program       : ChgRexx                                      */
/* Edit macro called  : ChgMac                                       */
/*———————————————————————————————————————————————————————————*/
Status = Msg('Off')
/*———————————————————————————————————————————————————————————*/
/* Let exec process errors. This way we can check the Return Codes   */
/* and take the appropriate actions.                                 */
/*———————————————————————————————————————————————————————————*/
"Ispexec Control Errors Return"

/*———————————————————————————————————————————————————————————*/
/* Make necessary library allocations for REXX/Panel/Message members.*/
/*———————————————————————————————————————————————————————————*/
"Altlib  Activate Application(Exec) Da(Exp.Ctm.Rexx)"
"Ispexec Libdef Ispplib Dataset     Id(Exp.Ctm.Rexx)"
"Ispexec Libdef Ispmlib Dataset     Id(Exp.Ctm.Rexx)"

/*———————————————————————————————————————————————————————————*/
/* Clean some profile variables.                                     */
/*———————————————————————————————————————————————————————————*/
"Ispexec Verase (Mes,Mem,Col1,Col2,Rc1,Rc2,Rc3,Cnt,Chg,Err,Msg1,Msg2)"

REPEAT:
/*———————————————————————————————————————————————————————————*/
/* Display the panel Panel0 to get the dataset name, and "Ispf Edit  */
/* command" parameters.                                              */
/*———————————————————————————————————————————————————————————*/
```

```
"Ispexec Display Panel(Panel0)"

/*-------------------------------------------------------------------------*/
/* Control that if PF03 or PF04 key is pressed on the Panel0. If so, */
/* deallocate the libraries allocated by Libdef & Altlib commands.   */
/*-------------------------------------------------------------------------*/
"Ispexec Vget (Spfkey,Dsn,Col1,Col2,From,To) Profile"
If (Spfkey = PF03 3 Spfkey = PF04) Then Do
                       "Ispexec Setmsg Msg(Edtm006)"
                       "Altlib Deactivate Application(Exec)"
                       "Ispexec Libdef Ispplib"
                       "Ispexec Libdef Ispmlib"
                       Exit
                      End

/*-------------------------------------------------------------------------*/
/* Check whether dset name entered on Panel0 is an existing PO dset. */
/* From now on, we will be calling this dataset 'SOURCE dset.'       */
/*-------------------------------------------------------------------------*/
IF Sysdsn(Dsn) = 'OK' Then
             Do
              X = Listdsi(Dsn)
              If X ¬= 0 Then
                Say 'Some Listdsi info not available. Function code =' X
                Else Do
                      Dsorg   = Sysdsorg
                      If Dsorg ¬= PO Then
                                    Do
                                      "Ispexec Setmsg Msg(Edtm009D)"
                                      Signal Repeat
                                    End
                     End
             End
             Else Do
                   "Ispexec Setmsg Msg(Edtm009C)"
                   Signal Repeat
                  End

/*-------------------------------------------------------------------------*/
/* Check whether the SOURCE DSET has any members?                    */
/*-------------------------------------------------------------------------*/
 x = Outtrap('Var.')
 "Listds" Dsn "Members"
 x = Outtrap('Off')        /* Turns trapping OFF */
  Cnt = Var.0-6
  "Ispexec Vput Cnt Profile"
  If Cnt = 0 Then Do
                   "Ispexec Setmsg Msg(Edtm009B)"
                   Signal Repeat
                  End
```

```
/*-------------------------------------------------------------------------*/
/* Check whether From or To string entered on Panel0 are                   */
/* greater than the record length of the SOURCE dataset.                   */
/*-------------------------------------------------------------------------*/
 L2    = Syslrecl
 If Length(From) > L2    Then Do
                                   L1 = Length(From)
                                   "Ispexec Vput (L1,L2) Profile"
                                   "Ispexec Setmsg Msg(Edtm008)"
                                   Signal Repeat
                              End
 If Length(To)   > L2    Then Do
                                   L1 = Length(To)
                                   "Ispexec Vput (L1,L2) Profile"
                                   "Ispexec Setmsg Msg(Edtm009)"
                                   Signal Repeat
                              End

/*-------------------------------------------------------------------------*/
/* Check whether Col1 or Col2 is greater then the Lrecl of the             */
/* SOURCE dataset. If so, do not continue any more. In addition,           */
/* display the Lrecl of the SOURCE dset for informative purposes.          */
/*-------------------------------------------------------------------------*/
  If (Col2 > L2) 3 (Col1 > L2)  Then Do
                                     "Ispexec Setmsg Msg(Edtm009J)"
                                     Mes = '*** Lrecl : 'L2
                                     "Ispexec Vput Mes Profile"
                                     Signal Repeat
                                   End

/*-------------------------------------------------------------------------*/
/* Compress the SOURCE dataset to prevent it from producing S37 abend*/
/*-------------------------------------------------------------------------*/
Chgiebcp Dsn Dsn

/*-------------------------------------------------------------------------*/
/* Allocate a back-up copy of the SOURCE dataset. From now on we           */
/* will be calling it BKPCPY dataset.                                      */
/*-------------------------------------------------------------------------*/
Dsn_bkp = Dsn'.BKP'
 "Alloc Da("Dsn_bkp") Like("Dsn")"
 If Rc ¬=0 Then
            Do
             Say 'BKPCPY dset allocation is not successful'
             If Rc =12 Then Say 'The dset,' Dsn_bkp 'already exists.',
                              'Please check it out.'
              Exit
             End
```

```
 "Free Da("Dsn_bkp")"   /*   Free BKPCPY dataset. */

/*-------------------------------------------------------------------------*/
/* Call the system utility IEBCOPY to copy the SOURCE dataset              */
/* members into the BKPCPY dataset.                                        */
/*-------------------------------------------------------------------------*/

Chgiebcp Dsn Dsn_bkp

/*-------------------------------------------------------------------------*/
/* At this point, we have BKPCPY dataset built. So we can start            */
/* executing the Edit macro over the SOURCE dataset members to do          */
/* batch string updates.                                                   */
/*-------------------------------------------------------------------------*/

/*-------------------------------------------------------------------------*/
/* A table which will show edit-macro execution status for each            */
/* member of the SOURCE dataset wll be created.                            */
/*                                                                         */
/* Table_output_library = Ispf profile dataset.(File Name = ISPTABL)       */
/* Table_name = TABLExxx (xxx = 1,...,100 )                                */
/*                                                                         */
/* A user can ask to run this application more than one on a split         */
/* screen. Each time a different table name will be generated.             */
/*-------------------------------------------------------------------------*/

 Prfxusr  = Sysvar(sysuid)
 "Alloc Fi(Isptabl) Da('"33 Prfxusr 33 ".Ispf.Ispprof') Shr"

 Ran = Random(1,100)
 Table_name = 'TABLE' 33 ran
 "Ispexec Tbcreate" Table_name "Names(Mem Chg Err Msg1 Rc1 Rc2 Msg2)"

/*-------------------------------------------------------------------------*/
/* Get members of SOURCE dset and execute the edit macro in a loop.        */
/*-------------------------------------------------------------------------*/
 Do i = 7  To Var.0    /* Loop-Martapv */
   Mem = Strip(Var.i)
   Call Editmac
   Rc1 = Result
   "Ispexec Vput (Mem,Rc1) Profile"
   "Ispexec Vget (Chg,Err)      Profile"
   "Ispexec Vget (Msg2,Rc2,Rc3) Profile"  /*  Get "ISPEXEC EDIT" and */
                                          /*  "SAVE" command Rc.     */
   Select
    When (Rc1 = 0 )           Then Msg1= 'Member updated.  '
    When ((Rc1=4) & (Rc3>=4)) Then Msg1= 'No save.Abend S37'
    When (Rc1 = 4 )           Then Msg1= 'Member not saved.'
    When (Rc1 = 14)           Then Msg1= 'Member in use.   '
    When (Rc1 = 20)           Then Msg1= 'Severe error.    '
```

```
    Otherwise                 Nop
   End

   "Ispexec Vput Msg1 Profile"

/*-------------------------------------------------------------------------*/
/* Set the ISPF Statistics for the SOURCE dataset members that are   */
/* updated by the edit macro. (Changed members will have the "CHGMAC"*/
/* string in their ID field.)                                        */
/*-------------------------------------------------------------------------*/
If Rc3 = Ø Then
     Do
      "Alloc Fi(Stats)  Da("Dsn")     Shr Reuse"
      "Ispexec Lminit   Dataid(Sta)   Ddname(Stats) Enq(Shr)"
      "Ispexec Lmmstats Dataid("Sta") Member("Mem") User(Chgmac)"
      "Ispexec Lmfree   Dataid("Sta")"
     End

/*-------------------------------------------------------------------------*/
/* Add a new row to the "Edit-macro Result Table".                   */
/*-------------------------------------------------------------------------*/
  "Ispexec Tbadd" Table_name

 End  /* End-of-Loop-Martapv */

/*-------------------------------------------------------------------------*/
/*  At last, display the "Edit-macro Result Table" and delete the    */
/*  virtual storage copy of this table.                              */
/*-------------------------------------------------------------------------*/
    "Ispexec Tbtop"   Table_name
    "Ispexec Tbdispl" Table_name "Panel(Panel3)"
    "Ispexec Tbend"   Table_name

/*-------------------------------------------------------------------------*/
/* Compare the SOURCE and BKPCPY datasets with the SuperC utility.   */
/*-------------------------------------------------------------------------*/

"Alloc Fi(Newdd) Da("Dsn")     Shr Reuse"
"Alloc Fi(Olddd) Da("Dsn_bkp") Shr Reuse"
"Alloc Fi(Outdd) Space(1,1) Cylinders New Keep Dsorg(Ps)"

/*-------------------------------------------------------------------------*/
/* If RC is zero, it means that no change has been made to the       */
/* SOURCE dataset. So we can delete the BKPCPY dataset and terminate */
/* the dialog. If RC is not zero, in this case this means that some  */
/* changes have been made to SOURCE dataset.                         */
/*-------------------------------------------------------------------------*/

Address Tso "Call 'Isp.Sisplpa(Isrsupc)' 'LONGL,LINECMP'"
 If Rc = Ø Then
```

```
               Do
                  "Ispexec Setmsg Msg(Edtm009I)"
                  Delete Dsn_bkp
                  Signal De_Alloc
               End

"Ispexec Lminit Dataid(Villar) Ddname(Outdd) Enq(Shr)"
 Rc7 = Rc
 If Rc7 ¬= 0 Then
               Do
                  "Ispexec Vput Rc7 Profile"
                  "Ispexec Setmsg Msg(Edtm009G)"
                 Exit
               End

/*-------------------------------------------------------------------------*/
/* Call POPUP1 proc to display an explanatory message.                     */
/*-------------------------------------------------------------------------*/
Call Popup1

/*-------------------------------------------------------------------------*/
/* Display the SuperC output to the user.                                  */
/*-------------------------------------------------------------------------*/
"Ispexec Browse Dataid("Villar")"

/*-------------------------------------------------------------------------*/
/* Call POPUP2 to ask the user if he/she agrees with the changes on  */
/* SOURCE dataset. If he/she does, BKPCPY dataset will be deleted.   */
/*-------------------------------------------------------------------------*/
Call Popup2

De_Alloc:
"Ispexec Lmfree Dataid("Villar")"
"Free File(Newdd,Olddd,Outdd)"
"Altlib Deactivate Application(Exec)"
"Ispexec Libdef Ispplib"
"Ispexec Libdef Ispmlib"

 EXIT         /* End-of-the-main-Rexx */

POPUP1:
 Zwinttl =
 "Ispexec Addpop Row(6) Column(25)"
 "Ispexec Display Panel(Panel1)"
 "Ispexec Vget Spfkey Profile"
 "ispexec Rempop"
RETURN /* End-of-the-procedure-POPUP1 */

POPUP2:
 Zwinttl = CONFIRMATION SCREEN
```



```
 "Ispexec Addpop Row(6) Column(25)"
 "Ispexec Display Panel(Panel2)"
 "Ispexec Vget Spfkey Profile"

/*———————————————————————————————————————————————————————————*/
/* Keep displaying the Panel2 until the user enters 'Y' or 'N'.      */
/*———————————————————————————————————————————————————————————*/
 Do While( Spfkey=PFØ3 3 Spfkey = PFØ4 )
   "Ispexec Setmsg Msg(Edtm00ØA)"
   "ispexec Rempop"
   "Ispexec Browse Dataid("Villar")"
   "Ispexec Addpop Row(6) Column(25)"
   "Ispexec Display Panel(Panel2)"
   "Ispexec Vget Spfkey Profile"
 End
 "ispexec Rempop"

 "Ispexec Vget Resp Profile"
 If Resp = 'Y' Then Do
                       Delete Dsn_bkp
                       "Ispexec Setmsg Msg(Edtm00ØO)"
                    End
               Else "Ispexec Setmsg Msg(Edtm00ØN)"
RETURN /* End-of-the-procedure-POPUP2 */
```

EDITMAC

```
/*———————————————————————————————————————————————————————————*/
/* Control if the member is still there.                             */
/*———————————————————————————————————————————————————————————*/

Rs = Sysdsn(Dsn"("Mem")")
If Rs="OK" Then Nop
           Else Do
                  Say 'The specified member is not found in dataset.'
                  Return
                End

"Ispexec Edit Dataset('"Dsn"("Mem")') Macro(ChgMac)"

Return Rc /* End-of-the-procedure-EDITMAC */
```

REXX : BATCHREX

```
/* REXX */
/*———————————————————————————————————————————————————————————*/
/* REXX program       : Batchrex                                     */
/* Edit macro called  : Batchmac                                     */
/* Purpose            : Change multiple strings in 3 libraries. */
```

```
/*-------------------------------------------------------------------*/
"Prof nopref"
"Altlib  Activate Application(Exec) Da(Exp.Ctm.Rexx)"
Mpv.Ø = 1
Mpv.1 = Exp.Ctm.Test1
Mpv.2 = Exp.Ctm.Test2
Mpv.3 = Exp.Ctm.Test3

Do i = 1 to 3
 Dsn = Mpv.i
 x = Outtrap('Var.')
 "Listds" Dsn "Members"
 x = Outtrap('Off')       /* Turns trapping OFF */

/*-------------------------------------------------------------------*/
/* Get members of SOURCE dset and execute the edit macro in a loop.  */
/*-------------------------------------------------------------------*/
Do j = 7  To Var.Ø
  Mem = Strip(Var.j)
  "Ispexec Edit Dataset('"Dsn"("Mem")') Macro(Batchmac)"
  End
End

"Altlib Deactivate Application(Exec)" /* Deallocate the REXX library */
Exit
```

REXX PROCEDURE : CHGIEBCP

```
/* REXX */
/*-------------------------------------------------------------------*/
/* Allocate files needed by Iebcopy.                                 */
/*-------------------------------------------------------------------*/
Arg Dsn1 Dsn2

"Alloc Fi(Input)    Da("Dsn1") Shr Reuse"
"Alloc Fi(Output)   Da("Dsn2") Shr Reuse"
"Alloc Fi(Sysout)   Dummy Reuse"
"Alloc Fi(Sysprint) Dummy Reuse"
"Alloc Fi(Sysut1)   Unit(VIO) Space(1,1) Cyl New Delete Reuse"
"Alloc Fi(Sysut2)   Unit(VIO) Space(1,1) Cyl New Delete Reuse"
"Alloc Fi(Sysut3)   Unit(VIO) Space(1,1) Cyl New Delete Reuse"
"Alloc Fi(Sysut4)   Unit(VIO) Space(1,1) Cyl New Delete Reuse"

/*-------------------------------------------------------------------*/
/* Build the SYSIN control statements of the Iebcopy utility.        */
/*-------------------------------------------------------------------*/
"Alloc Fi(Sysin)    Unit(VIO) Space(1,Ø) Blksize(8Ø) Lrecl(8Ø),
 Recfm(F B) Dsorg(PS) New Delete Reuse"
```

```
"Ispexec Lminit Dataid(Martapv) Ddname(Sysin) Enq(Exclu)"
Rc5 = Rc
If Rc5 ¬= Ø Then
             Do
                "Ispexec Vput Rc5 Profile"
                "Ispexec Setmsg Msg(EdtmØØ9E)"
               Exit
             End
           Else
             Do
               Card = ' COPY OUTDD=OUTPUT,INDD=INPUT'
               "Ispexec Lmopen  Dataid("Martapv") Option(Output)"
               "Ispexec Lmput   Dataid("Martapv"),
               Dataloc(Card) Datalen(8Ø) Mode(Invar)"
               "Ispexec Lmclose Dataid("Martapv")"
               "Ispexec Lmfree  Dataid("Martapv")"
               Rc4 = Rc
             End

If Rc4 ¬= Ø Then Do
                  "Ispexec Vput Rc4 Profile"
                  "Ispexec Setmsg Msg(EdtmØØ9F)"
                  Exit
                End
           Else Do

                  "Ispexec Select Pgm(IEBCOPY)"
                   Rc6 = Rc
                  If Rc6 ¬= Ø Then Do
                                    "Ispexec Vput Rc6 Profile"
                                    "Ispexec Setmsg Msg(EdtmØØ9H)"
                                    Exit
                                   End

/*———————————————————————————————————————————————————————————————*/
/* Free all DDnames of Iebcopy.                                      */
/*———————————————————————————————————————————————————————————————*/
"Free File(Input,Output,Sysin,Sysprint,Sysut1,Sysut2,Sysut3,Sysut4)"
Exit                     /* End-of_procedure */
```

EDIT MACRO : CHGMAC

```
"Isredit Macro"
/*———————————————————————————————————————————————————————————————*/
/* Edit macro    : ChgMac                                            */
/* Called from   : ChgRexx                                           */
/* Purpose       : Change strings in a member of a given dataset.    */
/*———————————————————————————————————————————————————————————————*/
Status = Msg('Off')
```

```
"Ispexec Control Errors Return"    /* Let EXEC process errors       */
Chg = Ø; Err = Ø                   /* Change-count, Error-count = Ø */

/*———————————————————————————————————————————————————————————————*/
/* Get the Number Mode.                                              */
/* Edit macro has to take into account that if Number_Mode is 'ON'  */
/* then the number of editable characters is Lrecl-8. Because right-*/
/* end side will include "sequence numbers". (NUMBER FIELDS)        */
/* So if the user enters a Column2 value that is bigger than        */
/* Lrecl-8, It is modified by this edit macro so that we will       */
/* prevent errors.                                                   */
/*———————————————————————————————————————————————————————————————*/
"Isredit (Mode) = Number"

/*———————————————————————————————————————————————————————————————*/
/* Get the variables from the Profile Variable Pool. These are the  */
/* strings which will be used on the 'Ispf Change' edit command.    */
/*———————————————————————————————————————————————————————————————*/

"Ispexec Vget (From,To,Mem,Qual,Col1,Col2,L2) Profile"

If ((Mode = ON) & (Col2>=(L2-8)))  Then  Col2 = L2-8

/*———————————————————————————————————————————————————————————————*/
/* Build the limiting keyword.                                       */
/*———————————————————————————————————————————————————————————————*/
Select
   When Qual = 1  Then Qual = CHARS
   When Qual = 2  Then Qual = PREFIX
   When Qual = 3  Then Qual = SUFFIX
   When Qual = 4  Then Qual = WORD
   Otherwise      Nop
End

If Length(From) = 1 Then
          "Isredit Change" "'"From"'" To "ALL" Qual Col1 Col2
                    Else
          "Isredit Change" From To "ALL" Qual Col1 Col2
          Rc2 = Rc

"Isredit (Chg,Err) = CHANGE_COUNTS"
Chg = Abs(Chg)
Err = Abs(Err)

If Rc2 = 4 Then Msg2='The string is not found.  '
If Rc2 = 8 Then Msg2='Str2 is longer than Str1. '
If Rc2 =12 Then Msg2='Inconsistent parameters.  '
If Rc2 =2Ø Then Msg2='Severe error.             '
If Rc2 = Ø Then Do
                Msg2='Change macro command Rc=Ø.'
                /*———————————————————————————————————————————*/
```

```
                /* If Error-Count is Ø, then we can save the   */
                /* current member in the SOURCE dataset.       */
                /*-------------------------------------------*/
                If Err= Ø Then Do
                               "Isredit Save"
                               Rc3 = Rc   /* Rc3 = Save Return Code */
                               End
                End

"Ispexec Vput (Chg,Err,Msg2,Rc2,Rc3) Profile"
/*-------------------------------------------------------------------*/
/* No matter it was saved or not, exit from the member to be able to */
/* process the next member in the SOURCE dataset.                    */
/*-------------------------------------------------------------------*/
"Isredit Cancel"
Return
```

EDIT MACRO: BATCHMAC

```
"Isredit Macro"
"Isredit Change V312 V41Ø ALL"
"Isredit Change '.LIB312' '.LIB41Ø' ALL"
"Isredit Save"
«Isredit Cancel»
```

ISPF PANEL 0

```
)ATTR
    < TYPE(INPUT) INTENS(HIGH) COLOR(YELLOW) CAPS(OFF)
    { TYPE(INPUT) INTENS(HIGH) COLOR(GREEN)  CAPS(OFF)
    } TYPE(TEXT)  INTENS(HIGH) COLOR(PINK)
    $ TYPE(TEXT)  INTENS(HIGH) COLOR(PINK)
    % TYPE(TEXT)  INTENS(HIGH) COLOR(TURQ)
    + TYPE(TEXT)  INTENS(LOW)
    _ TYPE(INPUT) INTENS(HIGH)
    # TYPE(INPUT) INTENS(HIGH) COLOR(WHITE)
    [ TYPE(TEXT)  INTENS(HIGH) COLOR(RED) HILITE(REVERSE)
)BODY
%--------------[ AUTOMATED LIBRARY UPDATE %--------------
%
+
% Please enter the dataset name on which you will make changes:
+ Dataset      %===>_Dsn                                          %
                                                                #Mes
% Please enter a "FROM" and a "TO" string: (Without quotation marks)
+ From string %===>{Z

                                     %
+
+ To   string %===><Z
```

```
                                        %
+
%Please qualify the search string :(Default is "CHARS")
%(<Z%) $1-CHARS  2-PREFIX  3-SUFFIX  4-WORD %
+
% Please enter the Column Limitations : (Optional)
+ Column-1    %===><Z           + Column-2    %===><Z
}
}Hit%ENTER}to proceed. Hit%PF3}o%PFØ4}key to exit from the application.
)INIT
.CURSOR = DSN
.ZVARS = '( From To Qual Col1 Col2)'
&Qual = '1'

)REINIT
 Refresh(Mes)

)PROC
&MES = ''
&SPFKEY = .PFKEY
  Ver (&Dsn,Nonblank,Msg=EDTMØØØ)
  Ver (&Dsn,Dsname,Msg=EDTMØØ3)

  &A = Trunc(&Dsn,1)
  Ver (&A,Pict,'A')

  Ver (&From,Nonblank,Msm=EDTMØØ1)
  Ver (&To,Nonblank,Msg=EDTMØØ2)
  Ver (&Qual,Nonblank,Msg=EDTMØØ4)
  Ver (&Qual,List,1,2,3,4,Msg=EDTMØØ5)
  Ver (&Col1,Num,Msg=EDTMØØ7)
  Ver (&Col2,Num,Msg=EDTMØØ7)

IF (&Col1 > &Col2 & &Col2 NE '')
   .Msg = EDTMØØ9K

Vput(Spfkey From To Dsn Qual Col1 Col2 ) Profile
)END
```

ISPF PANEL1

```
)ATTR
 % TYPE(TEXT)  COLOR(WHITE) CAPS(OFF) HILITE(USCORE) INTENS(LOW)
 @ TYPE(TEXT)  COLOR(RED)   CAPS(OFF) HILITE(USCORE)
 + TYPE(TEXT)  COLOR(TURQ)  CAPS(OFF) JUST(LEFT)
 $ TYPE(TEXT)  COLOR(GREEN) HILITE(REVERSE)
)BODY WINDOW(65,9)
+
```

```
$                   ****  ATTENTION!  ****                      +
+                                                               +
+ You are about to  browse  the  listing of the comparison of  +
+ SOURCE and BKPCPY datasets. Please check the changes that    +
+ have been made on the SOURCE dataset. After browsing, you    +
+ will be presented with another panel which asks if you agree +
+ with the changes.                                            +
%                    Hit@<Enter>%to continue.                  +
)INIT
)PROC
&SPFKEY = .PFKEY
   Vput Spfkey Profile
)END
```

ISPF PANEL2

```
)ATTR
 % TYPE(TEXT)  COLOR(WHITE) CAPS(OFF) HILITE(USCORE) INTENS(LOW)
 @ TYPE(TEXT)  COLOR(RED)   CAPS(OFF) HILITE(USCORE)
 + TYPE(TEXT)  COLOR(TURQ)  CAPS(OFF) JUST(LEFT)
 $ TYPE(TEXT)  COLOR(GREEN) HILITE(REVERSE)
 [ TYPE(INPUT) COLOR(PINK)  CAPS(ON)  HILITE(REVERSE)
)BODY WINDOW(32,9)
+
$       DO YOU ACCEPT THE
$        CHANGES ON THE
$        SOURCE DATASET?
+
+ Yes(YS) / No(N)......:[Z+
+
% Hit@<Enter>%to proceed.+
)INIT
.Zvars = '( Resp )'
&Resp = 'N'
.Cursor = Resp
)PROC
&Spfkey = .Pfkey
Vput Spfkey Profile
Ver  (&Resp,Nonblank,Msg=EDTMØØ9L)
Ver  (&Resp,List,Y,N,Msg=EDTMØØ9M)
Vput (Resp) Profile
)END
```

ISPF PANEL3

```
)ATTR DEFAULT(%+_)
     @ TYPE(OUTPUT) INTENS(HIGH) JUST(ASIS) COLOR(GREEN)
     % TYPE(TEXT)   INTENS(HIGH) COLOR(TURQ)
     $ TYPE(OUTPUT) INTENS(HIGH) JUST(ASIS) COLOR(YELLOW) CAPS(OFF)
     * TYPE(TEXT)   INTENS(HIGH) JUST(ASIS) COLOR(RED)
```

```
     # TYPE(TEXT)   INTENS(HIGH) JUST(ASIS) COLOR(WHITE)
     [ TYPE(TEXT)   INTENS(HIGH) COLOR(GREEN) HILITE(REVERSE)
)BODY
%---------------[ AUTOMATED LIBRARY UPDATE %-------------
%                              [ EXECUTION RESULTS %
+
%Command ===>_ZCMD
+
* DATASET     :$DSN                                          *DATE
:+&ZDATE
* MEMBER-COUNT :$CNT*                                        *USER-ID
:+&ZUSER
* FROM         :$FROM
* TO           :$TO
* COL1 & COL2  :$COL1  *$COL2      +
+
* NOTE:#R2 = Isredit Change command Rc.  R1 = Ispexec Edit command Rc.
+              Hit#<F1>+to get more info on Return Codes.
+
+%MEMBER  %CHANGE%ERROR%MESSAGE1         %RC%RC%MESSAGE2
+%NAME    %COUNT %COUNT%                 %1 %2 %
+%------%---%-%--------- %-%-%---------------
+
)MODEL
 @MEM     @CHG   @ERR  @MSG1            @Z @Z @MSG2
)INIT
 &Zcmd =''
.Help = Panelh
.Zvars = '(Rc1 Rc2)'
)PROC
)END
```

ISPF PANEL : PANELH

```
)ATTR DEFAULT(%+_)
     % TYPE(TEXT)   INTENS(HIGH) COLOR(GREEN)
     < TYPE(TEXT)   INTENS(HIGH) COLOR(PINK)  HILITE(REVERSE)
     * TYPE(TEXT)   INTENS(HIGH) JUST(ASIS)   COLOR(RED)
     [ TYPE(TEXT)   INTENS(HIGH) COLOR(GREEN) HILITE(REVERSE)
)BODY
%---------------[ AUTOMATED LIBRARY UPDATE %-------------
%
*  Rc1 Message1   ("Ispexec Edit Dataset" command Return Codes)
*  === ========================================================
+  Ø   Member updated.
+  4   Member not saved. / No save. Abend S37'
+  14  Member in use.
+  2Ø  Severe error.
+
<  NOTE :+In case you see "S37" on the Message1 text, it's necessary
+       +to compress or enlarge the SOURCE dataset.
```

```
+
+
*  Rc2   Message2 ("Isredit Change" command Return Codes)
*  ===== ================================================
+  Ø    Change macro command normal completion.
+  4    The string is not found.
+  8    String-2 is longer than String-1 and
+       substitution was not performed on at least one change.
+ 12    Inconsistent parameters. The string to be found does
+       not fit between the specified columns.
+ 2Ø    Severe error.
+
)INIT
)PROC
)END
```

ISPF MESSAGES : EDTM00

```
* EDTMØØ MESSAGE DEFINITIONS
*
EDTMØØØ  'Enter a dataset name. '      .HELP=*          .ALARM=YES
'Enter a dataset in which you''d like to run an edit macro for its each
member'
*
EDTMØØ1  'Enter a FROM string.   '     .HELP=*          .ALARM=YES
'Enter a FROM string that will be used for the ''ISPF Change '' Edit command.'
*
EDTMØØ2  'Enter a TO string.     '     .HELP=*          .ALARM=YES
'Enter a TO string that will be used for the ''ISPF Change '' Edit command.'
*
EDTMØØ3  'Invalid Dset-qualifier.'     .HELP=*          .ALARM=YES
'Each qualifier must be 1-8 alphanumeric chars  & first one must be
alphabetic.'
*
EDTMØØ4  'Enter 1, 2, 3, or 4.'        .HELP=*          .ALARM=YES
'Enter a number between 1 and 4 which corresponds to the qualifying keyword.'
*
EDTMØØ5  'Out of range.'               .HELP=*          .ALARM=YES
'You should enter a value between 1 and 4 for this field.'
*
EDTMØØ6 'No changes are made.    '     .HELP=*          .ALARM=YES
'No changes are made in the specified SOURCE dataset.'
*
EDTMØØ7 'Enter a numeric value.  '     .HELP=*          .ALARM=YES
'Column limitation that will be used for ''CHANGE'' command, must be numeric.'
*
EDTMØØ8 '«FROM» string > Lrecl.  '     .HELP=*          .ALARM=YES
'«FROM» string (&L1) can not be longer than the dataset's Record Length (&L2)'
*
EDTMØØ9 '«TO» string > Lrecl.'         .HELP=*          .ALARM=YES
'«TO» string (&L1) can not be longer than the dataset's Record Length (&L2)'
*
```

```
EDTM009A 'Hit <PF03+Enter> '            .HELP=*          .ALARM=YES
'Hit <Enter> then <PF03> to exit from the Comparison output.'
*
EDTM009B 'PO dset has no member.'       .HELP=*          .ALARM=YES
'The PO dset you entered has no member. Please enter it again.'
*
EDTM009C 'Dset doesn''t exist.'         .HELP=*          .ALARM=YES
'The PO dset you entered does not exist. Please enter it again.'
*
EDTM009D 'Dataset is not PO.'          .HELP=*           .ALARM=YES
'The PO dset you entered is not a PO. Please enter it again.'
*
EDTM009E 'Lminit Rc = &Rc5    '         .HELP=*          .ALARM=YES
'1st Lminit is not successful.'
*
EDTM009F 'Lmfree error.'                .HELP=*          .ALARM=YES
'Lmfree error. Rc = &Rc4'
*
EDTM009G 'Lminit Rc = &Rc7    '         .HELP=*          .ALARM=YES
'2nd Lminit is not successful.'
*
EDTM009H 'Iebcopy error.'               .HELP=*          .ALARM=YES
'Iebcopy Return Code =  &Rc6'
*
EDTM009I 'No change was made.'          .HELP=*          .ALARM=YES
'The dataset was not updated, since there was no string change.'
*
EDTM009J 'Col value is incorrect.'      .HELP=*          .ALARM=YES
'Col1 or Col2 can't be bigger than the Lrecl of the SOURCE dataset.'
*
EDTM009K 'Col2 must be bigger.'         .HELP=*          .ALARM=YES
'Col2 value must be bigger than Col1 value. Please enter it again.'
*
EDTM009L 'Enter Y or N.       '         .HELP=*          .ALARM=YES
'This field must be either Y or N.'
*
EDTM009M 'Enter Y to accept.  '         .HELP=*          .ALARM=YES
'Enter ''Y'' to accept changes on SOURCE dset and delete BKPCPY dset.'
*
EDTM009N 'BKPCPY dset kept.'            .HELP=*          .ALARM=YES
'BKPCPY dataset is kept. SOURCE dataset has now changes.'
*
EDTM009O 'BKPCPY dset deleted.'         .HELP=*          .ALARM=YES
'BKPCPY dataset is deleted. SOURCE dataset has now changes.'
*
```

Atalay Gul
Systems Programmer (Turkey)

Automatic starting and stopping of the system

INTRODUCTION

The number of components that make up OS/390 is rapidly increasing. In our company we have a system based on OS/390 Version 2 Release 5. This has 34 started tasks and 25 system address spaces and we plan to add more. Proper starting and stopping of such a complex system manually is almost an impossible task. Because we do not use any commercial software with automatic operator functionality, we wrote an Assembler program for executing commands from batch.

ACTIVATION

To specify a statement to start the system executing a command from the console or put it in a COMMND*xx* member such as:

```
COMM='S OPER,M=$STRTSYS'
```

To stop the system from any console we have to specify:

```
S OPER,M=$STOPSYS
```

SOURCE

```
*-------------------------------------------------------------------
* PROGRAM FOR EXECUTING OPERATOR COMMANDS FROM BATCH PLUS COMMANDS
* PAUSE OR WAIT
*-------------------------------------------------------------------
OPERBTCH CSECT
         SAVE  (14,12)
         BALR  12,Ø
         USING  *,12
         LA    2,SAVEREG
         ST    13,SAVEREG+4
         ST    2,8(13)
         LA    13,SAVEREG
*-------------------------------------------------------------------
         OPEN  (CONSSTMT,(INPUT))
         LA    2,CONSSTMT
         TM    DCBOFLGS--IHADCB(2),DCBOFOPN  .OPEN OK ?
         BZ    END
*
         OPEN  (SYSPRINT,(OUTPUT))
         LA    2,SYSPRINT
```

```
         TM    DCBOFLGS--IHADCB(2),DCBOFOPN  .OPEN OK ?
         BZ    END
*---------------- ACTIVATE CONSOLE --------------------------------
         MCSOPER REQUEST=ACTIVATE,NAME=CONSNAME,CONSID=CONSID,         X
               TERMNAME=CONSTERM,MSGECB=CONSECB,                       X
               MCSCSA=CONSCSA,MCSCSAA=CONSCSAA,                        X
               RTNCODE=CONSRC,RSNCODE=CONSRC
         CLC   CONSRC,=F'Ø'
         BNE   ERROR_MSG_ACTIVATE
GET_NEW  GET   CONSSTMT
         LR    11,1
         MVC   STMTTEXT(72),Ø(11)
         LA     1Ø,79               .LENGTH INPUT STMT
*---------------  WRITE INPUT STMT --------------------------------
         PUT   SYSPRINT
         MVI   Ø(1),C' '
         MVC   1(12Ø,1),Ø(1)
         MVC   STMTLEN(2),=C'> '
         MVC   1(8Ø,1),STMT
*-------- SKIP COMMENT ------------------------------------------
         CLI   Ø(11),C'*'
         BE    GET_NEW
*------  SKIP LEADING BLANKS -------------------------------------
IFBLANK  CLI    Ø(11),C' '
         BNE    CHECK_PAUSE
         LA     11,1(11)
         BCT    1Ø,IFBLANK
         B      GET_NEW            .ALL RECORD IS BLANK AND GET NEW
*------  CHECK PAUSE  ------------------------------------------
CHECK_PAUSE EQU *
         LA     Ø,4
         LA     1,TXTPAUSE
         LR     2,11
         BAL    14,CHECK_TXT
         LTR    15,15
         BNZ    CHECK_WAIT
         LA     11,5(11)
         SH     1Ø,=H'5'
         B      EXEC_PAUSE_WAIT
*------  CHECK WAIT --------------------------------------------
CHECK_WAIT EQU *
         LA     Ø,3
         LA     1,TXTWAIT
         LR     2,11
         BAL    14,CHECK_TXT
         LTR    15,15
         BNZ    EXEC_STMT
         LA     11,4(11)
         SH     1Ø,=H'4'
*-------- IF STMT IS PAUSE OR WAIT GO TO SUBROUTINE EXECUTE IT ----
EXEC_PAUSE_WAIT EQU *
         SR    15,15
```

```
         LR    Ø,1Ø
         LR    1,11
         BAL   14,WAIT_SAME_TIME
         LTR   15,15
         BZ    GET_NEW
*-------- IF NOT COMMENT AND NOT WAIT ----------------------------
EXEC_STMT EQU *
         MVC   STMTLEN(2),=H'8Ø'
         LA    2,STMTLEN           .R2=LENGTH STMT
         SR    15,15
         MODESET KEY=ZERO,MODE=SUP
         MGCRE   MF=(E,LAREA),TEXT=(2),CONSID=CONSID
         MODESET KEY=NZERO,MODE=PROB
         B     GET_NEW
*-------------- DEACTIVATE CONSOLE -------------------------------
ENDCONS  EQU *
         MCSOPER REQUEST=DEACTIVATE,CONSID=CONSID,                    X
               RTNCODE=CONSRC,RSNCODE=CONSRC
         CLC   CONSRC,=F'Ø'
         BNE   ERROR_MSG_DEACTIVATE
         B     END
*--------------  PREPARE ERROR MESSAGE  --------------------------
ERROR_MSG_ACTIVATE EQU *
         MVC   MSGREQ(11),TXT_ACT
         B     PUT_ERR
ERROR_MSG_DEACTIVATE EQU *
         MVC   MSGREQ(11),TXT_DEA
*--------------*  CONVERT RETURN CODE TO CHAR  --------------------
PUT_ERR  EQU *
         L     Ø,CONSRC
         LA    1,ERRRC
         BAL   14,CONV_BIN_DEC
*--------------  CONVERT REASON CODE TO CHAR  --------------------
         L     Ø,CONSREAS
         LA    1,ERRREAS
         BAL   14,CONV_BIN_DEC
*--------------  WRITE ERROR MSG ---------------------------------
         PUT   SYSPRINT,MSGØ1
         MVI   Ø(1),C' '
         MVC   1(12Ø,1),Ø(1)
         MVC   1(MSGØ1E--MSGØ1,1),MSGØ1
*--------------  END AND EPILOG  ---------------------------------
END      EQU   *
         CLOSE  (CONSSTMT,,SYSPRINT)
         L      13,SAVEREG+4
         RETURN (14,12),,RC=(15)
*********************************************************************
**    SUBROUTINE CHECK TEXT
**    PARAMETERS:
**        @RØ  -- LENGTH OF CHARACTERS TO CHECK
**        @R1  -- INPUT TEXT1
**        @R2  -- INPUT TEXT2
```

```
*************************************************************************
CHECK_TXT EQU *
         STM    14,2,SAVE_C_T
CHECK_NEXT_TXT EQU *
         XR     14,14
         CLI    Ø(2),C'A'            .COMPARE INPUT CHAR WITH A
         BNL    COMPARE_CHAR
         IC     14,Ø(2)              .CONVERT LOWERCASE TO UPERCASE
         AH     2,=H'64'             .CHARACTER
         STC    14,Ø(2)
COMPARE_CHAR  EQU *
         CLC    Ø(1,1),Ø(2)
         BE     NEXT_CHAR
         LA     15,4
         B      RET_C_T
NEXT_CHAR EQU *
         LA     1,1(1)
         LA     2,1(2)
         BCT    Ø,CHECK_NEXT_TXT
         XR     15,15
RET_C_T  EQU *
         L      14,SAVE_C_T
         LM     Ø,2,SAVE_C_T+8
         B      Ø(14)
*------  LOCAL DECLARATIONS ----------------------------------------
CLI_TXT  CLI    Ø(1),C' '
SAVE_C_T DS     5F
*************************************************************************
**    SUBROUTINE EXECUTE PAUSE OR WAIT STMT
**    PARAMETERS:
**        @RØ  -- LENGTH OF INPUT STMT
**        @R1  -- INPUT STMT WITH NUMBER OF SECONDS
*************************************************************************
WAIT_SAME_TIME EQU *
         STM    14,3,SAVE_WAIT
         LR     3,Ø
         LR     2,1
*------  SKIP NEXT BLANK -------------------------------------------
IFNEXTBL CLI    Ø(2),C' '          .SKIP BLANKS TO NUMBER OF SECONDS
         BNE    CHECK_NUMERIC_PARM .IF NOT BLANK GO TO CHECK NUMBER
         LA     2,1(2)             .GO TO THE NEXT CHAR
         BCT    3,IFNEXTBL         .COUNT NEXT CHAR AND GO TO CHECK IT
         B      NUMERIC_NOT_OK     .IF ALL RECORD IS BLANK
*------  CHECK NUMERICS (NUMBER OF SECONDS)-----------------------
CHECK_NUMERIC_PARM EQU *
         LR     1,2
IFNUM    CLI    Ø(2),C' '          .IF END NUMBER PARAMETER
         BE     END_NUMBER
         CLI    Ø(2),C'Ø'
         BL     NUMERIC_NOT_OK
         CLI    Ø(2),C'9'
         BH     NUMERIC_NOT_OK
```

```
         LA     2,1(2)
         BCT    3,IFNUM
         B      NUMERIC_NOT_OK
*--------------  CHECK IS NUMBER TO HIGH -------------------------
END_NUMBER EQU *
         LR     3,2                .COMPUTE NUMBER LEN R3=R2--R1
         SR     3,1                .R3 IS NUMBER LEN
         LA     1,6                .R1 = 6
         CR     1,3                .IF NUMBER LEN > 6
         BL     NUMERIC_TO_HIGH    .THEN NUMBER IS TO HIGH
*--------------  STORE NUMBER IN CHARSEC FIELD --------------------
         SR     2,3                .RETURN PTR TO BEGIN OF NUMERIC
         LA     1,CHARSEC+6
         SR     1,3
         BCTR   3,Ø                .R3=R3--1
         EX     3,MVCSEC
*--------------  WRITE PAUSE MSG ON CONSOLE -----------------------
         LA    2,MSGWØ1L
         XR    Ø,Ø
         WTO   TEXT=(2),ROUTCDE=11
*--------------  WRITE PAUSE MSG IN SYSPRINT ----------------------
         PUT   SYSPRINT
         MVI   Ø(1),C' '
         MVC   1(12Ø,1),Ø(1)
         MVC   1(MSGWØ1E--MSGWØ1,1),MSGWØ1
*--------------  WAIT ---------------------------------------------
         NI     CHARSEC+5,X'CF'
         PACK   PACKSEC(8),CHARSEC(6)
         CVB    2,PACKSEC
         MH     2,=H'1ØØ'
         ST     2,BINSEC
         STIMER WAIT,BINTVL=BINSEC
         B      RET_WAIT_OK
NUMERIC_TO_HIGH EQU *
         LA     15,8
         B      RET_WAIT
NUMERIC_NOT_OK  EQU *
         LA     15,4
         B      RET_WAIT
RET_WAIT_OK EQU    *
         XR     15,15
RET_WAIT EQU    *
         L      14,SAVE_WAIT
         LM     Ø,3,SAVE_WAIT+8
         B      Ø(14)
*------  LOCAL DECLARATIONS -------------------------------------
MVCSEC    MVC    Ø(Ø,1),Ø(2)
SAVE_WAIT DS     6F
TXTPAUSE  DC     C'PAUSE'
TXTWAIT   DC     C'WAIT'
PACKSEC   DS     D
BINSEC    DS     F
```

```
MSGW01L   DC     AL2(MSGW01E--*--1)
MSGW01    DC     C'>>> WAITING '
CHARSEC   DC     CL6'      0'
          DC     C' SECONDS <<<'
MSGW01E   EQU *
          DS     0H
*********************************************************************
**    SUBROUTINE CONVERT BIN NUMBER TO CHARACTER
*********************************************************************
CONV_BIN_DEC EQU *
         STM    14,1,SAVE_CONV
         CVD    0,POM
         UNPK   0(3,1),POM
         OI     2(1),X'F0'
*------  TRANSFORM LEADING 0 INTO BLANK --------------------------
         LA     0,2
ZERO     CLI    0(1),X'F0'
         BNE    RET_CONV
BLANKO   MVI    0(1),X'40'
         LA     1,1(1)
         BCT    0,ZERO
RET_CONV EQU    *
         LM     14,1,SAVE_CONV
         B      0(14)
*------  LOCAL DECLARATIONS --------------------------------------
SAVE_CONV DS     4F
POM       DS     D
*********************************************************************
**    DECLARATIONS
*********************************************************************
SAVEREG  DS  18F
CONSSTMT DCB DDNAME=SYSIN,DSORG=PS,MACRF=GL,EODAD=ENDCONS
SYSPRINT DCB DDNAME=SYSPRINT,DSORG=PS,MACRF=PL,LRECL=121,BLKSIZE=0
*
STMT     DC  C'--'
STMTLEN  DS  H
STMTTEXT DC  CL80' '
STMTE    EQU *
*
CONSNAME DC  CL8'BATCHCON'
CONSID   DS  CL4
CONSTERM DS  CL8'BATCH'
CONSECB  DS  A
         DS  0D
CONSCSA  DS  CL4
CONSCSAA DS  F
CONSRC   DS  F
CONSREAS DS  F
LAREA    MGCRE MF=L
*
TXT_ACT  DC  CL11'ACTIVATE   '
TXT_DEA  DC  CL11'DEACTIVATE '
```

```
*
MSGØ1    DC  C'>>> ERROR '
MSGREQ   DC  CL11' '
         DC  C' CONSOLE RC='
ERRRC    DS  CL3
         DC  C' REASON='
ERRREAS  DS  CL3
MSGØ1E   EQU *
         IHAPSA
         DCBD   DSORG=PS,DEVD=DA
         END
/*
```

JOB FOR COMPILATION AND LINK

```
//S1  EXEC ASMACL,PARM.C='OBJ',PARM.L='AC=1'
//C.SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR
//         DD DSN=SYS1.MODGEN,DISP=SHR
//C.SYSIN  DD DSN=USERID.USER.ASM(OPERBTCH),DISP=SHR
//L.SYSLMOD DD DSN=SYS1.LINKLIB(OPERBTCH),DISP=SHR
```

Note: the program must be linked with authorization code 1 and placed in the APF library.

PROCEDURE OPER

```
//OPER     PROC M=,D='SYS1.OPERLIB',PREFIX=NONE   <== CHANGE
//OPER     EXEC PGM=OPERBTCH,REGION=ØK,TIME=144Ø
//STEPLIB  DD  DSN=SYS1.LINKLIB,DISP=SHR
//SYSPRINT DD  SYSOUT=X,HOLD=YES
//PARMLIB  DD  DSN=&D(&M),DISP=SHR
```

LIST FOR STARTING THE SYSTEM

This should be placed in SYS1.OPERLIB($STRTSYS)

```
*-------------------------------------------------------------------
* START VTAM
*-------------------------------------------------------------------
S VTAM
Pause 12Ø
*-------------------------------------------------------------------
* Automatic starting of tasks
*-------------------------------------------------------------------
S TSO                                    /* Time Sharing Option */
*-------------------------------------------------------------------
* Start DB2
*-------------------------------------------------------------------
Start DB2
Pause 1Ø
```

```
#Start DB2
Pause 1Ø
*-------------------------------------------------------------------
* Start the other tasks
*-------------------------------------------------------------------
S IXFP
S DFSMSHSM
S DFRMM
S ZARA13
Pause 2Ø
*-------------------------------------------------------------------
* Start CICS
*-------------------------------------------------------------------
S CICSPROD                                /* CICS for production */
Pause 1Ø
S CICSTEST                                       /* CICS for test*/
Pause 1Ø
*-------------------------------------------------------------------
* Start OMVS
*-------------------------------------------------------------------
S TCPIP
Pause 1Ø
S NAMED
Pause 5
S FTPD
S IMWEBSRV
*-------------------------------------------------------------------
* Start of es328x
*-------------------------------------------------------------------
Pause 3
$S LOGON1
S JSX1
*-------------------------------------------------------------------
* Start of additional action
*-------------------------------------------------------------------
S JOB,N=AUTOCMD
*-------------------------------------------------------------------
* Connection  CICS with DB2
*-------------------------------------------------------------------
Pause 2Ø
F CICSPROD,DSNC STRT
F CICSTEST,DSNC STRT
*-------------------------------------------------------------------
D T                                       /* Display Ending time */
*-------------------------------------------------------------------
```

LIST FOR STOPPING THE SYSTEM PLACED IN SYS1.OPERLIB($STOPSYS)

```
*-------------------------------------------------------------------
* Message to TSO users to perform log-off
*-------------------------------------------------------------------
```

```
F Tso,usermax=Ø    /* Don't allow anyone else to logon right now */
Send 'Please LOGOFF - system will shut down in two minutes!',all,now
Pause 6Ø
*-------------------------------------------------------------------
* Setting of DFSMSHSM in the emergency mode
*-------------------------------------------------------------------
F DFSMSHSM,SETSYS EMERGENCY
*-------------------------------------------------------------------
* Stopping local printers
*-------------------------------------------------------------------
F JES328X,P,ALL,I
*-------------------------------------------------------------------
* Disconnecting  CICS from DB2
*-------------------------------------------------------------------
F CICSTEST,DSNC STOP FORCE
F CICSPROD,DSNC STOP FORCE
Pause 1Ø
*-------------------------------------------------------------------
* Stopping the rest of the tasks
*-------------------------------------------------------------------
P IXFP
F DFSMSHSM,STOP
P ZARA13
P DFRMM
P RMF
P JSX1
*-------------------------------------------------------------------
* Stop Db2
*-------------------------------------------------------------------
Pause 6Ø
--Stop DB2 mode(force)
#Stop DB2 mode(force)
*-------------------------------------------------------------------
* Stopping CICS
*-------------------------------------------------------------------
F CICSTEST,CEMT P,SHUT,I
Pause 5
F CICSPROD,CEMT P,SHUT,I
*-------------------------------------------------------------------
*-------------------------------------------------------------------
Pause 6Ø
P TSO
P LLA
Pause 2Ø
*-------------------------------------------------------------------
* Stop VTAM
*-------------------------------------------------------------------
Z NET,QUICK
D T                                    /* Display Ending time */
*-------------------------------------------------------------------
```

REXX over IP – Part 2

This month we complete our look at a REXX utility that uses TCP/IP to provide cross-system communication.

RVOLDATA

The following code does not require any APF authorization:

```
//XXXXASM  JOB XX,YYY,CLASS=X,MSGCLASS=T,MSGLEVEL=1 - Your job card
//STEPA EXEC ASMFCL,PARM.LKED='NORENT,NOREUS'
//ASM.SYSLIB DD DSN=SYS1.MACLIB,DISP=SHR
//           DD DSN=SYS1.MODGEN,DISP=SHR
//ASM.SYSIN DD *
         MACRO
         REXREGS
         LCLA  &CNT
&CNT     SETA  Ø
.LOOP    ANOP
R&CNT    EQU   &CNT
&CNT     SETA  &CNT+1
         AIF   (&CNT LT 16).LOOP
         MEND
         MACRO
  SHOWARAY &LABEL,&ASNAME,&ERR=ABEND ØØ1,&LEN=,&SUBARRAY=,&DEBIN=,&LINK=
         PRINT NOGEN
*********************************************************************
*
* MACRO TO CREATE REXX ARRAY VARIABLES
*
* NOTE RESTRICTION: THIS MACRO IS LIMITED TO CREATING UP TO 9,999,999
*                   ENTRIES FOR EACH ARRAY.
*
* MACRO  FORMAT:
*        SHOWARAY &LABEL,&ASNAME,&ERR=,&LEN=,&SUBARRAY=,&DEBIN=
* WHERE:
*        &LABEL  IS THE NAME OF THE LABEL WHICH ADDRESSES THE FIELD
*                FROM WHERE THE DATA TO BE DEFINED IN A REXX VARIABLE
*                IS LOCATED
*        &ASNAME IS THE NAME TO BE ASSIGNED TO THE DATA FOR USE IN REXX
*        &ERR=   IS THE LABEL TO BRANCH TO SHOULD AN ERROR OCCUR WHILE
*                CREATING THE REXX VARIABLE. BY DEFAULT IT IS ABEND ØØ1
*        &LEN=   IF THE DATA AT &LABEL IS NOT DEFINED SUCH THAT THE
*                LENGTH OF THE DATA IS WHAT YOU WANT, SIMPLY ENTER A
*                NUMBER HERE THAT DEFINES THE LENGTH REQUIRED. CAN ALSO
*                BE USEFUL IF IT IS NECESSARY TO DUMP OUT A LARGE AREA.
```

```
*        &SUBARRAY= IF A MULTI LEVEL ARRAY IS REQUIRED EG A.1.1 THEN
*                SET THIS VALUE ACCORDINGLY.
*        &DEBIN= IF THE DATA TO BE CREATED IS BINARY, SETTING THIS TO A
*                VALUE WILL CONVERT THE SPECIFIED NUMBER OF BYTES FROM
*                BINARY TO CHARACTER. THE DEFAULT LENGTH FOR THE
*                OUTPUT DATA IS 4 BYTES. IF THIS IS INSUFFICIENT, THEN
*                SPECIFY A SUITABLE &LEN VALUE TO OVERRIDE IT.
*        &LINK=  THIS IS A REXX NAME LABEL TO WHICH THE ARRAY COUNT IS
*                LINKED. THE PURPOSE OF THIS IS TO ALLOW A BRANCH OUT
*                OF ARRAY LOOPS WHILE STILL MAINTAINING NUMERIC
*                CONSISTENCY.
*
*********************************************************************
         PRINT GEN
         LCLA  &DEFLEN
&DEFLEN  SETA  16
         SHOWSET
LITLOC   LOCTR
&LABCHECK SETC '@_&ASNAME&SUBARRAY'
&LINKNAME SETC '@_&LINK'
         AIF   (D'&LABCHECK).BYPASS
         AIF   (T'&SUBARRAY EQ 'O').NORMNAME
&LABCHECK DC   C'&ASNAME..&SUBARRAY'
         AGO   .EOFARRAY
.NORMNAME ANOP
&LABCHECK DC   C'&ASNAME'
.EOFARRAY ANOP
&LABCHECK._ARRAY DC C'.       '
&LABCHECK._COUNTER DC PL4'Ø'                * COUNTER FIELD FOR THIS
ITEM
.BYPASS  ANOP
&SYSECT  LOCTR
         AIF   (T'&LINK EQ 'O').DOADD
         MVC   &LABCHECK._COUNTER,&LINKNAME._COUNTER
         AGO   .DOUNPK
.DOADD   ANOP
         AP    &LABCHECK._COUNTER,=P'1'      * INCREMENT THE COUNTER
THIS PASS
.DOUNPK  ANOP
         UNPK  @_UNPACKER,&LABCHECK._COUNTER * UNPACK THE VALUE
         OI    @_UNPACKER+7,X'FØ'            * REMOVE THE SIGN
* NOW NEED TO WORK OUT THE LENGTH OF THE COUNTER BIT TO ADD TO ARRAY
         L     R15,&LABCHECK._COUNTER        * LOAD THE COUNTER VALUE TO
WORK
*                                              OUT THE LENGTH
         SRL   R15,4                         * REMOVE THE SIGN
         XR    R14,R14                       * CLEAR R14 FOR A COUNTER
LOOP&SYSNDX DS ØH
         SRA   R15,4                         * MOVE DIGIT BY DIGIT
         LTR   R15,R15
```

```
         BZ    COUNT&SYSNDX
         LA    R14,1(,R14)
         B     LOOP&SYSNDX
COUNT&SYSNDX DS ØH
* NOW ADD COUNT FIELD TO NAME
         LA    R15,@_UNPACKER+7         * POINT TO END OF FIELD
         SR    R15,R14                  * AND COME BACK TO FIRST DIGIT.
         MVC   &LABCHECK._ARRAY+1(7),Ø(R15)
         LA    1,&LABCHECK
         ST    1,SHVNAMA
* NOW CALCULATE NEW LENGTH
         LA    1,L'&LABCHECK
         LA    1,2(R14,R1)
         ST    1,SHVNAML
         AIF   (T'&DEBIN EQ 'O').NORMLAB
*
*** NOW ALLOW FOR A BINARY CONVERSION
*** FIST CALCULATE THE ICM VALUE
*
&ICM     SETA  (1 SLL &DEBIN)-1
         XR    R15,R15
         ICM   R15,&ICM,&LABEL         * LOAD THE BINARY VALUE
         CVD   R15,@_DWORD             * CONVERT TO PACKED
         OI    @_DWORD+7,X'ØF'
         UNPK  @_UNPACK,@_DWORD
*
*** IF THE LEN VALUE IS SUPPLIED THIS OVERRIDES THE DEFAULT OF 16
*
         AIF   (T'&LEN EQ 'O').SETDEF  * LENGTH NOT SUPPLIED USE DEFLEN
&DEFLEN  SETA  &LEN                    * RESET DEFLEN TO SUPPLIED LEN
.SETDEF  ANOP
         LA    R1,@_UNPACK+(16-&DEFLEN)
         ST    R1,SHVVALA
         LA    R1,&DEFLEN
         AGO   .OK
.NORMLAB ANOP
         LA    1,&LABEL
         ST    1,SHVVALA
         AIF   (T'&LEN NE 'O').DOLEN
         LA    1,L'&LABEL
         AGO   .OK
.DOLEN   ANOP
         LA    1,&LEN
.OK      ANOP
         ST    1,SHVVALL
         LR    Ø,1Ø
         LA    1,COMS
         L     15,IRXEXCOM
         BALR  14,15
         LTR   15,15
```

```
         BNZ   &ERR
         MEND
         MACRO
         SHOWBASE &LABEL,&ERR=ABEND00,&SUBARRAY=
*********************************************************************
*
* MACRO TO CREATE REXX BASE VARIABLES
* SHOULD BE USED IN ASSOCIATION WITH A SHOWARAY MACRO. NOTE THAT A
* SHOWBASE MACRO IS OPTIONAL IF YOU ALREADY KNOW THE NUMBER OF
* VARIABLES BEING SET. THIS WILL CREATE THE A.0 ENTRY
*
* MACRO  FORMAT:
*        SHOWBASE &LABEL,&ERR=,&SUBARRAY=
* WHERE:
*        &LABEL IS THE NAME OF THE REXX ARRAY LABEL WHICH HAS BEEN
*               CREATED. THIS WILL CREATE THAT LABEL.0 ENTRY
*        &ERR=  IS THE LABEL WHICH BRANCH TO SHOULD AN ERROR OCCUR
*               WHILE CREATING THE REXX VARIABLE. BY DEFAULT IT IS
*               ABEND01.
*        &SUBARRAY= IF SUBARRAYS HAVE BEEN USED THIS WILL INSERT THE
*                   APPROPRIATE VALUE EG A.1.0
*
*********************************************************************
         SHOWSET
         AIF   (T'&SUBARRAY EQ 'O').NORMNAME
&ASNAME  SETC  '&LABEL..&SUBARRAY..0'
         AGO   .CHECKER
.NORMNAME ANOP
&ASNAME  SETC  '&LABEL..0'
.CHECKER ANOP
&LABCHECK SETC '@_&LABEL&SUBARRAY._COUNTER'
         AIF   (D'&LABCHECK).ITSOK
MNOTE    NO ARRAY ELEMENTS DEFINED.
         MEXIT
.ITSOK   ANOP
LITLOC   LOCTR
@_A&SYSNDX DC  C'&ASNAME'
&SYSECT  LOCTR
         LA    1,@_A&SYSNDX
         ST    1,SHVNAMA
         LA    1,L'@_A&SYSNDX
         ST    1,SHVNAML
         UNPK  @_UNPACKER,&LABCHECK
         OI    @_UNPACKER+L'@_UNPACKER-1,C'0'
         LA    1,@_UNPACKER
         ST    1,SHVVALA
         LA    1,L'@_UNPACKER
         ST    1,SHVVALL
         LR    0,10
         LA    1,COMS
         L     15,IRXEXCOM
```

```
         BALR  14,15
         LTR   15,15
         BNZ   &ERR
         MEND
         MACRO
         SHOWSET
         AIF   (D'SHOW_START).NONEED
         B     BY_SHOW_START
SHOW_START DS  ØH
         ST    R1Ø,COMRET
         LA    6,COMSHVB
         USING SHVBLOCK,R6
         XC    COMSHVB(SHVBLEN),COMSHVB
         XC    SHVNEXT,SHVNEXT
         MVI   SHVCODE,C'S'
         BR    14
ABEND ØØ1 DS   ØH
         ABEND 1    * REQUIRED FOR THE OTHER MACROS. SAVES SOME CODING.
BY_SHOW_START  DS ØH
LITLOC   LOCTR
@_UNPACK DC    CL16' '
         DC    CL8' '           * FILL FIELD
         ORG   @_UNPACK+8
@_UNPACKER DC  CL8' '
         ORG
@_DWORD  DS    CL8              * USED FOR THE DEBIN FUNCTION
&SYSECT  LOCTR
.NONEED  ANOP
         BAL   14,SHOW_START
         MEND
*********************************************************************
*
* THIS ROUTINE ANALYSES THE UCBS OF THE DASD STORAGE DEVICES ON THE
* SYSTEM AND RETRIEVES THE SPACE UTILIZATION DETAILS.
* THE VARIABLES CREATED ARE AS FOLLOWS:
*
* UNIT_TYPE.X ......THE DASD TYPE (339Ø/338Ø)
* VOLSER.X .........VOLUME SERIAL NUMBER
* FREE_EXTENTS.X ...NUMBER OF FREE EXTENTS.
* FREE_CYLINDERS.X  NUMBER OF FREE CYLINDERS
* FREE_TRACKS.X ... NUMBER OF FREE TRACKS.
* LARGEST_CYLINDER_EXTENT.X ... LARGEST CONTIGUOUS CYLINDER EXTENT
* LARGEST_TRACK_EXTENT.X ... LARGEST CONTIGUOUS TRACKS EXTENT
* FRAGMENTATION_INDEX.X ... THE FRAGMENTATION INDEX
* ADDRESS.X ........DEVICE ADDRESS (RETRIEVED FOR DASDR8)
* INDEX_STATUS.X ...IS THE VTOCIX ACTIVE OR NOT.
*
* NOTE:  THE BASE VARIABLE FOR ALL THESE ITEMS IS VOLSER.Ø
*
*********************************************************************
```

```
RVOLDATA TITLE 'REXX FUNCTION TO RETRIEVE DISK SPACE DETAILS'
*
RVOLDATA AMODE 31
RVOLDATA RMODE 24
RVOLDATA CSECT
         REXREGS
         BAKR  14,Ø
         LR    12,15
         USING RVOLDATA,12
*
         LR    R1Ø,RØ                  * R1Ø -> A(ENVIRONMENT BLOCK)
         USING ENVBLOCK,R1Ø
*
*
         L     R9,ENVBLOCK_IRXEXTE     * R9 -> A(EXTERNAL EP TABLE)
         USING IRXEXTE,R9
*
*
*
         STORAGE OBTAIN,LENGTH=GETLEN,ADDR=(8)
         STORAGE OBTAIN,LENGTH=RETLEN,ADDR=(7)
         USING COMSDS,R8
         USING RETAREA,R7
*
* PREPARE THE REXX AREA FOR USE
*
         XC    COMS(COMSLEN),COMS      * SET TO LOW VALUES
         LA    15,COMID
         ST    15,COMS
         LA    15,COMDUMMY
         ST    15,COMS+4
         ST    15,COMS+8
         LA    15,COMSHVB
         ST    15,COMS+12
         LA    15,COMRET
         ST    15,COMS+16
         OI    COMS+16,X'8Ø'           * INDICATE END OF PARMS
         MVC   COMID,=C'IRXEXCOM'
*
*
* COMMENCE THE LOOP OF UCBS LISTING ALL THE DEVICE TYPES.
*
         XC    HUNDRED,HUNDRED
*
*
UCBLOOP  DS    ØH
         UCBSCAN COPY,WORKAREA=HUNDRED,RANGE=ALL,UCBAREA=UCBWORK,      +
               DYNAMIC=YES,DEVCLASS=DASD,DEVNCHAR=MYDEV,DEVN=Ø,        +
               PLISTVER=MAX
*
```

```
* IF R15 CONTAINS Ø4, THEN LAST UCB RETRIEVED SO SET END FLAG
* FOR DEFENSIVE CODE, ASSUME ALL NON-ZERO RETURN CODES ARE THE
* EQUIVALENT OF END OF UCBS
*
         LTR   15,15            * END OF UCBS?
         BNZ   RETURN           * YES SO GO SET THE BASE ARRAY VALUES
         LA    R3,UCBWORK
         USING UCBOB,R3
         LA    R4,UCBDEV
         USING UCBDEV,R4
         TM    UCBSTAT,X'8Ø'    * IS THE DEVICE ONLINE?
         BNO   UCBLOOP          * NO, SO LOOP ROUND
*
*** NOW SET THE ADDRESS VARIABLE FOR DASDR8
*
         SHOWARAY MYDEV,ADDRESS * KEEP THE DEVICE ADDRESS
*
*** NOW LET'S CONFIRM THE DEVICE TYPE
*
         MVI ATTRIBS,X'ØA'      * NEEDS TO BE DONE PRIOR TO EDTINFO CALL
*
         EDTINFO RTNUNIT,DEVTYPE=UCBTYP,OUTUNIT=UNIT
         SHOWARAY UNIT,UNIT_TYPE
*
*** NOW LET'S GET THE FREE SPACE INFORMATION
*
         LSPACE UCB=(3),DATA=(7)
         SHOWARAY UCBVOLI,VOLSER
         SHOWARAY LSPDNEXT,FREE_EXTENTS,DEBIN=4
         SHOWARAY LSPDTCYL,FREE_CYLINDERS,DEBIN=4
         SHOWARAY LSPDTTRK,FREE_TRACKS,DEBIN=4
         SHOWARAY LSPDLCYL,LARGEST_CYLINDER_EXTENT,DEBIN=4
         SHOWARAY LSPDLTRK,LARGEST_TRACK_EXTENT,DEBIN=4
         SHOWARAY LSPDFRAG,FRAGMENTATION_INDEX,DEBIN=4
         TM    LSPDSTAT,X'4Ø'
         BO    INDEX_THERE
         SHOWARAY NO_INDEX,INDEX_STATUS
         B     UCBLOOP
INDEX_THERE DS ØH
         SHOWARAY INDEX_ACTIVE,INDEX_STATUS
         B     UCBLOOP
RETURN   DS    ØH
         SHOWBASE VOLSER
         STORAGE RELEASE,LENGTH=GETLEN,ADDR=(8)
         STORAGE RELEASE,LENGTH=RETLEN,ADDR=(7)
         PR
         LTORG
*
NO_INDEX DC C'INACTIVE'
INDEX_ACTIVE DC C'ACTIVE'
```

```
TDESCT   DSECT
         IEFUCBOB DEVCLASS=DA
*************************************************************************
*
*************************************************************************
***      IRXEXCOM PARAMETER AREA                                      ***
*************************************************************************
*
COMSDS   DSECT
COMS     DS    5AL4
COMID    DS    CL8              * IRXEXCOM ID - C'IRXEXCOM'
COMDUMMY DS    AL4              * NOT USED
COMSHVB  DS    (SHVBLEN)X       * IRXEXCOM SHVBLOCK (LENGTH FROM DSECT)
COMRET   DS    AL4              * IRXECOM RC
COMSLEN  EQU   *-COMS
HUNDRED  DS    XL100
         DS    0D
UCBWORK  DS    CL250
MYDEV    DS    CL4
UNIT     DS    CL8
ATTRIBS  DS    CL10
NAMES    DS    F
GETLEN   EQU   *-COMS
RETAREA  LSPACE MF=(D,DATA)
RETLEN   EQU   *-RETAREA
         DS    0D
         IRXEFPL
         IRXARGTB
         IRXEVALB
         IRXENVB
         IRXEXTE
         IRXSHVB
         END
/*
//LKED.SYSLMOD DD DSN=your.load.library,DISP=SHR,UNIT=
//LKED.SYSIN   DD *
         ENTRY RVOLDATA
         NAME  RVOLDATA(R)
```

SAMPLE.XLS

The following segment of a table was created from the sysout report by copying the output from SDSF into a file, and then downloading the data into Excel. Alternatively if you use the LPARHTML command, this will create an HTML table that when downloaded to your PC with an htm suffix, will display automatically as a table similar to the following.

```
Partition Address Volser Free_Extents Free_Cyls  Free_Trks  Large_Cyl  Large_Trk Index   Frag
PRD1      1417    ICE001      1       2415       36238      2415        36238    ACTIVE    0
PRD1      1416    ICE002      3       2381       35736      2381        35725    ACTIVE    0
PRD1      1413    ICE003      7       1309       19654      1305        19575    ACTIVE    3
PRD1      1412    ICE005     24       2064       34319      1655        24836    ACTIVE    222
```

LPARHTML

```
/* REXX */
/*                                                                    */
/**********************************************************************/
/*                                                                    */
/* The first part of this REXX checks to see if the socket has been   */
/* left active in error and terminates it. It then issues requests    */
/* to the specified server.                                           */
/*                                                                    */
/* This REXX requires that a string of information is supplied as     */
/* follows:                                                           */
/* WORD 1 - The ipaddress of the host. Use NONE if using the same     */
/*          host as the client.                                       */
/* WORD 2 onwards - the command string to issue to the server.        */
/* DS=dsname - If this is specified it should be the last parameter   */
/*             passed and should specify a datasetname to allocate.   */
/*             If not specified the REXX will attempt to create a     */
/*             suitable. It cannot however guarantee to be unique.    */
/*                                                                    */
/* It is a variant of the LPARQUIZ command in that this REXX creates  */
/* an output file for each host request which is then turned into an  */
/* HTML table display for subsequent Web disply.                      */
/* Note that for safety the SHUTDOWN command does not write to the    */
/* file.                                                              */
/*                                                                    */
/**********************************************************************/
/*                                                                    */
x='SOCKET'('SocketSetStatus')
/* */
IF WORD(x,1)='0' THEN DO
   x='SOCKET'('Terminate')
   END
/*                                                                    */
/**********************************************************************/
/*                                                                    */
/* An example of a client request REXX. This client sends a request   */
/* to the server so that it can carry out an action. This client      */
/* then retrieves the information line-by-line until the connection   */
/* is terminated by the server.                                       */
/*                                                                    */
/* If the ipaddress is set to NONE then a gethostid will be issued    */
/* to get the ipaddress of our host.                                  */
```

```
/* If the server detects an error with the request supplied from     */
/* this client, then the string SERVER ERROR is returned from the    */
/* server followed by the command that was sent through from here.   */
/*                                                                   */
/* Note the userid of the caller is also supplied to the server for  */
/* diagnostic purposes.                                              */
/*                                                                   */
/*********************************************************************/
/* */
ip_proc:
/* */
ARG string
/* */
PARSE VAR string ipaddress string 'DS=' holdds
/* */
/* Initialize control information */
/* */
IF holdds='' THEN DO   /* if no dsname supplied */
/* */
/* If the user has supplied a numeric ipaddress rather than  */
/* a user friendly host name then holdds will be altered to  */
/* allow the creation of a user friendly name of NONE.       */
/* */
   SELECT
   WHEN LENGTH(ipaddress)>8 THEN holdds=userid()||'.hlq.NONE' Á modify
to suit
   WHEN DATATYPE(LEFT(ipaddress,1),'M')=Ø THEN DO
        holdds=userid()||'.hlq.NONE' - modify to suit
        END
   WHEN DATATYPE(ipaddress,'A')=Ø THEN holdds=userid()||'.hlq.NONE'Á
modify to suit
   OTHERWISE DO
        UPPER ipaddress
        holdds=userid()||'.hlq.'||ipaddress - modify to suit
        END
   END
END
port = '1952'               /* The port used by the server */
/* */
/* Initialize */
/* */
x='SOCKET'('Initialize','RSCLIENT')
/* */
IF WORD(x,1)¬='Ø' THEN DO
   SAY 'error initialising RSCLIENT'
   EXIT
   END
/* */
IF ipaddress='NONE' THEN DO
  x='SOCKET'('GetHostId')
```

```
  IF WORD(x,1)¬='Ø' THEN DO
     SAY 'error trying to get host id'
     SIGNAL clean_up
     END
  ELSE ipaddress=WORD(x,2)
  END
/* */
/* Initialize for receiving lines sent by the server. */
/* */
x = 'SOCKET'('Socket')
/* */
IF WORD(x,1)¬='Ø' THEN DO
   SAY 'error issuing socket'
   SIGNAL clean_up
   END
/* */
/* pick up the client socket id */
/* */
clisock=WORD(x,2)
/* */
/* */
/* now get the host name */
/* */
x='SOCKET'('GetHostName')
/* */
IF WORD(x,1)¬='Ø' THEN DO
   SAY 'error getting host name'
   SIGNAL clean_up
   END
/* */
hostname = WORD(x,2)
/* */
/* now issue af_inet */
/* */
x='SOCKET'('Connect',clisock,'AF_INET' port ipaddress)
/* */
IF WORD(x,1)¬='Ø' THEN DO
   SAY 'error issuing af_inet'
   SIGNAL clean_up
   END
/* */
/* now send the information to the server */
/* */
x='SOCKET'('Send',clisock,userid() string)
/* */
IF WORD(x,1)¬='Ø' THEN DO
   SAY 'error issuing send'
   SIGNAL clean_up
   END
/* */
```

```
/* Wait for lines sent by the server */
/* */
DO FOREVER
/* */
/* now read the data. Data is returned as a rc len data field */
/* */
x='SOCKET'('Read',clisock)
/* */
IF WORD(x,1)¬='Ø' THEN DO
   PARSE VAR x . error
   SAY 'error issuing recv' error
   SIGNAL clean_up
   END
/* */
/* allow for the line being null. Abort the connection if it is. */
/* */
IF WORD(x,2)='Ø' THEN LEAVE
/* */
/* get the actual data */
/* */
PARSE VAR x . . dataline
/* */
/* As the data may have become strung together thanks to slow */
/* networks, the datalines have been prepared by the server   */
/* with a x'Ød' between the lines.                            */
/* */
DO UNTIL INDEX(dataline,'ØD'x)=Ø
   PARSE VAR dataline trueline 'ØD'x dataline
/* */
/* This is the point in the client REXX where the data is returned */
/* as a string and it is possible to insert your own processing.   */
/* */
   IF WORD(string,1)¬='SHUTDOWN' THEN CALL html_proc
   END
END
/* */
/* Terminate and exit */
/* */
clean_up:
IF WORD(string,1)¬='SHUTDOWN' THEN DO
   QUEUE '</TABLE>'
   QUEUE '</HTML>'
   ADDRESS TSO
   'FREE FI(SPONGE)'
   IF SYSDSN(holdds)¬='OK' THEN DO
      "ALLOC FI(SPONGE) CATALOG DA("holdds")",
      "DSORG(PS) REC(F B) LR(2ØØ) BLK(4ØØØ)",
      "SPACE(5,5) TRACKS"
   END
   ELSE "ALLOC FI(SPONGE) DA("holdds")"
```

```
   QUEUE ''
  'EXECIO * DISKW SPONGE (FINIS'
  /* */
  /* Issue a SAY to let the user know that the REXX has finished */
  /* */
   SAY 'Command' string 'has completed'
   SAY 'Please review file' holdds 'for success'
   'FREE FI(SPONGE)'
END
/* */
x='SOCKET'('Terminate','RSCLIENT')
RETURN
/* */
/* now create the HTML information */
/* */
html_proc:
IF ipaddress/='' THEN CALL html_header /* use this as a 1st time flag */
/* */
/* now generate the table rows */
QUEUE '<TR>'
DO x=1 TO WORDS(trueline)
QUEUE '<TD VALIGN="MIDDLE">'
QUEUE '<P><FONT FACE="Arial">'||WORD(trueline,x)||'</FONT></TD>'
END
QUEUE '</TR>'
RETURN
html_header:
QUEUE '<HTML>'
QUEUE '<HEAD>'
QUEUE '<TITLE>Sample Table</TITLE>'
QUEUE '</HEAD>'
QUEUE '<BODY LINK="#0000ff" VLINK="#800080">'
QUEUE '<FONT COLOR="#008000"><H1 ALIGN="CENTER">',
||'Results of' string 'Command</H1>'
QUEUE '<TABLE BORDER CELLSPACING=1>'
ipaddress=''
RETURN
```


Internet resources for Linux on System/390

INTRODUCTION

On 18 December, 1999, IBM published its modifications and additions to the Linux 2.2.13 code base for the support of the S/390 architecture. In *MVS Update* Issue 168, September 2000, we considered how IBM's 'Linux for S/390' port has given users the ability to support Open Source Linux applications on the System/390 without the need for modification to the source code, and how this could have a radical impact on the future of the computer industry.

This article documents some of the Internet resources that are available for systems programmers who are deploying, or considering deploying Linux on the System/390. We recommend that users 'bookmark' links on their browsers that they visit regularly. This will save you time, because you will not need to re-type the URL (Universal Resource Locator) each time you access the site.

THE RESOURCES

The following Web sites are recommened for those researching or deploying Linix for S/390.

http://www.s390.ibm.com/linux/

This is IBM's official homepage for Linux for S/390 and is a good starting point both for those who are considering a Linux deployment and for those who have already got Linux for S/390 running. The site has a Linux for S/390 in the news section, the latest announcements and press releases, and a good links page. It also has links to Linux Web publications such as Linux Journal, Linux Focus, Linux Headquarters, LINUX.COM, and Linux Online. The IBM Linux FAQ (Frequently Asked Questions) for System/390 page (http://www.s390.ibm.com/linux/faq_intro.html) is an extremely useful resource for those who require background information quickly.

http://oss.software.ibm.com/developerworks/opensource/linux390/index.html

The IBM Linux for S/390 site is the original location at which IBM made its patches the kernel and compilers available. This page is well linked to other IBM sites with background material about OS/390, VM, Linux, and their relationship.

Users can go to the OS download area to get the modifications (source patches) needed to build Linux for S/390. These patches are available free of charge and, except for the patches of strace which is under a 'distributable license', are distributed under the GPL/LGPL license. Users who need the network device driver need to go to the OCO (Object Code Only) download area. The network device driver is available free of charge as OCO under an IBM license.

http://www.marist.edu/linuxvm

The Marist College site, in Poughkeepsie, is an IBM sponsored site that provides pre-compiled kernel and disk images, installation instructions, and a QuickStart guide. Much of the development work covering the Linux for S/390 port is supported here. This is the place to go when you are ready to install Linux for S/390. The Marist College site has the most active mailing list covering Linux on System/390.

http://vm.ibm.com/linux

The VM/ESA Linux page is the starting point for VM product information at IBM. Using VM/ESA to support Linux could turn out to be the 'killer app' for Linux on the mainframe. Tens of thousands of Linux images can be supported on a single System/390 running VM/ESA, while the new S/390 'Virtual Image Facility' only enables users to run "tens to hundreds" of Linux images on a single System/390. This will certainly extend the lifespan of the VM operating system, although IBM has been trying to migrate VM users to OS/390 for a long time. This site is certainly worth a visit for users who are still using VM/ESA.

http://www.linas.org/linux/i370.html

The original port to the S/390, called Bigfoot or i370 port, was initiated by an interested group of programmers headed by Linas Vepstas. However, before the i370 port could reach maturity, IBM announced its own port, which has essentially superseded Bigfoot. Since the IBM port is the one receiving most or all of the active development work today, the Bigfoot project is suspended. However, the *Linux on the IBM ESA/390* site is still a useful resource for those interested in the issues of running Linux on the System/390.

DOCUMENTATION

Overabundance of documentation has long been recognized as a weakness by the Linux community and efforts are being made to bring the Linux documentation in line with that provided by commercial systems. This is not a problem with Linux for S/390 because it is specifically designed for the corporate environment by IBM.

http://www.redbooks.ibm.com/redpieces/abstracts/sg244987.html

The Linux for S/390 Redbook is probably the most comprehensive documentation for Linux on the S/390. This is available for download in PDF format and covers everything from downloading to running 'Linux for S/390'. This is essential reading for those considering a deployment. This page also links back to the IBM Redbook homepage (http://www.redbooks.ibm.com), which contains the IBM Redbooks, Redpieces, and Redpapers.

http://www.linuxdoc.org/

The Linux Documentation Project (LDP) has links to most of the important manuals relating to all aspects of Linux. Paradoxically the problem with Linux is not scarcity of documentation, but an overabundance of it, which makes it difficult to search. The Linux Documentation Project attempts to provide a centralized access point to Linux documents. Technically, a Web-based document library is better than physical manuals, because it allows for dynamic updates as and when they are needed.

There are a variety of non-IBM sites on the Web that provide information about running Linux of the System/390. Both the quality and the diversity of the information on these sites is excellent.

http://linux.s390.org/download/rpm2html/

The Think Blue Linux distribution contains nearly five hundred pre-compiled RPM's for Linux/390. This is currently the *de facto* Linux distribution for the System/390. One of the goals of the rpm2html site is to identify the dependencies between various packages and to find the package(s) providing the resources needed to install a given package. Every package is analyzed to retrieve its dependencies and the resources it offers. These relationships are expressed using hyperlinks making the site very easy to use.

http://penguinvm.princeton.edu/

The Linux/390 at Princeton University site is run by Adam Thornton and hosts a wide variety of material devoted to Linux for S/390. Princeton University is a big VM site, so much of the material is devoted to Linux and VM. The site provides a good overview of Linux and the System/390.

http://penguinvm.princeton.edu/~neale/linux390.htm

The Linux/390 – Notes and Observations site at Princeton University is run by Neale Fergusen (one of the team who worked with Linas Vepstas on the Bigfoot port). The site provides a technical overview of the Linux kernel internals. Some very useful and detailed information is provided on topics as diverse as common device support, DASD device driver, Linux system calls, control register usage, access register usage, IPL under VM/ESA, copying file systems, debugging on Linux for S/390, simplified network access, etc.

http://source.rfc822.org/pub/mirror/s390-ibm-linux/Peter.Schulte-Stracke/

The standard distribution of Linux for S/390 comes with a compiler that produces code that can run only on newer post-G2 S/390 equipment, because it exploits the Halfword Immediate and Relative Branch Feature. Peter Shulte-Stracke's *Vintage for S/390* site has a set of scripts that provide a way of running the IBM-supplied S/390 kernel on pre-G2 machines. This site is invaluable for the numerous users who are supporting older hardware. Peter Shulte-Stracke was another member of Linas Vepstas's Bigfoot team – hence the interest in providing this crucial pre-G2 support. Peter has also provided an e-mail contact on his site for those who have any queries about the patches.

GENERAL LINUX SITES

The following general Linux sites provide additional information that will be of use to those deploying or supporting Linux.

http://www-4.ibm.com/software/is/mp/linux/

The Linux at IBM site has a useful array of resources, which cover all of IBM's Linux initiatives. This is useful for putting IBM's Linux for S/390 initiative into a wider enterprise perspective. The site also has links to Linux certification and training options in the industry, and links to the IBM Linux computing centres across Europe and North America. IBM is devoting hundreds of millions of dollars to Linux development, this site shows users what has been achieved.

http://linas.org/linux/index.html

The Enterprise Linux site is produced and maintained by Linas Vepstas. This is a well-organized site that provides a large number of links to all aspects of enterprise Linux use. This is a particularly useful site for those users who require management support for a 'Linux for S/390' deployment (for example, there are lists of sites, and case studies with cost benefits from large organizations who have deployed Linux for S/390).

http://www.ibm.com/developer/linux/

The DeveloperWorks Linuxzone site has a weekly newsletter, a tips section, columns, and Linux tools for download. The site is oriented towards Linux generally rather than Linux for S/390 specifically. But it is still a useful resource for developers.

http://www.opensource.org/

Linux is Open Source software that may be downloaded free of charge. The Open Source site provides information on the wide variety of open source initiatives currently active, including Linux. This is very useful for enterprise users who need to put Linux for S/390 into context.

NEWSGROUPS AND LISTSERVERS

Perhaps the greatest aid to the systems programmer introduced by the Internet is the newsgroup or listserv. These essentially e-mail-based services take on a new role as an almost instantly available expert assistant.

These e-mail lists have been around for a while. IBM-MAIN, the most widely subscribed to general System/390 and OS/390 group started back in the mid-'80s. However, for detailed information on deploying and running Linux on the System/390 users should consult specialist lists.

http://www.marist.edu/htbin/wlvindex?linux-vm

As mentioned previously the Marist College site, in Poughkeepsie has the most active mailing list covering Linux on System/390. You can use this site to join the Linux-390 mailing list. This list is actually devoted to all environments in which Linux for System/390 will run. To join the mailing list simply send an e-mail to the listserver (listserv@vm.marist.edu) with a body containing SUBSCRIBE LINUX-390 *YOUR NAME*.

The mailing list archives for the Marist College S/390 Linux listserv, represent a huge body of useful information. This invaluable resource goes back to 1998 and a unique historical record of the Linux development for the System/390.

IBM's e-mail contact

The IBM 'Linux for S/390' port emanated from a 'Skunk Works' project in IBM's laboratories in Germany, which was developing a GCC compiler that could produce System/390 machine instructions. When this was completed the team compiled the Linux source code. It is here that IBM provides an e-mail contact (linux390@de.ibm.com), where users can send problems specific to System/390 implementation of the kernel, glibc, and the compiler.

CONCLUSION

There are considerable benefits to be gained from deploying Open Source software such as Linux in an enterprise environment. Linux has become a serious contender as a mid-range corporate server operating system and there is the potential for the operating system to make inroads into the desktop environment, although this still has some way to go yet. Using the System/390 as a platform for a Linux deployment provides an enviable cost benefit. Reviewing the Web sites in this article will provide users with an invaluable resource for their Linux deployment.

IBM has announced Version 4.1 of WebSphere Commerce Suite, Pro Edition for OS/390, formerly Net.Commerce, using WebSphere Commerce Studio as the new design and development environment.

Based on WebSphere Studio, it is available in the Developer and Professional Developer Editions, and is used to develop the commerce site. Each includes the features of the WebSphere Studio with additional commerce tools to support the Commerce Suite. The Professional Developer Edition includes WebSphere Catalog Architect (CA), V4.1, tools and support for rules-based personalization, tools for setting up auctions, and Product Advisor tools.

WebSphere Catalog Architect is used to create, import, and update product information, reduce the time for traditional catalogue information management, selectively publish information into production, automate validation of catalogue information, and provide spreadsheet and drag and drop interfaces. WebSphere Commerce Suite, Pro Edition for OS/390 does not include WebSphere Commerce Studio or Catalog Architect.

WebSphere Catalog Architect provides features and functions for content management, relationship marketing, ordering, and payment management for business transformation, integration, and growth, it manages the interface to the buyers, who can browse, save, query, and order items in the interactive catalogue.

Contact your local IBM representative for further information.

http://www.ibm.com

* * *

Advanced Software Products Group has released its WizDom suite of multi-function general file manipulation tools for ongoing System/390 maintenance.

File maintenance lets users investigate and directly update files and databases using a series of ISPF panels. Direct update of sequential, VSAM, IMS/DB, DB2, and IDMS files and databases are supported. File ageing allows users to increment dates on files and databases, which speeds up the creation of meaningful test data.

The impact analysis/program edit tool helps during installations, addressing changes that may be necessary to any mainframe applications. It will analyse code in any language including COBOL, PL/I, Assembler, Mantis, Ideal, Natural, Focus, JCL, Easytrieve, Mark IV, and others.

The date/time simulation feature allows users to execute any of their programs, both batch and on-line, under a simulated system date of choice, independent of the actual system date.

For further information contact:

Advanced Software Products Group, 3185 Horseshoe Drive South Naples, Florida 34104-6138, USA.
Tel: 941 649 1548
Fax: 941 649 6391

Computerbility, Ltd, Midland House, 5th Floor, New Road, Halesowen, West Midlands, B63 3HY, UK.
Tel: (0121) 550 6262
Fax: (0121) 550 6363

http://www.aspg.com

* * *