126 VM

February 1997

update

In this issue

Published by
Xephon
27-35 London Road
Newbury
Berkshire RG14 1JL
England
Telephone: 01635 38342
From USA: 01144 1635 38342
E-mail: xephon@compuserve.com

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75067
USA
Telephone: 817 455 7050

Australian office
Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 08 223 1391

Editorial panel
Articles published in *VM Update* are reviewed by our panel of experts. Members of the panel include John Illingworth (UK), Reinhard Meyer (Germany), Philippe Taymans (Belgium), Romney White (USA), Martin Wicks (UK), and Jim Vincent (USA).

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

Editor
Trevor Eddolls

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

Subscriptions and back-issues
A year's subscription to *VM Update*, comprising twelve monthly issues, costs £165.00 in the UK; $250.00 in the USA and Canada; £171.00 in Europe; £177.00 in Australasia and Japan; and £175.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1990 issue, are available separately to subscribers for £14.00 ($21.00) each including postage.

VM Update on-line
Code from *VM Update* can be downloaded from our Web site at http://www.xephon.com; you will need the user-id shown on your address label. Code is also available from our bulletin boards in the USA (630 980 4581 or 4751) or the UK (01635 30998); you will need the user-id and password shown on your address label.

Printed in England.

Generate system components

We don't very often change things in our VM environment, but there are times when new equipment is added or an emergency fix is applied. Seeing that system generations are so far apart, it's easy to forget what steps are needed to build the new system without hunting down the right manuals and then finding the right procedures. To that end I was prompted to write the GENNEW EXEC.

At our site, we use ACF2 for security. Any nucleus that is put into production has to include this. But, as any good sysprog who uses a security package knows, there are times when it breaks. I like to keep a non-secured copy of the nucleus around, just in case of an emergency. I set up GENNEW to build both, based on the parameters passed.

Since I have a tendency to forget what type of PARM to pass, there is even some checking code up front, in case nothing is entered. The common routines are broken out to prevent code duplication. No pipe commands were used during the creation of this EXEC.

HCPRIO should be rebuilt if you have added new equipment and are running with a preferred guest, and you want guest recovery in effect. If you sense your devices at start-up and define them in the CONFIG file, this isn't necessary. The same holds true for HCPAC0 – the ACF2 intercept. If there have been changes to ACF2, you must rebuild this module. Both have been set up to allow the new text deck to go to the LOCAL MODS disk.

If you run another security package, such as Top Secret, RACF, or VM:Secure, you will need to modify accordingly.

GENNEW performs the following steps:

- Re-assembles HCPRIO (optional)
- Re-assembles HCPAC0 (optional)
- Builds an ACF2 CP or non-AFC2 CP
- Examines the load map (should have one match on 'undefined')
- Moves the new nucleus to the system parameter disk after backing up the old one.

The move is optional in either case.

The EXEC is invoked with one of two arguments. If CP is used, a nucleus without ACF2 is created. If ACFCP is specified, then you get the ACF2 version. If you invoke with no options, you will be prompted.

GENNEW EXEC

```
/*****************************************************************/
/*  Generate system components: CP  ACF2CP                       */
/*  Invocation: GENNEW compname                                  */
/*  Result:  Set-up mini-disks for proper component              */
/*           Assemble any required files                         */
/*           Validate results - check maps, etc.                 */
/*           If desired, copy new modules to production          */
/*****************************************************************/
Trace O
Arg compname
 Select
  When compname = 'CP' then signal Do_CP
  When compname = 'ACFCP' then signal Do_ACF
  When compname = ' ' then do
     Say 'Please enter CP for no ACF2, or ACFCP for ACF2....'
     Pull compname
     If compname = 'CP' then signal Do_CP
      Else signal Do_ACF
   End
 End
/*****************************************************************/
Do_CP:                  /* Build a non-ACF2 nucleus */
 VMFCLEAR
 Say 'This part of the EXEC builds a CPNUC with no ACF2...'
 Say 'Is that what you want to do?'
 Say 'Enter Y to continue, N to EXIT, or ACF2 to build with ACF2.'
 Pull Ans
  Select
   When Ans = 'N' then Exit 99
   When Ans = 'ACF2' then signal Do_ACF
   When Ans = 'Y' then nop
  End
 Exec VMFSETUP ESA CP
 VMFCLEAR
 Say 'Do you want to rebuild HCPRIO? (Y/N)'
 Pull YorN
    If YorN = 'Y' Then CALL Bld_RIO
 Say 'I will now build the new CP nucleus - this takes a few minutes...'
 Exec VMFBLD PPF ESA CP CPLOAD '(ALL SETUP'
```

```
 Say 'Review the build output....'
 'CP Sleep 5 Sec'
 Exec VMFVIEW BUILD
 Call Chk_Map
 VMFCLEAR
 Say 'Do you want to move this nucleus to production? (Y/N)'
 Pull YorN
 If YorN = 'N' then Exit Ø
    Else CALL Move_To_Prod
 Say ' Your new nucleus now resides on the primary PARM disk...'
 Say ' It will be active at the next system IPL....'
Exit Ø
/*********************************************************************/
Do_ACF:                 /* Build an ACF2 nucleus    */
 'CP LINK ACFMAINT 2AØ 2AØ MW'
 'CP LINK ACFMAINT 2A1 2A1 MW'
 'CP LINK ACFMAINT 2A3 2A3 MW'
 Exec VMFSETUP ACFE22 CP
 VMFCLEAR
 Say 'Do you want to rebuild HCPRIO? (Y/N)'
 Pull YorN
    If YorN = 'Y' Then CALL Bld_RIO
 VMFCLEAR
 Say 'Do you want to rebuild HCPACØ? (Y/N)'
 Pull YorN
 If YorN = 'Y' then do
    Say 'I will rebuild HCPACØ - this may take a few minutes...'
    'Exec VMFHLASM HCPACØ ACFE22 CP (LOGMOD OUTMOD LOCALMOD'
 End
 Say 'I will now build the new CP nucleus - this takes a few minutes...'
 Exec CAXABLD ACFCP VMFBLD PPF ACFE22 CP CPLOAD '(ALL SETUP'
 Say 'Review the build output....'
 'CP Sleep 5 Sec'
 Exec VMFVIEW BUILD
 VMFCLEAR
 Call Chk_Map
 VMFCLEAR
 Say 'Do you want to move this nucleus to production? (Y/N)'
 Pull YorN
 If YorN = 'N' then Exit Ø
    Else Call Move_To_Prod
 Say ' Your new ACF2 NUC now resides on the primary PARM disk...'
 Say ' It will be active at the next system IPL....'
Exit Ø
/*********************************************************************/
/*    Subroutines follow                                             */
/*      Bld_RIO - rebuild HCPRIO                                     */
/*      Chk_Map - check nucleus maps for unresolved undefined        */
/*      Move_To_Prod - move new nucleus to production                */
/*********************************************************************/
```

```
Bld_RIO:
    Say 'I will rebuild HCPRIO - this may take a few minutes...'
    'Exec VMFHLASM HCPRIO ESA CP (LOGMOD OUTMOD LOCALMOD'
Return
/*****************************************************************/
Chk_Map:
 VMFCLEAR
 Say 'Look for undefined or unresolved in CPNUC map....'
 Say 'If any are found, resolve before moving to production...'
 'CP Sleep 3 Sec'
 Queue 'top'
 Queue 'set case mixed ignore'
 Queue 'all/unresol/ & all/undefin/'
 'Xedit CPLOAD MAP *'     /* The map is on the Maint 493 mdisk */
Return
/*****************************************************************/
Move_To_Prod:
    'CPREL A'; 'CP LINK * CF1 CF1 MW'
    'CPREL B'; 'CP LINK * CF2 CF2 MW'
    'Access CF1 X'; 'Access CF2 Z'
/* If it is ACF2 we copy CPLOAD, else replace NOACF2 */
/* For our site when ACF2 is used 493 = N, else 493 = K for FM */
  Select
   When compname = 'ACFCP' then do
    'Erase CPLOAD MODULE Z'              /* remove old from CF2     */
    Say 'Backing up old nucleus to secondary parm disk...'
    'COPYF CPLOAD MODULE X = = Z (OLDD'  /* back-up old from CF1-CF2*/
    'Erase CPLOAD MODULE X'              /* remove old from CF1     */
    Say 'Copying new nucleus to primary parm disk...'
    'COPYF CPLOAD MODULE N = = X'        /* put new one on CF1      */
    Signal Reacc_CP
   End
   When compname = 'CP' then do
    'Erase NOACF2 MODULE Z'              /* remove old from CF2     */
    Say 'Backing up old nucleus to secondary parm disk...'
    'COPYF NOACF2 MODULE X NOACF2 MODULE Z (OLDD' /* backup old     */
    'Erase NOACF2 MODULE X'              /* remove old from CF1     */
    Say 'Copying new nucleus to primary parm disk...'
    'COPYF CPLOAD MODULE K NOACF2 MODULE X (OLDD' /* copy in new    */
    Signal Reacc_CP
   End
 End
Reacc_CP:
    'Release X'; 'Release Z'
    'CPACCESS MAINT CF1 A RR'
    'CPACCESS MAINT CF2 B RR'
Return
```

Daniel A McLaughlin
Senior Technical Support Analyst (USA)

Dynamic menus system for CMS – part 2

This month we continue the code for a dynamic menu system, which creates procedures needed by users ‘on the fly’.

LOGO EXEC

```
/* LOGO EXEC */
'SET LANGUAGE (ADD TAF USER'
TITLE=CENTER('YOUR COMPANY NAME',22)
SUBTITLE=CENTER('YOUR DEPARTMENT',22)
MESSAGE.1=''
/* SET HERE THE NAME OF THE PASSWORDS CHECKING SERVICE MACHINE */
SERVPASW='SYSMAINT'
'PIPE CP Q SET | SPLIT AT /,/ | LOCATE / SMSG / | SPECS W2 1 | VAR SMSG'
/* IS PASSWORDS SERVER ON AND ACTIVE ? */
'PIPE CP SMSG 'SERVPASW' TEST | STEM CP_MSG.'
IF CP_MSG.Ø¬=Ø THEN DO                  /* MENU SYSTEM PASSWORDS      */
   CALL MENU USERID()                   /* VERIFICATION IS NOT ACTIVE */
   'CP SET SMSG 'SMSG
   EXIT
   END
$F1='PFØ1'; $F2='PFØ2'; $F3='PFØ3'; $F4='PFØ4'; $F5='PFØ5'; $F6='PFØ6'
$F7='PFØ7'; $F8='PFØ8'; $F9='PFØ9'; $7A='PF1Ø'; $7B='PF11'; $7C='PF12'
$C1='PF13'; $C2='PF14'; $C3='PF15'; $C4='PF16'; $C5='PF17'; $C6='PF18'
$C7='PF19'; $C8='PF2Ø'; $C9='PF21'; $4A='PF22'; $4B='PF23'; $4C='PF24'
$Ø1='PA1'; $6E='PA2'; $7D='ENTER'; $6D='CLEAR'
'PIPE CP Q CON | TAKE 1 | SPECS 14-21 1.8 RIGHT | VAR TERMID'
USER_NAME=RIGHT(USERID(),8)
LOGO.=''
'VMFCLEAR'
 LOGO :
LOGO.2='';LOGO.3=''
ERROR_MESSAGE=MESSAGE.1
TOWER=,
'11ØØØ52903C02Ø410042F62841F2'X||,
'   '||'284ØØ'X||'   '||'2841F2'X||'   '||'284ØØ'X||'
'||'2841F2'X||,
'   '||'284ØØ'X||'   '||'2841F2'X||'   '||'28ØØØØ'X||,
'11ØØ552903C02Ø410042F62841F2'X||,
'   '||'284ØØ'X||'   '||'2841F2'X||'   '||'284ØØ'X||'
'||'2841F2'X||,
'   '||'284ØØ'X||'   '||'2841F2'X||'   '||'28ØØØØ'X||,
'11ØØA52903C02Ø410042F62841F2'X||'                    '||'28ØØØØ'X||,
'11ØØF92903C02Ø410042F62841F2'X||'             '||'28ØØØØ'X||,
'11Ø1492903C02Ø410042F62841F2'X||'             '||'28ØØØØ'X||,
```

```
'1101992903C020410042F62841F2'X||'             '||'280000'X||,
'1101E92903C020410042F62841F2'X||'             '||'280000'X||,
'1102332903C020410042F62841F2'X||'
'||'280000'X||,
'1102852903C020410042F62841F2'X||'                   '||'280000'X||,
'1102D72903C020410042F62841F2'X||'                '||'280000'X||,
'1103272903C020410042F62841F2'X||'                '||'280000'X||,
'1103772903C020410042F62841F2'X||'                '||'280000'X||,
'1103C02903C020410042F62841F2'X||'   '||'284100'X||'    '||'2841F2'X||,
'   '||'284100'X||'    '||'2841F2'X||'   '||'280000'X||,
'284100'X||'    '||'2841F2'X||'   '||'284100'X||'    '||'2841F2'X||,
'   '||'280000'X||,
'1104102903C020410042F62841F2'X||'   '||'284100'X||'    '||'2841F2'X||,
'   '||'284100'X||'    '||'2841F2'X||'   '||'280000'X||,
'284100'X||'    '||'2841F2'X||'   '||'284100'X||'    '||'2841F2'X||,
'   '||'280000'X||,
'1104612903C020410042F62841F2'X||'             '||'284100'X||'   '||,
'2841F2'X||'             '||'280000'X||,
'1104B32903C020410042F62841F2'X||'          '||'284100'X||'     '||,
'2841F2'X||'          '||'280000'X||,
'1105052903C020410042F62841F2'X||'        '||'284100'X||'     '||,
'2841F2'X||'        '||'280000'X||,
'1105552903C020410042F62841F2'X||'                   '||'280000'X||,
'1105A52903C020410042F62841F2'X||'         '||'284100'X||'   '||,
'2841F2'X||'         '||'280000'X||,
'1105F52903C020410042F62841F2'X||'        '||'284100'X||'     '||,
'2841F2'X||'        '||'280000'X||,
'1106452903C020410042F62841F2'X||'        '||'284100'X||'     '||,
'2841F2'X||'        '||'280000'X||,
'1106932903C020410042F62841F2'X||'          '||'284100'X||'     '||,
'2841F2'X||'          '||'280000'X||,
'1106E12903C020410042F62841F2'X||'                            '||,
'280000'X||,
'11072F2903C020410042F62841F2'X||'                               '||,
'280000'X
SCREEN='8003284100'X||TOWER||,
'1100212903C020410042F4'X||'TERMINAL ID:'||TERMID||'1100361DF0'X||,
'1100712903C020410042F4'X||' USER NAME :'||USER_NAME||'1100861DF0'X||,
'1100442903C020410042F6'X||CENTER(DATE(W),10)||,
'1100942903C020410042F6'X||DATE()||,
'1100E42903C020410042F6'X||TIME()||,
'11016C2903C02041F242F1'X||TITLE||'1101831DF0'X||,
'11025C2903C02041F242F2'X||SUBTITLE||'1102731DF0'X||,
'11052C2903C02041F242F3'X||'MENU NAME :  '||'11053B1D20'X||,
'11053D2903C00141F442F3'X||LOGO.1||'1105461DF0'X||,
'1105CC2903C02041F242F3'X||'PASSWORD :   '||'1105DB1D20'X||,
'1105DD2903C00D41F442F3'X||LOGO.2||'1105E61DF0'X||,
'11066C2903C02041F242F3'X||'NEW PASSWORD:'||'11067B1DF0'X||,
'11067D2903C00D41F442F3'X||LOGO.3||'1106861DF0'X||,
'1107512903C02041F142F2'X||ERROR_MESSAGE||'11077F1DF0'X||,
```

```
'11Ø53E13'X
'PIPE (ENDCHAR ?) VAR SCREEN | FULLSCREEN ',
'| SPLIT AT ANYOF /'"11"X'/',
'| A: FANOUT',
'| TAKE FIRST',
'| SPECS /$/ 1 1-1 C2X NEXT',
'| VAR AIDKEY',
'? A:',
'| DROP FIRST',
'| SPECS 3-* 1',
'| STEM LOGO.'
MESSAGE.1=''
SELECT
WHEN VALUE(AIDKEY)='PF24' | VALUE(AIDKEY)='PF12' THEN DO
     'CP SET SMSG 'SMSG
     EXIT
     END
WHEN VALUE(AIDKEY)='CLEAR' THEN DO
     MESSAGE.1=''
     SIGNAL LOGO
     END
WHEN VALUE(AIDKEY)='ENTER' THEN DO
   SELECT
   WHEN STRIP(LOGO.1)='' THEN DO            /* NO MENU ID  */
     'XMITMSG 12 (APPLID TAF CALLER LOG NOCOMP VAR'
     END
   WHEN STRIP(LOGO.2)='' THEN DO            /* NO PASSWORD */
     'XMITMSG 13 (APPLID TAF CALLER LOG NOCOMP VAR'
     END
   WHEN STRIP(LOGO.2)=STRIP(LOGO.3) THEN DO /* NEW PASSWORD = OLD ONE */
     'XMITMSG 14 (APPLID TAF CALLER LOG NOCOMP VAR'
     END
   OTHERWISE DO
     'CP SET SMSG ON'
     'SMSG ' SERVPASW LOGO.1 LOGO.2 LOGO.3
     'SMSG ' SERVPASW LOGO.1 LOGO.2 LOGO.3
     'WAKEUP (SMSG QUIET'
     PARSE PULL TYPE SERVER TEXT
SELECT
 WHEN STRIP(TEXT)='GOGOGO' THEN DO
  CALL MENU LOGO.1
 'XMITMSG Ø (APPLID TAF CALLER LOG NOCOMP VAR'
  LOGO.=''
  END
 WHEN STRIP(TEXT)='NOTAF' THEN DO
 'XMITMSG 16 LOGO.1 (APPLID TAF CALLER LOG NOCOMP VAR'
  END
 WHEN STRIP(TEXT)='NOMATCH' THEN DO
 'XMITMSG 17 (APPLID TAF CALLER LOG NOCOMP VAR'
  END
```

```
 WHEN STRIP(TEXT)='ADDPSWDNOTOK' THEN DO
 'XMITMSG 18 (APPLID TAF CALLER LOG NOCOMP VAR'
  END
 WHEN STRIP(TEXT)='ADDPSWDOK' THEN DO
 'XMITMSG 2Ø LOGO.1 (APPLID TAF CALLER LOG NOCOMP VAR'
  SIGNAL LOGO
  END
 WHEN STRIP(TEXT)='NOADDPSWD' THEN DO
 'XMITMSG 21 LOGO.1 (APPLID TAF CALLER LOG NOCOMP VAR'
  SIGNAL LOGO
  END
 WHEN STRIP(TEXT)='CHDPSWDOK' THEN DO
 'XMITMSG 19 (APPLID TAF CALLER LOG NOCOMP VAR'
  END
 WHEN STRIP(TEXT)='PSWDSHUT' THEN DO
 'VMFCLEAR'
  SAY 'PASSWORD SERVICE MACHINE HAS BEEN ORDERLY SHUT DOWN'
  SAY 'PRESS <ENTER> TO GET YOU DYNAMIC MENU'
  PULL ANSWER
  CALL MENU USERID()                    /* VERIFICATION IS NOT ACTIVE */
  END
 OTHERWISE DO
 'XMITMSG 15 (APPLID TAF CALLER LOG NOCOMP VAR'
  END
END
     END  /* OTHERWISE */
   END
   SIGNAL LOGO
END /* END ENTER */
OTHERWISE DO
     'XMITMSG 1 'VALUE(AIDKEY)' (APPLID TAF CALLER LOG NOCOMP VAR'
     SIGNAL LOGO
     END
END
RETURN
```

MENU EXEC

```
/* MENU EXEC */
'VMFCLEAR'
DATA_FILES_DISK='1ØØ'          /*THIS IS THE ADDRESS OF THE MINI-DISK */
                               /* WHERE WE FIND THE DATABASE FILES   */
PARSE UPPER ARG USER_NAME
IF USER_NAME='' THEN USER_NAME=USERID()
USER_NAME=LEFT(STRIP(USER_NAME),8)
TITLE=CENTER('MENU FOR USER '||STRIP(USER_NAME),28)
MENU_TITLE=CENTER('YOUR COMPANY NAME',27)
'SET LANGUAGE (ADD TAF USER '
MENU_START:
```

```
'SET CMSTYPE HT';'ACC 'DATA_FILES_DISK' X/A';'SET CMSTYPE RT'
$F1='PF01'; $F2='PF02'; $F3='PF03'; $F4='PF04'; $F5='PF05'; $F6='PF06'
$F7='PF07'; $F8='PF08'; $F9='PF09'; $7A='PF10'; $7B='PF11'; $7C='PF12'
$C1='PF13'; $C2='PF14'; $C3='PF15'; $C4='PF16'; $C5='PF17'; $C6='PF18'
$C7='PF19'; $C8='PF20'; $C9='PF21'; $4A='PF22'; $4B='PF23'; $4C='PF24'
$01='PA1'; $6E='PA2'; $7D='ENTER'; $6D='CLEAR'
ERROR_MESSAGE=COPIES(' ',63)
NXTMENU=0
L_MORE_ATTR='C03C41004200'X
R_MORE_ATTR='C03C41004200'X
/* GET MENU NAMES FROM AUTHORIZATION FILE */
NO_AUTH='! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! ! !'
'PIPE < MENU XAUTH * ',
'| LOCATE (1.8) /'USER_NAME'/',
'| NLOCATE /'NO_AUTH'/',
'| SPECS 10-17 1.8',
'| STEM TAFRIT.'
IF TAFRIT.0=0 THEN DO          /* NO MENU FOR THIS GUY */
        'VMFCLEAR'
        SAY 'THERE IS NO MENU FOR USER 'USER_NAME
        RETURN
 END
PF10PF11:                      /* SCROLL THROUGH MENUS LIST */
'PIPE STEM TAFRIT.',
'| DROP 'NXTMENU,
'| TAKE 3',
'| STEM MENU.'
IF TAFRIT.0<=NXTMENU+3 THEN R_MORE_ATTR='C03C41004200'X
                       ELSE R_MORE_ATTR='C03041F242F2'X
IF NXTMENU<1 THEN L_MORE_ATTR='C03C41004200'X
             ELSE L_MORE_ATTR='C03041F242F2'X
FILL_BLOCK1:                   /* GET DATA FOR FIRST MENU */
'PIPE (ENDCHAR ?) < MENU XLINES * | DROP 4 ',
'| LOCATE (1.8) /'MENU.1'/',
'| A: FANOUT ',
'| TAKE 1',
'| SPECS 20.18 1',
'| VAR BLOCK1_TITLE',
'? A: | DROP FIRST',
'| DROP LAST',
'| SPECS 10-* 1',
'| STEM MENU1.'
'PIPE < MENU XAUTH * | DROP 4',
'| LOCATE (1.8) /'USER_NAME'/',
'| LOCATE (10.8) /'MENU.1'/',
'| SPECS 19-* 1 ',
'| SPLIT AT /!/',
'| STEM MENU1_AUTH.'
L=1
DO K=1 TO MENU1.0
```

```
  IF STRIP(MENU1_AUTH.K)='X' THEN DO
     EXEC1.L=SUBSTR(MENU1.K,1,8)
     DESC1.L=SUBSTR(MENU1.K,11,19)
     TYPE1.L=SUBSTR(MENU1.K,3Ø,7)
     L=L+1
  END
END
B1_LINE.Ø=L-1
DO U=1 TO 11
 IF SYMBOL('DESC1.U')¬=VAR THEN B1_LINE.U=COPIES(' ',24)
                           ELSE B1_LINE.U=RIGHT(U,3)||'.'||DESC1.U
END
 BLOCK1 =,
'1101922903C02041F242F7'X||'                          '||'1101AB1DFØ'X||,
'1101E22903C02041F242F7'X||' '||,
'1101E42903C01141F442F7'X||'  '||,
'1101E72903C03041ØØ42F7'X||LEFT(STRIP(BLOCK1_TITLE),18)||,
'2841F2'X||' '||'284100'X||'1101FB1DFØ'X||,
'1102322903C02041F242F7'X||B1_LINE.1 ||'11024B1DFØ'X||,
'1102822903C02041F242F7'X||B1_LINE.2 ||'11029B1DFØ'X||,
'1102D22903C02041F242F7'X||B1_LINE.3 ||'1102EB1DFØ'X||,
'1103222903C02041F242F7'X||B1_LINE.4 ||'11033B1DFØ'X||,
'1103722903C02041F242F7'X||B1_LINE.5 ||'11038B1DFØ'X||,
'1103C22903C02041F242F7'X||B1_LINE.6 ||'1103DB1DFØ'X||,
'1104122903C02041F242F7'X||B1_LINE.7 ||'11042B1DFØ'X||,
'1104622903C02041F242F7'X||B1_LINE.8 ||'11047B1DFØ'X||,
'1104B22903C02041F242F7'X||B1_LINE.9 ||'1104CB1DFØ'X||,
'1105Ø22903C02041F242F7'X||B1_LINE.1Ø||'11051B1DFØ'X||,
'1105522903C02041F242F7'X||B1_LINE.11||'11056B1DFØ'X
FILL_BLOCK2:                    /* GET DATA FOR SECOND MENU */
IF TAFRIT.Ø>1 THEN DO
'PIPE (ENDCHAR ?) < MENU XLINES * | DROP 4 ',
'| LOCATE (1.8) /'MENU.2'/',
'| A: FANOUT ',
'| TAKE 1',
'| SPECS 2Ø.18 1',
'| VAR BLOCK2_TITLE',
'? A: | DROP FIRST',
'| DROP LAST',
'| SPECS 1Ø-* 1',
'| STEM MENU2.I.'
'PIPE < MENU XAUTH * | DROP 4',
'| LOCATE (1.8) /'USER_NAME'/',
'| LOCATE (1Ø.8) /'MENU.2'/',
'| SPECS 19-* 1 ',
'| SPLIT AT /!/',
'| STEM MENU2_AUTH.'
L=1
DO K=1 TO MENU2.I.Ø
  IF STRIP(MENU2_AUTH.K)='X' THEN DO
```

```
     EXEC2.L=SUBSTR(MENU2.I.K,1,8)
     DESC2.L=SUBSTR(MENU2.I.K,11,19)
     TYPE2.L=SUBSTR(MENU2.I.K,30,7)
     L=L+1
  END
END
B2_LINE.0=L-1
DO U=1 TO 11
 IF SYMBOL('DESC2.U')¬=VAR THEN B2_LINE.U=COPIES(' ',24)
                           ELSE B2_LINE.U=RIGHT(U,3)||'.'||DESC2.U
END
 BLOCK2 =,
'11024B2903C02041F242F6'X||'                          '||'1102641DF0'X||,
'11029B2903C02041F242F6'X||' '||,
'11029D2903C01141F442F6'X||'  '||,
'1102A02903C03041004 2F6'X||LEFT(STRIP(BLOCK2_TITLE),18)||,
'2841F2'X||' '||'284100'X||'1102B41DF0'X||,
'1102EB2903C02041F242F6'X||B2_LINE.1 ||'1103041DF0'X||,
'11033B2903C02041F242F6'X||B2_LINE.2 ||'1103541DF0'X||,
'11038B2903C02041F242F6'X||B2_LINE.3 ||'1103A41DF0'X||,
'1103DB2903C02041F242F6'X||B2_LINE.4 ||'1103F41DF0'X||,
'11042B2903C02041F242F6'X||B2_LINE.5 ||'1104441DF0'X||,
'11047B2903C02041F242F6'X||B2_LINE.6 ||'1104941DF0'X||,
'1104CB2903C02041F242F6'X||B2_LINE.7 ||'1104E41DF0'X||,
'11051B2903C02041F242F6'X||B2_LINE.8 ||'1105341DF0'X||,
'11056B2903C02041F242F6'X||B2_LINE.9 ||'1105841DF0'X||,
'1105BB2903C02041F242F6'X||B2_LINE.10||' '||'1105D41DF0'X||,
'11060B2903C02041F242F6'X||B2_LINE.11||'1106241DF0'X
 END /* END TAFRIT.0>1 */
 ELSE BLOCK2=''
FILL_BLOCK3:                    /* GET DATA FOR THIRD MENU */
IF TAFRIT.0>2 THEN DO
'PIPE (ENDCHAR ?) < MENU XLINES * | DROP 4 ',
'| LOCATE (1.8) /'MENU.3'/',
'| A: FANOUT ',
'| TAKE 1',
'| SPECS 20.18 1',
'| VAR BLOCK3_TITLE',
'? A: | DROP FIRST',
'| DROP LAST',
'| SPECS 10-* 1',
'| STEM MENU3.I.'
'PIPE < MENU XAUTH * | DROP 4',
'| LOCATE (1.8) /'USER_NAME'/',
'| LOCATE (10.8) /'MENU.3'/',
'| SPECS 19-* 1 ',
'| SPLIT AT /!/',
'| STEM MENU3_AUTH.'
L=1
DO K=1 TO MENU3.I.0
```

```
  IF STRIP(MENU3_AUTH.K)='X' THEN DO
     EXEC3.L=SUBSTR(MENU3.I.K,1,8)
     DESC3.L=SUBSTR(MENU3.I.K,11,19)
     TYPE3.L=SUBSTR(MENU3.I.K,30,7)
     L=L+1
  END
END
B3_LINE.0=L-1
DO U=1 TO 11
 IF SYMBOL('DESC3.U')¬=VAR THEN B3_LINE.U=COPIES(' ',24)
                           ELSE B3_LINE.U=RIGHT(U,3)||'.'||DESC3.U
END
 BLOCK3 =,
'1103042903C02041F242F3'X||'                          '||'11031D1DF0'X||,
'1103542903C02041F242F3'X||' '||,
'1103562903C01141F442F3'X||'  '||,
'1103592903C030410042F3'X||LEFT(STRIP(BLOCK3_TITLE),18)||,
'2841F2'X||' '||'284100'X||'11036D1DF0'X||,
'1103A42903C02041F242F3'X||B3_LINE.1 ||'1103BD1DF0'X||,
'1103F42903C02041F242F3'X||B3_LINE.2 ||'11040D1DF0'X||,
'1104442903C02041F242F3'X||B3_LINE.3 ||'11045D1DF0'X||,
'1104942903C02041F242F3'X||B3_LINE.4 ||'1104AD1DF0'X||,
'1104E42903C02041F242F3'X||B3_LINE.5 ||'1104FD1DF0'X||,
'1105342903C02041F242F3'X||B3_LINE.6 ||'11054D1DF0'X||,
'1105842903C02041F242F3'X||B3_LINE.7 ||'11059D1DF0'X||,
'1105D42903C02041F242F3'X||B3_LINE.8 ||'1105ED1DF0'X||,
'1106242903C02041F242F3'X||B3_LINE.9 ||'11063D1DF0'X||,
'1106742903C02041F242F3'X||B3_LINE.10||'11068D1DF0'X||,
'1106C42903C02041F242F3'X||B3_LINE.11||'1106DD1DF0'X
END /* END TAFRIT.0>2 */
ELSE BLOCK3=''
 MENU :
 SCREEN='8003'X||,
'1100002903C020410042F6'X||,
'-
═══════════════════════════════════════════════════════════════════════════-
'||,
'1100501DF0'X||,
'1100502903C020410042F6'X||'|'||'1100521DF0'X||,
'1100692903C02041F242F7'X||MENU_TITLE||'1100851DF0'X||,
'11009E2903C020410042F6'X||'|'||'1100A01DF0'X||,
'1100A02903C020410042F6'X||,
'|═══════════════════════════════════════════════════════════════════════'||,
'1100E01DF0'X||,
'1100ED2903C020410042F6'X||'=|'||'1100F01DF0'X||,
'1100912903C020410042F6'X||LEFT(DATE(W),9)||,
'1100E12903C020410042F6'X||DATE()||,
'1101312903C020410042F6'X||TIME()||,
'1100F02903C020410042F6'X||'|'||'1100F21DF0'X||,
'1101092903C02041F242F5'X||TITLE||'1101251DF0'X||,
```

```
'11013E2903C020410042F6'X||'|'||'1101401DF0'X||,
'1101402903C020410042F6'X||,
'|═══════════════════════════════════════════════════════════════════════════════|'||,
'1101901DF0'X||,
'1101902903C020410042F6'X||'|'||'1101921DF0'X||'1101DE2903C020410042F6'X||,
'|'||'1101E01DF0'X||,
'1101D32903'X||R_MORE_ATTR||'PF11>>>'||'1101DB1DF0'X||,
'1101E02903C020410042F6'X||'|'||'1101E21DF0'X||'11022E2903C020410042F6'X||,
'|'||'1102301DF0'X||,
'1102302903C020410042F6'X||'|'||'1102321DF0'X||'11027E2903C020410042F6'X||,
'|'||'1102801DF0'X||,
'1102802903C020410042F6'X||'|'||'1102821DF0'X||'1102CE2903C020410042F6'X||,
'|'||'1102D01DF0'X||,
'1102D02903C020410042F6'X||'|'||'1102D21DF0'X||'11031E2903C020410042F6'X||,
'|'||'1103201DF0'X||,
'1103202903C020410042F6'X||'|'||'1103221DF0'X||'11036E2903C020410042F6'X||,
'|'||'1103701DF0'X||,
'1103702903C020410042F6'X||'|'||'1103721DF0'X||'1103BE2903C020410042F6'X||,
'|'||'1103C01DF0'X||,
'1103C02903C020410042F6'X||'|'||'1103C21DF0'X||'11040E2903C020410042F6'X||,
'|'||'1104101DF0'X||,
'1104102903C020410042F6'X||'|'||'1104121DF0'X||'11045E2903C020410042F6'X||,
'|'||'1104601DF0'X||,
'1104602903C020410042F6'X||'|'||'1104621DF0'X||'1104AE2903C020410042F6'X||,
'|'||'1104B01DF0'X||,
'1104B02903C020410042F6'X||'|'||'1104B21DF0'X||'1104FE2903C020410042F6'X||,
'|'||'1105001DF0'X||,
'1105002903C020410042F6'X||'|'||'1105021DF0'X||'11054E2903C020410042F6'X||,
'|'||'1105501DF0'X||,
'1105502903C020410042F6'X||'|'||'1105521DF0'X||'11059E2903C020410042F6'X||,
'|'||'1105A01DF0'X||,
'1105A02903C020410042F6'X||'|'||'1105A21DF0'X||'1105EE2903C020410042F6'X||,
'|'||'1105F01DF0'X||,
'1105F02903C020410042F6'X||'|'||'1105F21DF0'X||'11063E2903C020410042F6'X||,
'|'||'1106401DF0'X||,
'1106402903C020410042F6'X||'|'||'1106421DF0'X||'11068E2903C020410042F6'X||,
'|'||'1106901DF0'X||,
'1106902903C020410042F6'X||'|'||'1106921DF0'X||,
'1106DE2903C020410042F6'X||'|'||'1106E01DF0'X||,
'1106E02903C020410042F6'X||'|'||'1106E21DF0'X||,
'1106E42903'X||L_MORE_ATTR||'<<<PF10'||'1106EC1DF0'X||,
'1106ED2903C02041F142F2'X||ERROR_MESSAGE||'11072D1DF0'X||,
'11072E2903C020410042F6'X||'|'||'1107301DF0'X||,
'1107302903C020410042F6'X||,
'-
═══════════════════════════════════════════════════════════════════════════════-
'||,
BLOCK1||BLOCK2||BLOCK3||,
'1101E513'X
'PIPE (ENDCHAR ?) VAR SCREEN | FULLSCREEN ',
```

```
'| SPLIT AT ANYOF /'"11"X'/',
'| A: FANOUT',
'| TAKE FIRST',
'| SPECS /$/ 1 1-1 C2X NEXT',
'| VAR AIDKEY',
'? A:',
'| DROP FIRST',
'| SPECS 3-* STRIP 1',
'| STEM COM.'
ERROR_MESSAGE=''
SELECT                                  /* CHECK AID KEY PRESSED */
 WHEN VALUE(AIDKEY)='PF12',
    | VALUE(AIDKEY)='PF24' THEN EXIT     /* EXIT WITH PF12/24     */
 WHEN VALUE(AIDKEY)='CLEAR' THEN DO      /* RESET                 */
   SIGNAL MENU_START
 END /* END CLEAR - RESET */
 WHEN VALUE(AIDKEY)='PF11' THEN DO       /* SHIFT RIGHT SUB MENU  */
   IF NXTMENU+3<TAFRIT.Ø THEN NXTMENU=NXTMENU+1
                         ELSE DO
                       'XMITMSG 2 (APPLID TAF CALLER MEN NOCOMP VAR'
                        ERROR_MESSAGE=MESSAGE.1
                        SIGNAL MENU
                           END
   DROP DESC1.;DROP DESC2.;DROP DESC3.
   SIGNAL PF1ØPF11
 END /* END 'PF11' PRESSED */
 WHEN VALUE(AIDKEY)='PF1Ø' THEN DO       /* SHIFT LEFT SUB MENU   */
   IF NXTMENU>Ø THEN NXTMENU=NXTMENU-1
                         ELSE DO
                       'XMITMSG 3 (APPLID TAF CALLER MEN NOCOMP VAR'
                        ERROR_MESSAGE=MESSAGE.1
                        SIGNAL MENU
                           END
   DROP DESC1.;DROP DESC2.;DROP DESC3.
   SIGNAL PF1ØPF11
 END /* END 'PF1Ø' PRESSED */
 WHEN VALUE(AIDKEY)='ENTER' THEN DO          /* PROCESS ENTER         */
  IF COM.1¬='' THEN DO
    IF COM.1>B1_LINE.Ø THEN DO
     'XMITMSG 5  BLOCK1_TITLE COM.1  (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END
                         ELSE DO             /* PROCESS COMMAND       */
     CALL PROCESS_CMD VALUE('TYPE1.'COM.1) VALUE('EXEC1.'COM.1) COM.1
     IF RESULT=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END /* END PROCESS COMMAND */
   END /* END COM.1 NOT NULL */
```

```
  IF COM.2¬='' THEN DO
    IF COM.2>B2_LINE.Ø THEN DO
     'XMITMSG 5  BLOCK2_TITLE COM.2  (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END
                       ELSE DO               /* PROCESS COMMAND       */
     CALL PROCESS_CMD VALUE('TYPE2.'COM.2) VALUE('EXEC2.'COM.2) COM.2
     IF RESULT=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END /* END PROCESS COMMAND */
   END /* END COM.2 NOT NULL */
  IF COM.3¬='' THEN DO
    IF COM.3>B3_LINE.Ø THEN DO
     'XMITMSG 5  BLOCK3_TITLE COM.3  (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END
                       ELSE DO               /* PROCESS COMMAND       */
     CALL PROCESS_CMD VALUE('TYPE3.'COM.3) VALUE('EXEC3.'COM.3) COM.3
     IF RESULT=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
     ERROR_MESSAGE=MESSAGE.1
     SIGNAL MENU
     END /* END PROCESS COMMAND */
   END /* END COM.3 NOT NULL */
 'XMITMSG 4 (APPLID TAF CALLER MEN NOCOMP VAR'
  ERROR_MESSAGE=MESSAGE.1
 SIGNAL MENU
 END /* END ENTER */
 OTHERWISE DO
  'XMITMSG 1 'VALUE(AIDKEY)' (APPLID TAF CALLER MEN NOCOMP VAR'
  ERROR_MESSAGE=MESSAGE.1
  SIGNAL MENU
 END /* END OTHERWISE - INVALID KEY PRESSED */
END /* SELECT AIDKEY */
EXIT
PROCESS_CMD : PROCEDURE EXPOSE MESSAGE.1
ARG TYPE EXEC_NAME OPTION
'SET CMSTYPE HT'
 SELECT
   WHEN TYPE='DELETED' THEN DO
  'XMITMSG 6 (APPLID TAF CALLER MEN NOCOMP VAR'
    END
   WHEN TYPE='EXEC' THEN DO
   ADDRESS CMS EXEC_NAME
   IF RC=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
       ELSE 'XMITMSG 7 TYPE RC OPTION (APPLID TAF CALLER MEN NOCOMP VAR'
    END
   WHEN TYPE='REXX' THEN DO
```

```
   INTERPRET   EXEC_NAME
   IF RC=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
       ELSE 'XMITMSG 7 TYPE RC OPTION (APPLID TAF CALLER MEN NOCOMP VAR'
    END
   WHEN TYPE='MENU' THEN DO
   MENUCOM='ADDRESS COMMAND EXEC MENU 'EXEC_NAME
   INTERPRET MENUCOM
   IF RC=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
       ELSE 'XMITMSG 7 TYPE RC OPTION (APPLID TAF CALLER MEN NOCOMP VAR'
    END
   WHEN TYPE='MENPROC' THEN DO
   'MAKEXEC 'EXEC_NAME
   IF RC¬=Ø THEN DO
   'XMITMSG 9 EXEC_NAME RC (APPLID TAF CALLER MEN NOCOMP VAR'
   RETURN
   END
   MENPROC_COM='ADDRESS COMMAND EXEC 'EXEC_NAME
   INTERPRET MENPROC_COM
   IF RC=Ø THEN 'XMITMSG Ø (APPLID TAF CALLER MEN NOCOMP VAR'
       ELSE 'XMITMSG 7 TYPE RC OPTION (APPLID TAF CALLER MEN NOCOMP VAR'
    END
  OTHERWISE DO
  'XMITMSG 7 (APPLID TAF CALLER MEN NOCOMP VAR'
   END
 END /* END SELECT COMMAND TYPE */
'SET CMSTYPE RT'
RETURN
/* END OF PROCESS_CMD PROCEDURE */
/* SOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOFSOF */
```

MENU XAUTH

```
==============AUTHORIZATIONS=========================
                |1|2|3|4|5|6|7|8|9|1|1|1|1|1|1|1|1|1|2|2|2|2|2|2|
                | | | | | | | | | |Ø|1|2|3|4|5|6|7|8|9|Ø|1|2|3|4|5|6|
=USERID===MENU=============================================================
MENUADMN|CMSUTIL |X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MENUADMN|SYSCENT |X|X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MENUADMN|TLMENU  |X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MENUADMN|MLTPRT  |X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MENUADMN|FOCUS-1 |X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MENUADMN|FOCUS-2 |X|X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MAINT   |CMSUTIL |X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MAINT   |SYSCENT |X|X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
MAINT   |MLTPRT  |X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
FOCUSER |FOCUS-1 |X| |X| |X| | | | | | | | | | | | | | | | | | |
FOCUSER |FOCUS-2 | |X|X|X| |X| | | | | | | | | | | | | | | | | | |
FOCUSER |CMSUTIL |X|X|X| | |X| | | | | | | | | | | | | | | | | | |
INFOCENT|FOCUS-1 |X|X| | |X| | | | | | | | | | | | | | | | | | |
INFOCENT|FOCUS-2 |X| | | | | |X| | | | | | | | | | | | | | | | | | |
```

```
INFOCENT|CMSUTIL |X| | | | |X| | | | | | | | | | | | | | | | | | | |
TESTER  |FOCUS-1 | | |X|X|X| | | | | | | | | | | | | | | | | | | |
TEMPUS  |TLMENU  |X|X|X|X|X|X|X| | | | | | | | | | | | | | | | | | |
```

MENU XLINES

```
OPTION===EXEC======DESCRIPTION========TYPE========
         1         2         3         4         5
123456789Ø123456789Ø123456789Ø123456789Ø123456789Ø
OPTION===EXEC======DESCRIPTION========TYPE========
XLINEHLP TITLE     First line is menu title
XLINEHLP FUNCTION  this is the name of this menu line
XLINEHLP           DESCRIPTION        this is the menu line description
XLINEHLP                              TYPE    this is the menu line type
XLINEHLP ***END*** this is the end of menu dfinition
CMSUTIL  TITLE     CMS UTILITIES      TITLE
CMSUTIL  FILELIST  A-DISK FILE LIST   EXEC
CMSUTIL  RDRL      READER QUEUE LIST  EXEC
CMSUTIL  SENDFILE  SEND FILE          EXEC
CMSUTIL  DITTO     DITTO              EXEC
CMSUTIL  INTM      CMS SUBSET         EXEC
CMSUTIL  GOOUT     EXIT & LOGOUT      EXEC
CMSUTIL  ***END***
SYSCENT  TITLE     SYSTEM CENTER      TITLE
SYSCENT  VMBATCH   VM-BATCH           EXEC
SYSCENT  VMBACKUP  VM-BACKUP          EXEC
SYSCENT  VMOPER    VM-OPERATOR        EXEC
SYSCENT  EXPLORE   PERFORMANCE MON.   EXEC
SYSCENT  TEMPUS    TEMPUS LINK MENU   MENU
SYSCENT  INTM      CMS SUBSET         EXEC
SYSCENT  GOOUT     EXIT & LOGOUT      EXEC
SYSCENT  ***END***
FOCUS-1  TITLE     FOCUS APPLS        TITLE
FOCUS-1  REPSEX    FOCUS - REPORTS    MENPROC
FOCUS-1  EPICE     FOCUS - EPIC       MENPROC
FOCUS-1  TESTE     FOCUS - TEST       MENPROC
FOCUS-1  STAGX     FOCUS - STAGX      MENPROC
FOCUS-1  PRODE     FOCUS - PROD       MENPROC
FOCUS-1  ***END***
FOCUS-2  TITLE     INFO CENT APPLS    TITLE
FOCUS-2  EIS       FOCUS-EIS          EXEC
FOCUS-2  MATEREP   MATE REPORTS       EXEC
FOCUS-2  TVIOTREP  TVIOT REPORTS      EXEC
FOCUS-2  FORM3     FORM3 REPORTS      MENPROC
FOCUS-2  ALPH      ALPHONES           MENPROC
FOCUS-2  SUPER     SUPER-USER         MENPROC
FOCUS-2  TEMPUS    TEMPUS LINK MENU   MENU
FOCUS-2  ***END***
TLMENU   TITLE     TEMPUS-LINK MENU   TITLE
TLMENU   TLSAMPL   TLSAMPL LIBRARY    EXEC
```

```
TLMENU   TLFOCUS   TLFOCUS LIBRARY    EXEC
TLMENU   TLPOOL    TLPOOL  LIBRARY    EXEC
TLMENU   TLEXTR    EXTRACT TL-DATA    EXEC
TLMENU   TLSTORE   STORE TL-DATA      EXEC
TLMENU   TLINIT    INIT TL-LIBRARY    EXEC
TLMENU   EXIT      PREVIOUS MENU      REXX
TLMENU   ***END***
MLTPRT   TITLE     MULTIPRINT/VM      TITLE
MLTPRT   YPRT12    ACTIVE PRINTERS    EXEC
MLTPRT   YPRT11    PRINTERS           EXEC
MLTPRT   YPRT3     PRINT QUEUES       EXEC
MLTPRT   YPRT4     CONSOLE            EXEC
MLTPRT   MPROPER   MULTIPRINT         EXEC
MLTPRT   GOOUT     EXIT & LOGOUT      EXEC
MLTPRT   ***END***
```

MENU XSTATMS

```
================================================
         1         2         3         4
12345678901234567890123456789012345678901234567890
MENPROC==STATMS===PARMS====T_D==O_D==MD=LPSW====
Hxstatms OWNER    This is the procedure definition header
Hxstatms LINK     This is the definition  for CP LINK generation
Hxstatms LINK     Diskowner Disk_addr Virt_addr Mode Link_Password
Hxstatms LINK
Hxstatms ACCESS   This is the definition fo CMS ACCESS generation
Hxstatms ACCESS   Virt_addr Access_mode
Hxstatms ACCESS
Hxstatms ACCESS
Hxstatms TDISK    Temporary Disk specification: Y/N/Cyl_Number
Hxstatms EX       Focus program to execute
SABABA   OWNER    PROC_OWNER
SABABA   LINK     MAINT    0191 0001 RR E3D5C9C1D4D9
SABABA   LINK     LOGIC    0191 0002 RR C3C9C7D6D3D9
SABABA   LINK     RSCS     0401 0101 RR D3D3C1
SABABA   ACCESS   0001 B
SABABA   ACCESS   0002 C
SABABA   ACCESS   0100 X
SABABA   ACCESS   0101 L
SABABA   TDISK    Y
SABABA   EX       STAM
REPSEX   OWNER    PROC_OWNER
REPSEX   LINK     FOCUS    0065 0065 RR D3D3C1
REPSEX   LINK     FOCUS    0223 0223 RR E2E4C3D6C6D9
REPSEX   LINK     FOCUS    0401 0401 RR E2E4C3D6C6D9
REPSEX   LINK     FOCUS    0405 0405 RR E2E4C3D6C6D9
REPSEX   LINK     REPUSER  0401 0191 RW E2D7C5D9E6
REPSEX   ACCESS   0065 F
REPSEX   ACCESS   0223 L/A
```

```
REPSEX   ACCESS   Ø4Ø1 M/A
REPSEX   ACCESS   Ø4Ø5 N/A
REPSEX   TDISK    Y
REPSEX   EX       REPORTS
EPICE    OWNER    PROC_OWNER
EPICE    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
EPICE    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
EPICE    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
EPICE    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
EPICE    LINK     EPICDB   Ø192 Ø192 RR D3D3C1
EPICE    ACCESS   ØØ65 F
EPICE    ACCESS   Ø223 L/A
EPICE    ACCESS   Ø4Ø1 M/A
EPICE    ACCESS   Ø4Ø5 N/A
EPICE    ACCESS   Ø192 B
EPICE    TDISK     Y
EPICE    EX       FEPIC
TESTE    OWNER    PROC_OWNER
TESTE    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
TESTE    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
TESTE    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
TESTE    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
TESTE    LINK     FOCUSTST Ø4Ø1 Ø3Ø1 RR D3D3C1
TESTE    LINK     FOCUSTST Ø4Ø2 Ø3Ø2 RR D3D3C1
TESTE    ACCESS   ØØ65 F
TESTE    ACCESS   Ø223 L/A
TESTE    ACCESS   Ø4Ø1 M/A
TESTE    ACCESS   Ø4Ø5 N/A
TESTE    ACCESS   Ø3Ø1 B
TESTE    ACCESS   Ø3Ø2 C
TESTE    TDISK     Y
TESTE    EX       FTEST
STAGX    OWNER    PROC_OWNER
STAGX    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
STAGX    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
STAGX    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
STAGX    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
STAGX    LINK     FOCUSTAG Ø4Ø1 Ø3Ø1 RR D3D3C1
STAGX    LINK     FOCUSTAG Ø4Ø2 Ø3Ø2 RR D3D3C1
STAGX    ACCESS   ØØ65 F
STAGX    ACCESS   Ø223 L/A
STAGX    ACCESS   Ø4Ø1 M/A
STAGX    ACCESS   Ø4Ø5 N/A
STAGX    ACCESS   Ø3Ø1 B
STAGX    ACCESS   Ø3Ø2 C
STAGX    TDISK     Y
STAGX    EX       GFOC
PRODE    OWNER    PROC_OWNER
PRODE    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
PRODE    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
PRODE    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
```

```
PRODE    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
PRODE    LINK     FOCUSPRD Ø4Ø1 Ø3Ø1 RR D3D3C1
PRODE    LINK     FOCUSPRD Ø4Ø2 Ø3Ø2 RR D3D3C1
PRODE    ACCESS   ØØ65 F
PRODE    ACCESS   Ø223 L/A
PRODE    ACCESS   Ø4Ø1 M/A
PRODE    ACCESS   Ø4Ø5 N/A
PRODE    ACCESS   Ø3Ø1 B
PRODE    ACCESS   Ø3Ø2 C
PRODE    TDISK    3Ø
PRODE    EX       PFOC
FORM3    OWNER    PROC_OWNER
FORM3    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
FORM3    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø4Ø1 Ø4Ø1 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø4Ø5 Ø4Ø5 RR E2E4C3D6C6D9
FORM3    LINK     FOCUS    Ø5ØØ Ø4Ø2 RR E2E4C3D6C6D9
FORM3    ACCESS   ØØ65 F
FORM3    ACCESS   Ø223 L/A
FORM3    ACCESS   Ø4Ø1 M/A
FORM3    ACCESS   Ø4Ø5 N/A
FORM3    ACCESS   Ø4Ø2 O/A
FORM3    TDISK    3
FORM3    EX       FORMS
ALPH     OWNER    PROC_OWNER
ALPH     LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
ALPH     LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
ALPH     LINK     FOCUS    Ø224 Ø224 RR E2E4C3D6C6D9
ALPH     LINK     FOCUS    Ø225 Ø225 RR E2E4C3D6C6D9
ALPH     LINK     TESTDBS  Ø7ØØ Ø7ØØ RR E2C2C4D9
ALPH     ACCESS   ØØ65 F
ALPH     ACCESS   Ø223 L/A
ALPH     ACCESS   Ø224 M/A
ALPH     ACCESS   Ø225 N/A
ALPH     ACCESS   Ø7ØØ E
ALPH     TDISK    3Ø
ALPH     EX       YELOPAG
SUPER    OWNER    PROC_OWNER
SUPER    LINK     FOCUS    ØØ65 ØØ65 RR D3D3C1
SUPER    LINK     FOCUS    Ø223 Ø223 RR E2E4C3D6C6D9
SUPER    LINK     FOCUS    Ø224 Ø224 RR E2E4C3D6C6D9
SUPER    LINK     FOCUS    Ø225 Ø225 RR E2E4C3D6C6D9
SUPER    LINK     FOCSUPER Ø191 ØØØ5 RR D9C5D7E4E2D9
SUPER    ACCESS   ØØ65 F
SUPER    ACCESS   Ø223 L/A
SUPER    ACCESS   Ø224 M/A
SUPER    ACCESS   Ø225 N/A
SUPER    ACCESS   ØØØ5 B
SUPER    TDISK    22
SUPER    EX       FOCSUPER
```

GOOUT EXEC

```
/* GOOUT EXEC : CREATED BY XPANEL PRE-PROCESSOR */
$F1='PF01'; $F2='PF02'; $F3='PF03'; $F4='PF04'; $F5='PF05'; $F6='PF06'
$F7='PF07'; $F8='PF08'; $F9='PF09'; $7A='PF10'; $7B='PF11'; $7C='PF12'
$C1='PF13'; $C2='PF14'; $C3='PF15'; $C4='PF16'; $C5='PF17'; $C6='PF18'
$C7='PF19'; $C8='PF20'; $C9='PF21'; $4A='PF22'; $4B='PF23'; $4C='PF24'
$01='PA1'; $6E='PA2'; $7D='ENTER'; $6D='CLEAR'
ERROR_MESSAGE=''
 GOOUT : ;  SCREEN='8003'X||'1101A72903C02041F442F7'X||,
'                                  '||'1101CA1DF0'X||,
'1101F72903C020410042F7'X||'|'||'1101F91DF0'X||,
'1101FF2903C02041F142F4'X||ERROR_MESSAGE||'1102121DF0'X||,
'1102182903C020410042F7'X||'|'||'11021A1DF0'X||,
'1102472903C020410042F7'X||'|'||'1102491DF0'X||,
'1102682903C020410042F7'X||'|'||'11026A1DF0'X||,
'1102972903C020410042F7'X||'|'||'1102991DF0'X||,
'11029F2903C020410042F3'X||'CONFIRM EXIT (Y/N)'||'1102B21DF0'X||,
'1102B82903C020410042F7'X||'|'||'1102BA1DF0'X||,
'1102E72903C020410042F7'X||'|'||'1102E91DF0'X||,
'1103082903C020410042F7'X||'|'||'11030A1DF0'X||,
'1103372903C020410042F7'X||'|'||'1103391DF0'X||,
'1103472903C00141F242F3'X||' '||'1103491DF0'X||,
'1103582903C020410042F7'X||'|'||'11035A1DF0'X||,
'1103872903C020410042F7'X||'|'||'1103891DF0'X||,
'1103A82903C020410042F7'X||'|'||'1103AA1DF0'X||,
'1103D72903C020410042F7'X||'|'||'1103D91DF0'X||,
'1103F82903C020410042F7'X||'|'||'1103FA1DF0'X||,
'1104272903C020410042F7'X||'|'||'1104291DF0'X||,
'1104482903C020410042F7'X||'|'||'11044A1DF0'X||,
'1104772903C02041F442F7'X||'|                                  |'||,
'11049A1DF0'X||'11034813'X
'VMFCLEAR'
'PIPE (ENDCHAR ?) VAR SCREEN | FULLSCREEN ',
'| SPLIT AT ANYOF /'"11"X'/',
'| A: FANOUT',
'| TAKE FIRST',
'| SPECS /$/ 1 1-1 C2X NEXT',
'| VAR AIDKEY',
'? A:',
'| DROP FIRST',
'| SPECS 3-* 1',
'| VAR RESP'
SELECT
WHEN RESP='Y' | RESP='y' THEN 'CP LOGOFF'
WHEN RESP='N' | RESP='n' THEN EXIT
OTHERWISE DO
      ERROR_MESSAGE=CENTER(RESP' IS NOT VALID',18)
      SIGNAL GOOUT
      END
END
RETURN
```

INTM EXEC

```
/* INTM     EXEC    */
/* THIS IS USED AS A CMS SUBSET IN OUR MENUS */
SIGNAL ON NOVALUE
TRACE OFF
ADDRESS COMMAND
'NUCEXT WAKEUP'
IF RC/=Ø
   THEN 'STATE WAKEUP MODULE *'
WUTHERE=RC=Ø

DO FOREVER
   SAY 'CMS SUBSET. ENTER "RETURN", "QUIT" OR "EXIT" TO GO BACK TO YOUR
MENU'  ARG(1)
   DO FOREVER
      'CONWAIT'
      IF EXTERNALS()=Ø & WUTHERE
         THEN
            DO
               'NUCEXT INTCMD1'
               IF RC=1
                  THEN 'WAKEUP ( CONS'
            END
      PARSE PULL S;
      IF S='' | LEFT(S,1)='*'
         THEN LEAVE /* GO SAY WHERE WE ARE */
      PARSE UPPER VAR S A1 A2 A3
      IF FIND('RETURN QUIT EXIT', A1)>Ø & A2=''
         THEN EXIT
      FC=LEFT(S,1)
      SAVD=DIAG('C')
      SELECT
         WHEN POS(FC,':;,.')=Ø
            THEN IF POS('|',S)=Ø
               THEN ADDRESS CMS ''S
               ELSE CALL PIPELINE
         WHEN SUBSTR(S,2,1)=FC | FC=';'
            THEN ADDRESS CMS SUBSTR(S,2)
         WHEN FC='.'
            THEN SUBSTR(S,2)
         WHEN FC=':'
            THEN TRANSLATE(SUBSTR(S,2))
         WHEN FC=','
            THEN 'EXEC' TRANSLATE(SUBSTR(S,2))
      END
      NUD=DIAG('C')
      R = RC
      DO QUEUED()
         PARSE PULL L
```

```
         SAY 'STACK:' STRIP(L, 'T')
      END
      'FINIS * * *'; 'SET CMSTYPE RT'
      PARSE VAR NUD NUDAT+8 NUTIM+8 NUCP1+8 NUCP2+8
      PARSE VAR SAVD            +16 SACP1+8 SACP2+8
      ML=NUTIM ' ' NUDAT ' ' DELTA(NUCP1,SACP1) ' ' DELTA(NUCP2,SACP2),
          ARG(1)
      IF R/=Ø
         THEN SAY 'R('R').  ' ML
         ELSE
            IF A1<>'EXEC' | A2<>'INTINT'
               THEN SAY 'ØØ'X '    OK. ' ML
   END
END

DELTA: PROCEDURE
PARSE ARG A1, A2
IF A1==A2 /* STRING COMPARE DOES ALL OF IT */
   THEN RETURN Ø
NUMERIC DIGITS 15
T1=C2D(A1)
TØ=C2D(A2)
IF T1<TØ
   THEN RETURN '---'
RETURN ((T1-TØ)%1ØØØ)/1ØØØ

PIPELINE:
IF LEFT(STRIP(S),1)='|'
   THEN S='CONSOLE' S
IF RIGHT(STRIP(S),1)='|'
   THEN S=S 'CONSOLE'
'PIPE' S                              /* ISSUE COMMAND               */
RETURN
```

MAKETDSK EXEC

```
/*  MAKE TEMP DISK */
/* CALLED BY MAKEXEC PROCEDURE TO GENERATE THE PROPER TDISK FOR */
/* OUR APPLICATION */
/* LIMITATION : WE DO NOT WANT TDISKS BIGGER THAN 1ØØ/339Ø CYL */
ARG NUMCYL
IF STRIP(NUMCYL)='' THEN RETURN 999
IF DATATYPE(NUMCYL)¬='NUM' THEN RETURN 888
IF NUMCYL>1ØØ THEN RETURN 777
'SET EMSG ON'
'PIPE CP Q V 444 | SPECS WORD1 1 WORD4 15 WORD6 25 | VAR Q444'
PARSE VAR Q444 E_MES TEMP CUR_SIZE
IF STRIP(E_MES)¬='HCPQVDØ4ØE' THEN DO /* VDEV 444 ALREADY HERE */
  IF STRIP(TEMP)='(TEMP)' THEN DO /* TDISK ALREADY EXIST */
```

```
    IF CUR_SIZE>=NUMCYL THEN DO /* TDISK ALREADY EXIST, SIZE IS OK */
     'ACC 444 Z (ERASE'
      RETURN ØØ1
     END
                         ELSE 'CP DET 444'
   END
                           ELSE 'CP DET 444'
 END
                               ELSE 'CP DET 444'
'CP DEF T339Ø 444 'NUMCYL
IF RC¬=Ø THEN RETURN 666
PUSH TMP444
PUSH 1
'FORMAT 444 Z'
RETURN ØØØ
```

MAKEXEC EXEC

```
/*  GENERATE A REXX PROCEDURE FROM DEFINITIONS IN MENU XSTATMS FILE */
ARG OPTION
IF OPTION='' THEN EXIT Ø24
OPTION=LEFT(OPTION,8)
/* GET MY PROCEDURE BODY */
'PIPE (ENDCHAR ?) < MENU XSTATMS * ',
'| LOCATE (1.8) /'OPTION'/',
'| SPECS 1Ø-* 1 ',
'| A: FANOUT',
'| LOCATE (1.8) /LINK    /',
'| SPECS 1Ø-* 1 ',
'| B: FANOUT',
'| SPECS W1 1 | STEM LNKID.',
'? B:| SPECS W2 1 | STEM ORGADDR.',
'? B:| SPECS W3 1 | STEM TRGADDR.',
'? B:| SPECS W4 1 | STEM LNKMODE.',
'? B:| SPECS W5 1 | STEM LNKPSWD.',
'? A:| LOCATE (1.8) /ACCESS  /',
'| SPECS 1Ø-* 1 ',
'| C: FANOUT',
'| SPECS W1 1 | STEM ACCADDR.',
'? C:| SPECS W2 1 | STEM ACCMODE.',
'? A:| LOCATE (1.8) /TDISK   / | SPECS W2 1 | VAR NCYL',
'? A:| LOCATE (1.8) /EX      / | VAR EX_NAME'
IF SYMBOL('EX_NAME')¬='VAR' THEN EXIT Ø28
```

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

Jaakov J Hazan
Technical Support Manager
Ynon Technologies & Computers (Israel)

Display XEDIT ring and select a file

This XEDIT macro displays all the files that are in the current XEDIT ring, allowing one file to be selected and worked on.

The 'XEDIT ring' contains all the files that you are currently editing with XEDIT. That means all files you have called by typing 'XEDIT fn ft fm' from the command line of a file you are editing at the moment. You can normally browse through this 'ring' by retyping X(EDIT) in the command line, which leads you from one file in the ring to the next.

Unfortunately, it is not possible to jump directly to any particular one of these files.

Retyping X can be a tedious task, particularly when your ring has many entries. The QR macro, set to a PF key, can greatly simplify the selection of a file.

In your PROFILE XEDIT, set PF22, for example, with:

```
SET PF22 MACRO QR
```

When you press PF22 while XEDITing a file, you get a dynamic display of all the files in your current ring and you can type the number of the file to be selected.

With PF7 and PF8 you can page backward and forward as usual. PF12 exits the ring display.

Figure 1 shows an example of the XEDIT ring display.

CONFIGURATION AND INTERNALS

The XEDIT ring is determined by an EXTRACT /RING command.

The list of files in the XEDIT ring is built in the temporary file QR$$$$$$ QR$$$$$$ A and is presented to the user by the profile QR1 XEDIT.

QR XEDIT

```
/*********************************************************************/
/* Display XEDIT ring and select a file from it                      */
```

```
                      List of files in XEDIT ring
  ====>
  Please select a line:            (PF7=Backw  PF8=Forw  PF12=Exit)
                     No.  |            File
                    ------+------------------------
                       1   | SYSTEM   SEGID    S2
                       2   | MAINT    NETLOG   AØ
                       3   | USER     DOK      V1
                       4   | QR       EXEC     A1
                       5   | QR1      XEDIT    V1
                       6   | USER     DISKMAP  A1
                       7   | SUB      EXEC     V1
                       8   | TEST     TEST     L1
```

Figure 1: Example XEDIT ring display

```
/*********************************************************************/
trace off
parse upper arg op
if op = '?' then signal hilfe
bb = 'BB'X
'EXTRACT /RING'
'SET CMSTYPE HT'
'ERASE QR$$$$$$ QR$$$$$$ A'
'SET CMSTYPE RT'

zeile = '                    No. ' bb '           File           '
zeile = '                  -------+------------------------   '
'EXECIO 1 DISKW QR$$$$$$ QR$$$$$$ A (VAR ZEILE'
'EXECIO 1 DISKW QR$$$$$$ QR$$$$$$ A (VAR ZEILE'

do i = 2 to ring.Ø
   zeile = '                   ' format(i-1,3)'  ' bb substr(ring.i,1,2Ø)
   'EXECIO 1 DISKW QR$$$$$$ QR$$$$$$ A (VAR ZEILE'
end

'FINIS QR$$$$$$ QR$$$$$$ A'
'XEDIT QR$$$$$$ QR$$$$$$ A (PROFILE QR1'

pull zeile             /* has been stacked by QR1 XEDIT */
                       /* returns the selected line     */
                       /* or Ø for 'exit'               */
```

```
if zeile = Ø then exit
j = zeile + 1
file = substr(ring.j,1,2Ø)
push 'XEDIT' file

exit

/*********************************************************************/
/* Help                                                              */
/*********************************************************************/
hilfe:
'VMFCLEAR'
address cms 'type qr       xedit * 1 Ø3'
```

QR1 XEDIT

```
/*********************************************************************/
/* Auxiliary XEDIT profile for displaying the files of the ring      */
/* and prompting the user's selection                                */
/*********************************************************************/
trace off

'SET NUMBER OFF'
'SET CURLINE ON 4'
'SET CMDLINE TOP'
'SET PF 12 BEFORE Ø'
'EXTRACT /SIZE /PF *'
'SET SCALE OFF'
'SET TOFEOF OFF'
'SET PREFIX OFF'
'SET MSGLINE ON 4'
'SET RESERVED Ø1 WHITE NONE NOHIGH' ,
     center('List of files in XEDIT ring',7Ø)
'SET RESERVED Ø3 WHITE NONE NOHIGH'
'COLOR SCALE   YELLOW UND'
'COLOR TOFEOF  BLUE'
'COLOR CMDLINE WHITE'
'COLOR IDLINE  TUR'
'COLOR ARROW   PINK'
'COLOR SHADOW  BLUE'
'COLOR MSGLINE YELLOW UNDERL'
'COLOR CURLINE GREEN'

schleife:
zeile = '$'
do while datatype(zeile,'W') = Ø | zeile < Ø | zeile > (size.1 - 2)
   'MSG Please select a line:' copies(' ',23),
      '(PF7=Backw  PF8=Forw  PF12=Exit)'
   'READ'
```

```
   parse upper pull zeile
   if zeile = pf7.2 | zeile = pf8.2 then do
       address xedit zeile
       signal schleife
   end
end


'FILE'
push zeile    /* return selected line number or 0 for 'exit' */
exit
```

Dr Reinhard Meyer (Germany)

Moving to RAMAC

Our company DASD were rapidly approaching meltdown. We had a fairly large number of 3380E and 3380K devices, which failed on a regular basis. Indeed, we were becoming a member of the DASD Of The Week club – one club that was not too popular.

The decision was made to replace the 3380s with RAMAC II from IBM. On the first meeting I attended with the vendor, we in Tech Support learned that IBM assumed we were positioned for the arrival of the hardware. It was presumed that our IOCP was in shape and that our operating systems were ready to go. It just so happened that we were upgrading our MVS/ESA 4.2 system and prepping it to move to production. VM/ESA 1.2.2 is our hypervisor here, so research was started to see what we would need to have to be RAMAC friendly. Both VM and MVS required some PTFs, which were ordered and installed upon arrival. In normal cases, PTFs are applied to the test system first and then production. In our case we applied the PTFs to the production VM and the test MVS first. Once satisfied, the VM PTFs were applied to the test VM which was to become production later in the summer.

I had no documentation for the necessary IOCP changes and had to get

them sent in quickly from IBM. We installed an ESCON adapter for the RAMAC which involved some IOCP options that I'd never seen before. A note to the VM Bulletin Board got me an answer very quickly, which was that I had to use the IXP option to account for the descriptions RAMAC needed. We got the IOCDS updated, got MVS updated, made some HCPRIO changes for our V=R MVS guest, and got ready to bring the new DASD on-line.

After coming up on the new IOCDS, VM was trucking along quite happily and MVS was having major heartburn. The odd thing here was that MVS did not have any RAMAC genned into it, nor were any defined in the directory! MVS would run along for a short time and then go into a tight loop, eat up all its memory, and then everything would bomb off with OC1 abends. The call was to backout the IOCDS and figure out what was going on. Our boss recalled that we had a similar problem at disaster recovery with CMF from Boole and Babbage, so we regrouped to try again without bringing up CMF.

Now that we were back on the newer IOCDS we could concentrate on moving our VM users and software from 3380s to RAMAC. We use SSDM from Sterling as our directory manager, and I set up a few new DASD to begin the migration. SSDM has a DASD to DASD copy which made things run much quicker, since we were changing device architecture. There was an added benefit in SSDM that we hadn't seen before. One of our user departments has a large number of very large mini-disks. When SSDM began the migration, it calculated the required size of the target disk based on the occupancy rate of the source disk. A user brought this to my attention when he asked what happened to his 'missing' space.

While this user migration was under way, preparations were being made to finish moving the entire VM system to 3390 DASD. The extra PTFs were added and the merge/move was planned. We wrote an EXEC to compare the production and test directories and list out the duplicate entries. Some were selected because they were newer versions or more recent maintenance had been applied to them. Others, such as VMUTIL and SYBACK, were migrated from the old system to the new to account for all the local options and legacy-type files that went with them. Others, such as PAGE, TDISK, etc, were merged as required.

Attention was then turned to ACF2. All the rule sets from the production system were merged into test to get it ready for the big switch. ACF2 has a feature that allows for two or more system IDs to be present in the Virtual Machine Options (VMO) database. By entering ACF mode and setting the system ID, I could create a second set within the VMO with the 'new' system ID present. After getting all the VMO records in and having the security folks check them out, we changed the SYSID in the CONFIG file for a test IPL. It all checked out and the cut-over date was established.

Our network programmer got his VTAM and RSCS requirements set up for the big day. The Friday before the cut-over we declared a hold on directory adds. On Friday afternoon the directories were merged and a couple of test IPLs were performed, which was a last checkout. We had a few directory kinks to work out, some of which were quite odd. Two of the OfficeVision servers failed the edit on password. I did not find them in the restricted password file, so I just overtyped the password in the source and it worked!

Sunday was cut-over day. The MVS folks had to move the ACF2 log-on ID database to a shared DASD because VM uses it, too. They had a little bit of a problem with their VSAM catalog, but once it was fixed, VM came up on its new DASD after two tries. The first time, ACF2 couldn't get a good hookup to the log-on ID database. I fixed the server's hook to the shared file and tried again. VM came up with no problems and we got the MVS guest up and happy. We did a cursory checkout on getting to PROFS, TSO, and so on. It all looked good and we went home.

Monday was a slightly different story. VTAM and RSCS were not doing as expected. The network programmer spent some time redoing his definitions and after bouncing VTAM a couple of times got it all back together. The greatest thing about the whole transfer was that VM made the switch with no hiccups. We took out a fairly large number of 3380s and now run almost exclusively on RAMAC. The VM resident volume is on a 3390 at present, but is being moved this weekend. The engineer informed me that we lost a DDM last week, but we didn't even notice a slowdown in the system.

Daniel A McLaughlin
Senior Technical Support Analyst (USA)

Expanded mini-disk manager

Expanded mini-disk manager (EMDM) controls a virtual disk, which is built from up to 2,000 virtual DASD devices. EMDM provides the ability to handle very large files without any limitation on their size.

EMDM is written in Assembler and its utilities are written in REXX.

EMDM works in CMS with VM/SP Release 5.

EMDM occupies about 2,500 bytes. In addition, it dynamically allocates less than 46,000 bytes of main memory.

UTILIZATION OF EMDM

EMDM reads and writes physical sequential files on an expanded disk. RECFM, LRECL, and BLKSIZE are not supported. All file records have variable lengths of 0 up to 32,760 bytes. An expanded disk map is shown in Figure 1.

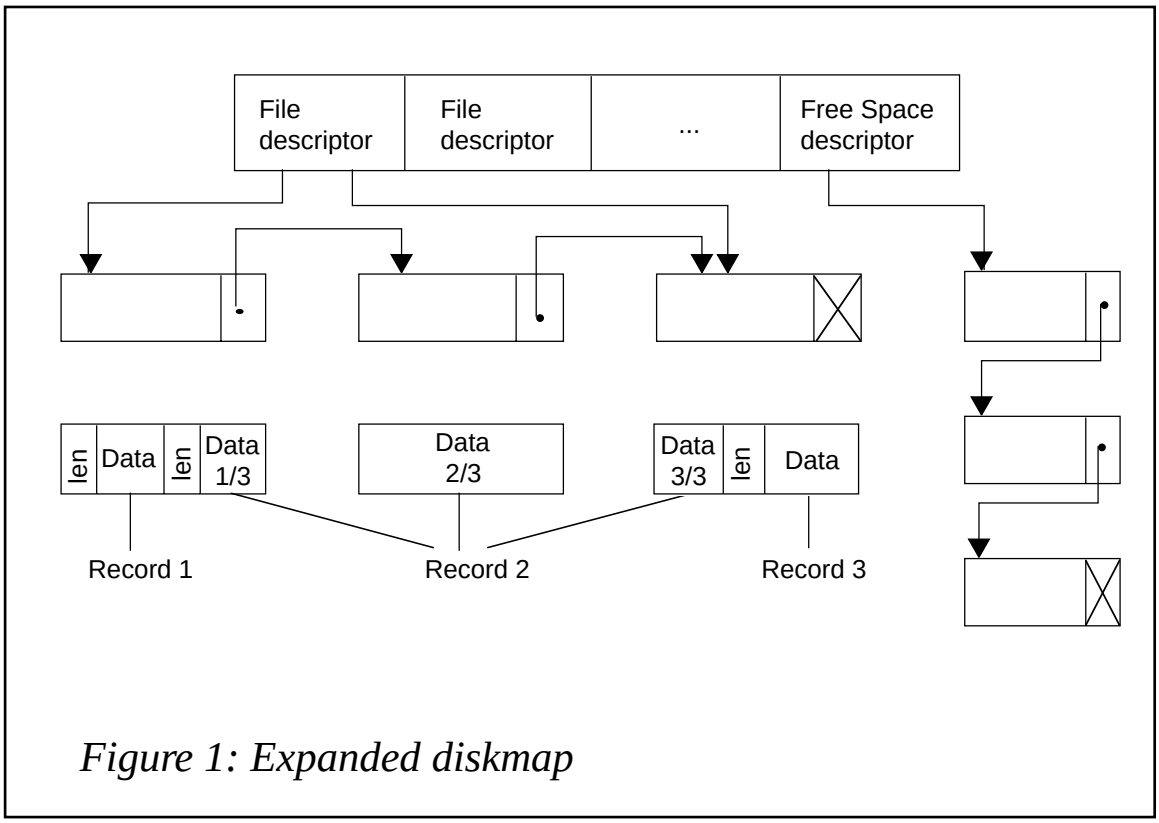

Figure 1: Expanded diskmap

The following parameters are used by application programs to pass a request to EMDM:

- Req – a character specifying the requested function.
- File – a string 16 bytes long containing the filename and filetype of the processed file.
- Record – a string whose length is between 2 and 32,762 bytes – in/out buffer.

In all cases the length of the parameter ‘record’ should be sufficient to take the majority of the processed records. EMDM always supposes that the string ‘record’ may contain records of any length, when transfering data to an application program.

The parameter ‘req’ may have the following values:

- I – open an existing file as the input file.
- O – open the file as an output file. If the file already exists, it is deleted before opening.
- M – open an existing file for modification, and add records to the end of the file.
- D – Delete an existing file.
- R – Read consecutive records and pass to the application program.
- W – Write consecutive records and pass to the application program.
- C – Close the file.
- Q – close all opened files and quit expanded disk processing.

To display files on the expanded disk EMDMLIST EXEC may be used. It has no parameters. An example of EMDMLIST EXEC output is shown in Figure 2.

Examples of EMDM calls from PL/I and Assembler are:

- Assembler:

```
MVC   FILE(8),=CL8'NAME'
MVC   FILE+8(8),=CL8'TYPE'
```

```
Filename Filetype   Record Start Blk/CUU  End Blk/CUU  Free off   Date   Time
TEST1    EMDMTEST      500         2 301      124 301   3950    06/28/96 16:58
TEST2    EMDMTEST     1000       125 301       76 303   1240    06/28/96 16:59
```

Figure 2: A EMDMLIST EXEC output

```
         MVI   REQ,C'O'
         CALL  EMDM,(REQ,FILE,RECORD)
         MVI   REQ,C'W'
         MVC   RECORD(2),=H'5'
         MVC   RECORD+2(5),=CL5'12345'
         CALL  EMDM,(REQ,FILE,RECORD)
         MVI   REQ,C'Q'
         CALL  EMDM,(REQ,FILE,RECORD)

REQ      DS    C
FILE     DS    CL16
RECORD   DS    CL256
```

- PL/I:

```
DCL EMDM ENTRY (CHAR, *, *) OPTIONS (ASM INTER);
DCL 1 FILE AUTO,
      2 FILENAME CHAR (8),
      2 FILETYPE CHAR (8);
DCL RECORD CHAR (32762) AUTO;
DCL 1 RECORD_MAP BASED (ADDR (RECORD)),
      2 DATA_LEN FIXED BIN (15),
      2 DATA CHAR (I REFER (DATA_LEN));

FILENAME = 'NAME';
FILETYPE = 'TYPE';
CALL EMDM ('O', FILE, RECORD);
DATA_LEN = 5;
DATA = '12345';
CALL EMDM ('W', FILE, RECORD);
CALL EMDM ('Q', FILE, RECORD);
```

EXPANDED DISK DEFINITION

EMDM EXEC creates an expanded disk from a number of free virtual DASD devices. Before this happens, this device must be attached to the user virtual machine in the VM system directory.

EMDM EXEC always formats a mini-disk, which is to be added to the expanded disk, if it detects any of the following:

- The physical blocksize of the mini-disk is not 4KB
- The CP/VM mini-disk size is not equal to CMS/VM size.

Otherwise EMDM EXEC asks the user about formatting. To cancel formatting, a user must be sure that all mini-disk records contain binary zeros. This ensures the correct functioning of EMDM.

If a user cancels RESERVE, then mini-disks should not be added to the expanded disk.

The expanded mini-disk description is saved in the file EXPANDED $MDISK$ A. This file cannot be deleted until the expanded disk contains the relevant data. The destruction of this file destroys the expanded disk.

An example of an EMDM EXEC report is shown in Figure 3.

EMDMCHK ASSEMBLE

```
*************************************************************************
****                                                         ***         ****
**** EMDMCHK     checks reserved  mini-disk                  ***         ****
****                                                         ***         ****
*************************************************************************
*                                                                       *
```

```
--- Next mini-disks are added to expanded disk ---

             E01 E02 E03 E04 E05 E06 E07 E08 E09 E10 E11 E12


                   --- Expanded disk definition ---

                  Mini-disks....................12
                     Blocks.................1589424
                      Bytes.............6510280704
                         MB.................6208.69
                         GB...................6.06
```

Figure 3: A EMDM EXEC report

```
EMDMCHK  CSECT
         USING *,12
         LR    11,14
         LA    Ø,EXTPLIST
         LA    1,PLIST
         ICM   1,8,=X'ØØ'
         SVC   2Ø2
         DC    AL4(1)
         LTR   15,15
         BZ    OKCHK
         LA    15,4ØØØ(15)
         ST    15,OFFSET
OKCHK    EQU   *
         LA    Ø,REXXPARM
         LA    1,COMMAND
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
         BR    11
EXTPLIST EQU   *
         DC    A(COMMVERB)
         DC    A(Ø)
         DC    A(Ø)
         DC    A(Ø)
COMMVERB DC    CL8'CMS'
PLIST    DS    ØD
         DC    CL8'DISKID'
         DC    CL8'CHKDISK'
         DS    XL2
         DS    H
OFFSET   DS    F
         DS    D
REXXPARM DC    A(COMMAND)
         DC    A(Ø)
         DC    A(Ø)
         DC    A(REQBLOK)
COMMAND  DC    CL8'EXECCOMM'
REQBLOK  EQU   *
         DC    A(Ø)
         DS    F
         DC    CL1's'
         DC    X'ØØ'
         DC    H'Ø'
         DC    F'Ø'
         DC    A(NAME)
         DC    F'6'
         DC    A(OFFSET)
         DC    F'4'
NAME     DC    CL6'OFFSET'
         END   EMDMCHK
```

EMDM ASSEMBLE

```
*************************************************************************
****                                                    ***         ****
**** EMDM        expanded mini-disk manager            ***         ****
****                                                    ***         ****
*************************************************************************
*                                                                       *
EMDM     CSECT
         SAVE  (14,11)
         BALR  11,Ø
         USING *,11
         ST    13,SA+4
         LA    13,SA
         LR    1Ø,1
         LA    9,IUCVPLST
         USING IPARML,9
         CLI   ISCON,C'Y'
         BE    PROCREQ
         DMSFREE DWORDS=5632,AREA=HIGH
         LR    3,1
         USING FL,7
         LA    1,9
         LA    7,OPNTAB
         LR    8,3
         LR    2,7
         LA    6,1
         STM   2,3,REG23
INIT     EQU   *
         LA    8,X'FFF'(6,8)
         ST    8,5(7)
         LA    7,11(7)
         BCT   1,INIT
         LA    8,X'FFF'(6,8)
         ST    8,CUU
         HNDIUCV SET,NAME=EMDMGR,EXIT=EXTHNDLR
         SR    6,6
CONNECT  EQU   *
         FSREAD 'EXPANDED $MDISK$ A',BSIZE=1Ø,BUFFER=INBUF
         C     15,=F'1'
         BE    NOTEDDEF
         C     15,=F'12'
         BE    CLOSEIN
         LA    8,Ø(6,8)
         MVC   Ø(2,8),INBUF+8
         IUCV  CONNECT,PRMLIST=IUCVPLST,PRMDATA=YES,USERID=BLOCKIO,MF=L
         CMSIUCV CONNECT,NAME=EMDMGR,PRMLIST=IUCVPLST
         WAITECB ECB=ECB,FORMAT=OS
         XC    ECB(4),ECB
         CLI   CHECK,X'FF'
```

```
         BNE   CONFAIL
         LTR   6,6
         BNZ   MODONLY
         MVI   REQ+3,X'Ø2'
         XC    PATH(2),PATH
         MVC   BLOCKNUM(4),=X'ØØØØØØØ1'
         ST    3,BUF
         BAL   5,SIOONLY
MODONLY  EQU   *
         LA    6,2(6)
         B     CONNECT
CLOSEIN  EQU   *
         FSCLOSE 'EXPANDED $MDISK$ A'
         MVI   ISCON,C'Y'
PROCREQ  EQU   *
         LM    2,3,REG23
         LM    4,6,Ø(1Ø)
         CLI   Ø(4),C'Q'
         BE    SEVER
         CLI   Ø(4),C'D'
         BE    CHECKDIR
         LR    1Ø,2
         LA    14,9
CHECKF   EQU   *
         CLI   Ø(1Ø),X'ØØ'
         BE    CHKNEXT
         L     7,1(1Ø)
         L     8,5(1Ø)
         ST    8,BUF
         CLC   Ø(16,7),Ø(5)
         BNE   CHKNEXT
         CLI   Ø(4),C'C'
         BNE   CHECKI
         LA    6,RET
         B     CLOSE
CHECKI   EQU   *
         CLI   Ø(4),C'I'
         BE    RET
         CLI   Ø(4),C'O'
         BE    RET
         CLI   Ø(4),C'M'
         BE    RET
         CLC   Ø(1,1Ø),Ø(4)
         BE    EXECIO
         B     ERDATA
CHKNEXT  EQU   *
         LA    1Ø,11(1Ø)
         BCT   14,CHECKF
         CLI   Ø(4),C'I'
         BE    CHECKDIR
```

```
         CLI   Ø(4),C'O'
         BE    CHECKDIR
         CLI   Ø(4),C'M'
         BNE   NOTOPEN
CHECKDIR EQU   *
         LA    14,X'57'
         LA    7,7(3)
FIND     EQU   *
         CLC   Ø(16,7),Ø(5)
         BE    OKNAME
         LA    7,47(7)
         BCT   14,FIND
         CLI   Ø(4),C'O'
         BNE   NOTFOUND
         CLC   3(4,3),=4X'FF'
         BE    NOTSPACE
         LA    14,X'57'
         LA    7,7(3)
FINDFREE EQU   *
         CLI   Ø(7),X'ØØ'
         BE    OKNAME
         LA    7,47(7)
         BCT   14,FINDFREE
         B     FULLDIR
OKNAME   EQU   *
         LR    1Ø,2
         LA    14,9
CHECKFL  EQU   *
         CLI   Ø(1Ø),X'ØØ'
         BE    ISFREE
         LA    1Ø,11(1Ø)
         BCT   14,CHECKFL
         B     FULLOPNT
ISFREE   EQU   *
         ST    7,1(1Ø)
         L     8,5(1Ø)
         ST    8,BUF
         CLI   Ø(4),C'O'
         BE    DELETE
         CLI   Ø(4),C'I'
         BE    OPEN
         CLI   Ø(4),C'M'
         BE    OPEN
         CLI   Ø(4),C'D'
         BNE   ERDATA
DELETE   EQU   *
         CLI   Ø(7),X'ØØ'
         BE    OPEN
         ST    5,REG5
         MVI   Ø(7),X'ØØ'
```

```
         MVI   REQ+3,X'Ø2'
         MVC   PATH(2),ENDPATH
         MVC   BLOCKNUM(4),ENDBL
         BAL   5,SIOONLY
         MVI   REQ+3,X'Ø1'
         MVC   Ø(6,8),1(3)
         MVC   1(2,3),BGNPATH
         MVC   3(4,3),BGNBL
         BAL   5,SIOONLY
         CLI   Ø(4),C'O'
         L     5,REG5
         BNE   RET
OPEN     EQU   *
         MVI   Ø(1Ø),C'W'
         MVC   9(2,1Ø),=X'ØØØ6'
         CLI   Ø(4),C'O'
         BNE   SETR
         MVC   BGNBL(4),3(3)
         MVC   ENDBL(4),3(3)
         MVC   BGNPATH(2),1(3)
         MVC   ENDPATH(2),1(3)
         MVC   BLKS(4),=X'ØØØØØØØ1'
         MVC   FNFT(16),Ø(5)
         B     SETM
SETR     EQU   *
         CLI   Ø(4),C'I'
         BNE   SETM
         MVI   Ø(1Ø),C'R'
         MVC   PATH(2),BGNPATH
         MVC   BLOCKNUM(4),BGNBL
         B     OKOPEN
SETM     EQU   *
         MVC   PATH(2),ENDPATH
         MVC   BLOCKNUM(4),ENDBL
OKOPEN   EQU   *
         MVI   REQ+3,X'Ø2'
         ST    8,BUF
         BAL   5,SIOONLY
         CLI   Ø(4),C'O'
         BNE   RET
         L     14,3(3)
         MVC   1(6,3),Ø(8)
         MVC   2(4,8),=4X'FF'
         CLC   3(4,3),=4X'ØØ'
         BNE   RET
         LA    14,1(14)
         ST    14,3(3)
         B     RET
EXECIO   EQU   *
         L     8,5(1Ø)
```

```
         ST    8,BUF
         CLI   Ø(4),C'W'
         BE    WRITE
         MVI   REQ+3,X'Ø2'
         ST    6,REG6
         XC    Ø(2,6),Ø(6)
         LA    6,2(6)
         L     7,1(1Ø)
READCYC  EQU   *
         LH    14,9(1Ø)
         AR    14,8
         CLI   Ø(14),X'FF'
         BNE   OKOFF
         CLC   2(4,8),=4X'FF'
         BE    RET
NEXTSEG  EQU   *
         BAL   5,SIO
         LA    14,6
         STH   14,9(1Ø)
         B     READCYC
OKOFF    EQU   *
         SR    5,5
         ICM   5,B'ØØ11',Ø(14)
         BZ    RET
         TM    Ø(14),X'8Ø'
         BNO   OKSIGN
         S     5,=F'32768'
         LNR   5,5
OKSIGN   EQU   *
         LPR   15,5
         L     1,REG6
         LH    Ø,Ø(1)
         AR    Ø,15
         STH   Ø,Ø(1)
         LA    14,2(14)
         LR    1,15
         LR    Ø,6
         LR    4,14
         AR    4,15
         SR    4,8
         STH   4,9(1Ø)
         AR    6,15
         MVCL  Ø,14
         LTR   5,5
         BM    NEXTSEG
         B     RET
WRITE    EQU   *
         LH    4,Ø(6)
         LA    6,2(6)
WRITECYC EQU   *
```

```
         LH    14,9(1Ø)
         LR    15,14
         AH    14,=H'-4Ø94'
         BM    CHKSPACE
         CH    14,=H'2'
         BE    CHKFULL
         LA    15,Ø(8,15)
         MVI   Ø(15),X'FF'
CHKFULL  EQU   *
         CLC   3(4,3),=4X'FF'
         BE    NOTSPACE
         L     7,1(1Ø)
         MVI   REQ+3,X'Ø1'
         MVC   PATH(2),ENDPATH
         MVC   BLOCKNUM(4),ENDBL
         MVC   ENDPATH(2),1(3)
         MVC   ENDBL(4),3(3)
         MVC   Ø(6,8),1(3)
         BAL   5,SIOONLY
         MVI   REQ+3,X'Ø2'
         BAL   5,SIO
         L     14,3(3)
         MVC   1(6,3),Ø(8)
         MVC   2(4,8),=4X'FF'
         CLC   3(4,3),=4X'ØØ'
         BNE   OKCHAIN
         LA    14,1(14)
         ST    14,3(3)
OKCHAIN  EQU   *
         LA    15,6
         STH   15,9(1Ø)
         L     15,BLKS
         LA    15,1(15)
         ST    15,BLKS
         B     WRITECYC
CHKSPACE EQU   *
         LPR   15,14
         AR    14,4
         BNP   FIT
         LR    4,14
         B     MOVE
FIT      EQU   *
         LR    15,4
         SR    4,4
MOVE     EQU   *
         LR    14,8
         AH    14,9(1Ø)
         STH   15,Ø(14)
         LTR   4,4
         BZ    OKLEN
         OI    Ø(14),X'8Ø'
```

```
OKLEN    EQU   *
         LA    5,2(15)
         AH    5,9(1Ø)
         STH   5,9(1Ø)
         LA    14,2(14)
         LR    Ø,6
         AR    6,15
         LR    1,15
         MVCL  14,Ø
         CH    5,=H'4Ø96'
         BE    OKBUF
         AR    5,8
         MVI   Ø(5),X'FF'
OKBUF    EQU   *
         LTR   4,4
         BZ    RET
         B     WRITECYC
CLOSE    EQU   *
         CLI   Ø(1Ø),C'W'
         BNE   CLEAR
         L     7,1(1Ø)
         MVI   REQ+3,X'Ø1'
         MVC   PATH(2),ENDPATH
         MVC   BLOCKNUM(4),ENDBL
         MVC   OFFSET(2),9(1Ø)
         MVC   BUF(4),5(1Ø)
         BAL   5,SIOONLY
         LA    15,DATE
         DC    X'83FFØØØC'
         MVC   LDATE(8),DATE
         MVC   LTIME(5),TIME
CLEAR    EQU   *
         MVI   Ø(1Ø),X'ØØ'
         BR    6
SEVER    EQU   *
         LA    4,9
         LR    1Ø,2
CLOSEF   EQU   *
         CLI   Ø(1Ø),C'W'
         BNE   NEXTF
         BAL   6,CLOSE
NEXTF    EQU   *
         LA    1Ø,11(1Ø)
         BCT   4,CLOSEF
         XC    PATH(2),PATH
         MVC   BLOCKNUM(4),=X'ØØØØØØØ1'
         MVI   REQ+3,X'Ø1'
         ST    3,BUF
         BAL   5,SIOONLY
         B     QUIT
NOTEDDEF EQU   *
```

```
         LINEDIT TEXT='--- EXPANDED $MDISK$ A not found',DOT=NO,COMP=NO
         B     RET
ERDATA   EQU   *
         LINEDIT TEXT='--- Wrong parameters',DOT=NO,COMP=NO
         B     SEVER
NOTFOUND EQU   *
         LINEDIT TEXT='--- Missing parameter or file not found',DOT=NO
         B     SEVER
NOTOPEN  EQU   *
         LINEDIT TEXT='--- File not open',DOT=NO,COMP=NO
         B     SEVER
FULLDIR  EQU   *
         LINEDIT TEXT='--- Too many files -> directory is full',DOT=NO
         B     SEVER
FULLOPNT EQU   *
         LINEDIT TEXT='--- Too many open files',DOT=NO,COMP=NO
         B     SEVER
NOTSPACE EQU   *
         LINEDIT TEXT='--- Expanded disk is full',DOT=NO,COMP=NO
         B     SEVER
CONFAIL  EQU   *
         MVC   MSG(1Ø),CNCT
         B     WRTMSG
DASDFAIL EQU   *
         MVC   MSG(1Ø),DASD
WRTMSG   EQU   *
         LH    14,PATH
         L     15,CUU
         SLL   14,1
         LH    1,Ø(14,15)
         SLL   1,4
         STH   1,DOUBLE
         UNPK  VADDR(3),DOUBLE(2)
         MVZ   VADDR(2),=2X'ØØ'
         TR    VADDR(3),HEX
         SR    1,1
         IC    1,CHECK
         CVD   1,DOUBLE
         UNPK  CODE(3),DOUBLE+6(2)
         OI    CODE+2,X'FØ'
         LINEDIT TEXTA=TXT,DOT=NO,COMP=NO
QUIT     EQU   *
         MVC   ISCON,C'N'
         XC    IUCVPLST(4Ø),IUCVPLST
         CMSIUCV SEVER,NAME=EMDMGR,PRMLIST=IUCVPLST,CODE=ALL
         HNDIUCV CLR,NAME=EMDMGR
         LR    1,3
         DMSFRET DWORDS=5632,LOC=(1)
RET      EQU   *
         SR    15,15
         L     13,SA+4
```

```
         RETURN (14,11)
SIO      EQU   *
         MVC   PATH(2),Ø(8)
         MVC   BLOCKNUM(4),2(8)
         ST    8,BUF
SIOONLY  EQU   *
         XC    IUCVPLST(4Ø),IUCVPLST
         IUCV  SEND,PRMLIST=IUCVPLST,DATA=PRMMSG,PATHID=PATH,          X
               PRMMSG=BLOCKNUM,TRGCLS=REQ
         WAITECB ECB=ECB,FORMAT=OS
         XC    ECB(4),ECB
         CLI   CHECK,X'ØØ'
         BNE   DASDFAIL
         BR    5
         DS    ØD
EXTHNDLR EQU   *
         USING *,15
         CLI   3(2),X'Ø2'
         BNE   SETFLAG
         MVC   PATH(2),Ø(2)
         MVC   CHECK(1),X'1Ø'(2)
         B     OKEXT
SETFLAG  EQU   *
         MVC   CHECK(1),X'F'(2)
OKEXT    EQU   *
         OI    ECB,X'4Ø'
         BR    14
SA       DC    2ØF'Ø'
         DS    ØD
INBUF    EQU   *+X'1Ø'
DATE     EQU   *
TIME     EQU   *+8
IUCVPLST DC    4ØX'ØØ'
EMDMGR   DC    CL8'EMDMGR'
BLOCKIO  DC    CL8'*BLOCKIO'
BLOCKNUM DS    F
BUF      DS    F
REQ      DC    F'Ø'
ECB      DC    F'Ø'
CUU      DS    F
REG23    DS    2F
REG5     EQU   *
REG6     DS    F
PATH     DS    H'Ø'
OPNTAB   DC    9XL11'ØØ'
ISCON    DC    C'N'
CHECK    DS    C
HEX      DC    C'Ø123456789ABCDEF'
DASD     DC    C'DASD i/o  '
CNCT     DC    C'Connection'
TXT      DC    AL1(TXTLEN)
```

```
TXTA     DC    C'--- XXXXXXXXXX failed for XXX with code XXX'
TXTLEN   EQU   *-TXTA
MSG      EQU   TXTA+4
VADDR    EQU   *-17
CODE     EQU   *-3
DOUBLE   EQU   IUCVPLST
         LTORG
FL       DSECT
FNFT     DS    CL16
BLKS     DS    F
BGNBL    DS    F
ENDBL    DS    F
BGNPATH  DS    H
ENDPATH  DS    H
OFFSET   DS    H
LDATE    DS    CL8
LTIME    DS    CL5
LEN      EQU   *-FL
         COPY  IPARML
         END   EMDM
```

EMDM EXEC

```
/*********************************************************************/
/***                                                    ***         ***/
/*** EMDM        creates expanded mini-disk            ***         ***/
/***                                                    ***         ***/
/*********************************************************************/
  CLRSCRN
  HI = '1DF8'X
  LO = '1DFØ'X
  QUERY V DA '(' STACK
  J = Ø
  K = Ø
  DO I = 1 TO QUEUED()
    PULL . CUU . . MODE DIR_CYL .
    IF MODE = 'R/W' & CUU ¬= '191' THEN
    DO
      J = J + 1
      CUU.J = CUU
      DIR_CYL.J = DIR_CYL
    END
  END
  STOR = Ø
ANOTHER_DISK_TO:
  SET CMSTYPE HT
  REL U
  SET CMSTYPE RT
  SAY '--- Type another'HI'CUU'LO'to build expanded disk or'         ,
          || HI'empty line'LO
```

```
  PULL CUU
  IF CUU = '' THEN
  SIGNAL REPORT
  DO I = 1 TO J
    IF CUU = CUU.I THEN
    SIGNAL CHECK_DISK
  END
  SAY '---'HI || CUU || LO'not attached or R/O disk'
  SIGNAL ANOTHER_DISK_TO
REP_CUU:
  SAY '---'HI || CUU || LO'has been already added to expanded disk'
  SIGNAL ANOTHER_DISK_TO
CHECK_DISK:
  DO I_ED = 1 TO K
    IF CUU = CUU_ED.I_ED THEN
    SIGNAL REP_CUU
  END
  SET CMSTYPE HT
  ACCESS CUU U
  IF RC ¬= Ø THEN
  SIGNAL FORMAT_IT
  QUERY DISK U '(STACK'
  PULL
  PULL . . . . CMS_CYL . BL .
  ADD_TXT = ''
FORMAT_IT:
  SET CMSTYPE RT
  SAY '--- Type'HI'1(YES) to format' CUU || ADD_TXT || LO
  PULL ANS
  IF ANS ¬= 1 THEN
  DO
    IF ADD_TXT ¬= '' THEN
    SIGNAL ANOTHER_DISK_TO
    IF DIR_CYL.I ¬= CMS_CYL THEN
    ADD_TXT = ' -> bad disk allocation map detected'
    IF BL ¬= 4Ø96 THEN
    ADD_TXT = ' -> block is not 4 Kb'
    IF ADD_TXT ¬= '' THEN
    SIGNAL FORMAT_IT;
    SAY 'EMDM'HI'do not work properly'LO'if' CUU 'has nonzeros records'
    SIGNAL RESERVE_IT
  END;
  SET CMSTYPE HT
  QUEUE 1
  QUEUE ED || K+1
  FORMAT CUU U '(BL 4K'
RESERVE_IT:
  SET CMSTYPE RT
  SAY '--- Type'HI'1(YES) to reserve' CUU || LO
  PULL ANS
  IF ANS ¬= 1 THEN
```

```
  SIGNAL ANOTHER_DISK_TO
  SET CMSTYPE HT
  QUEUE 1
  RESERVE EMDM $STORE$ U6
  FI CHKDISK DISK CUU
  SET CMSTYPE RT
  EMDMCHK
  IF X2D(C2X(OFFSET)) > 4000 THEN
  DO
    SAY '--- Code'HI||STRIP(X2D(C2X(OFFSET)) - 4000)LO'when checked' ,
        || HI || CUU || LO
    SIGNAL ANOTHER_DISK_TO
  END
  SET CMSTYPE HT
  LISTFILE EMDM $STORE$ U6 '(' STACK ALL
  PULL . . . . . BLK .
  STOR = STOR + BLK
  K = K + 1
  BUF.K = RIGHT(X2C(D2X(4096)), 4, '0'X)           ||                ,
          RIGHT(OFFSET, 4, '0'X)                   ||                ,
          RIGHT(X2C(CUU), 2, '0'X)
  LAST_CUU.K = CUU
  LAST_BLK.K = BLK
  AC CUU U
  REC = LEFT(RIGHT(X2C(D2X(K)), 2, '0'X)'FFFFFFFF'X, 4096, '0'X)
  EXECIO 1 DISKW EMDM $STORE$ U6 BLK '(FINI VAR REC'
  IF K > 1 THEN
  DO
    REL U
    IND = K - 1
    AC LAST_CUU.IND U
    BLK = LAST_BLK.IND
    REC = LEFT(RIGHT(X2C(D2X(IND)), 2, '0'X)'00000001'X, 4096, '0'X)
    EXECIO 1 DISKW EMDM $STORE$ U6 BLK '(FINI VAR REC'
  END
  ELSE
  DO
    REC = LEFT('00000000000002'X, 4096, '0'X)
    EXECIO 1 DISKW EMDM $STORE$ U6 1 '(FINI VAR REC'
  END
  CUU_ED.K = CUU
  SET CMSTYPE RT
  SAY '---'HI || CUU || ' is added'LO'to expanded disk'
  SIGNAL ANOTHER_DISK_TO
REPORT:
  SET CMSTYPE HT
  ERASE EXPANDED $MDISK$ A
  SET CMSTYPE RT
  IF K = 0 THEN
  EXIT
  EXECIO K DISKW EXPANDED $MDISK$ A 1 F 10 '(' FINI STE BUF.
```

```
  CLRSCRN
  SAY
  SAY '         --- Next mini-disks are added to expanded disk ---'
  SAY
  DO I = 1 TO K BY 2Ø
    LINE = ''
    DO J = I TO MIN(K, I + 19)
      LINE = LINE CUU_ED.J
    END
    SAY LINE
  END
  SAY
  SAY
  NUMERIC DIGITS 12
  SAY '         --- Expanded disk definition ---'
  SAY
  SAY HI'       Mini-disks' || RIGHT(K, 23, '.')LO
  SAY HI'           Blocks' || RIGHT(STOR, 23, '.')LO
  SAY HI'            Bytes' || RIGHT(STOR * 4Ø96, 23, '.')LO
  SAY HI'               MB' ||                                          ,
   RIGHT(STRIP(FORMAT(STOR * 4Ø96 / 1Ø24 / 1Ø24, 8, 2)), 23,'.')LO
  SAY HI'               GB' ||                                          ,
   RIGHT(STRIP(FORMAT(STOR * 4Ø96/1Ø24/1Ø24/1Ø24, 8, 2)), 23,'.')LO
  SAY
```

EMDMLIST EXEC

```
/*********************************************************************/
/***                                                  ***         ***/
/*** EMDMLIST     shows expanded mini-disk            ***         ***/
/***                                                  ***         ***/
/*********************************************************************/
  CLRSCRN
  HI = '1DF8'X
  LO = '1DFØ'X
  CLRSCRN
SIGNAL ON ERROR
  SET CMSTYPE HT
  MESSAGE = '    File EXPANDED $MDISK$ A not found'
  EXECIO '*' DISKR EXPANDED $MDISK$ A 1 '(' FINI STE CUU. M 9 1Ø
  DO I = 1 TO CUU.Ø
    CUU.I = RIGHT(C2X(CUU.I), 3, 'Ø'X)
  END
  MESSAGE = '   'CUU.1 'not formated or detached'
  AC CUU.1 U
  MESSAGE = '    Expanded disk is destroyed'
  EXECIO 1 DISKR EMDM $STORE$ U 1 '(' FINI VAR BUF
SIGNAL OFF ERROR
  SET CMSTYPE RT
  SAY
```

```
  SAY ' Filename Filetype   Record Start Blk/CUU  End Blk/CUU '      ,
      'Free off   Date   Time'
  DO I = 1 TO 87
    FILE_DESCR = SUBSTR(BUF, (I - 1) * 47 + 8, 47)
    IF SUBSTR(FILE_DESCR, 1, 1) = 'Ø'X THEN
    ITERATE
    PARSE VAR FILE_DESCR  1 FN                                      ,
                          9 FT                                      ,
                         17 BLKS                                    ,
                         21 BGNBL                                   ,
                         25 ENDBL                                   ,
                         29 BGNPATH                                 ,
                         31 ENDPATH                                 ,
                         33 OFFSET                                  ,
                         35 LDATE                                   ,
                         43 LTIME
    I_BGN = C2D(BGNPATH) + 1
    I_END = C2D(ENDPATH) + 1
    SAY HI || FN FT RIGHT(C2D(BLKS), 8)COPIES(' ',1)                ,
              RIGHT(C2D(BGNBL), 8)                                  ,
              RIGHT(CUU.I_BGN, 3)                                   ,
              RIGHT(C2D(ENDBL), 8)                                  ,
              RIGHT(CUU.I_END, 3)COPIES(' ',1)                      ,
              RIGHT(C2D(OFFSET), 5)COPIES(' ', 3)                   ,
              LDATE LTIME || LO
  END
  SAY
  EXIT
ERROR:
  SET CMSTYPE RT
  SAY
  SAY HI MESSAGE LO
  SAY
```

USER VIRTUAL MACHINE GETTING READY

Before you start using EMDM, you must ensure that the OPTION directory control statement for a user's virtual machine has the correct MAXCONN value. The MAXCONN parameter determines the maximum numbers of IUCV connections allowed for a virtual machine. EMDM uses CP DASD Block I/O System service and makes a connection for each expanded disk mini-disk. So MAXCONN must be set to be equal to or greater than the number of expanded disk virtual DASD devices.

Dobrin Goranov
Information Sevices Co (Bulgaria)

Sterling Software has released VM:Webserver, aimed at OfficeVision for VM/ESA operating environments. The OfficeVision extension to VM:Webserver provides a web browser interface for OfficeVision/VM electronic mail and calendar system (which is apparently accessed daily by more than eight million users).

The product provides a user-friendly graphical alternative to OfficeVision/VM's native 3270 character-based interface. The VM:Webserver for OfficeVision calendar interface allows users to manage their calendars, check colleagues' schedules, and arrange meetings through a point-and-click interface. The electronic mail interface also extends OfficeVision's e-mail features by allowing users to attach files to their notes, as can be done with Internet e-mail.

Sterling reckons customers benefit from a centralized mainframe environment which minimizes the costs of deployment, administration, and support. Organizations can continue to take advantage of the robustness, scalability, and security that are hallmarks of both the VM/ESA mainframe environment and OfficeVision/VM, while leveraging existing investments in hardware, software, and people.

The calendar interface is available now. VM:Webserver's price is based on CPU group size or on the number of users. The standard price on a model group 40 is $25,300. The standard price for 2,501-5,000 users is $25,000.

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

* * *

IBM has announced a range of client enhancements for its ADSTAR Distributed Storage Manager tools for VM, allowing for back-up and restore operations for ADSM OS/2, Windows NT, Windows 95, and AIX client machines to be carried out from a Web browser on almost any workstation platform, or network.

Among the functions being made available to ADSM users are the ADSM WebShell Interface, which runs on the client machine for any ADSM OS/2, Windows 95, Windows NT, or AIX client. It enables the initiation and management of ADSM back-up and restore operations for these client machines from almost any workstation, via a Web browser. It's downloaded and installed with the ADSM OS/2, Windows 32-bit, or AIX clients.

Also new is AFS/DFS support, which means users of AFS and DFS environments can use ADSM for VM Version 2.1 servers to back-up and restore AFS and DFS systems.

The software's back-up/restore and archive/retrieve functions now support files larger than 2GB. Finally, there's support for HP-UX clients, through the X/Open Back-up Services API.

For further information contact your local IBM representative.