130 VM

June 1997

update

In this issue

VM Update

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: 940 455 7050

Australian office
Xephon/RSM
GPO Box 6258
Halifax Street
Adelaide, SA 5000
Australia
Telephone: 088 223 1391

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 £170.00 in the UK; $255.00 in the USA and Canada; £176.00 in Europe; £182.00 in Australasia and Japan; and £180.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1990 issue, are available separately to subscribers for £14.50 ($21.50) 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.

Printed in England.

Save erased files

SAVERASE saves erased files for a specified period, called the saving period. During this period, the files may be restored repeatedly on every write-accessed mini-disk of the user's virtual machine. The restoration of the file does not destroy its saved copy.

For each erased file, SAVERASE performs the following actions:

- Sends the file to the common accessible virtual machine named ERASED, where it is kept for the saving period.
- Deletes the file from the user's mini-disk.

So SAVERASE protects files from accidental erasure.

When debugging complex projects, SAVERASE may be used to create generations of input test data. This is because for each erased file, SAVERASE generates a unique internal identifier, and subsequent use of SAVERASE against the new version of the file will create a new saved copy. The individual copy can be identified by the date and time of creation.

SAVERASE is written in REXX.

SAVERASE FORMAT

The format of SAVERASE is the same as the format of the CMS ERASE command without additional options:

```
SAVERASE <fn | *> <ft | *> <fm>
```

where fn is the filename, ft is the filetype, and fm is the filemode. There is no default value for fm – it must always be defined.

SAVERASE may be invoked directly from a FILELIST screen in the same way as CMS ERASE, or from the command line.

For example:

```
SAVERASE * * B
```

saves and erases all files on mini-disk B

```
SAVERASE * TESTDATA A
```

saves and erases the files with a filetype of TESTDATA on mini-disk A.

After successfully saving or erasing a file, the following message will be displayed:

```
>>>  fn ft <<< saved for 10 working days
```

where fn and ft are the filename and filetype of the last processing file.

SAVERASE RELIABILITY

In order to guarantee the restoration of erased files during the saving period, the working day counter is used. This counter is set when a file is saved. At log-on time for virtual machine ERASED, the counter is decreased only if the PROFILE EXEC identifies a new working day. If the working day counter is less than zero, the corresponding saved file is erased.

Error identification	Possible problem	Action
VM ERASED not in CP QUERY NAMES list	The size of erased file is greater than 4500 4K pages(60MB)	Make the size of TDISK bigger in PROFILE EXEC - VM ERASED
	191 disk of VM ERASED is full	Set more space for 191 disk in system directory - VM MAINT
	More than 1600 erased files are sent to VM ERASED when it is in sleep state	Decrease SLEEP value in PROFILE RESUME - VM ERASED

Figure 1: VM ERASED diagnostic guide

A new working day is identified when the system date is greater than the last saved date. Therefore the working day counter will not change or changes only once if a wrong system date is set unexpectedly and then the correct system date is entered. Generally, the erased files may be saved longer, but they will never be deleted before the end of the saving period.

If VM ERASED logs off, then to restore erased files look at the reader queue of VM ERASED. After this, use the diagnostic guide in Figure 1 for some other possible actions. In most cases no information will be lost. The erased files will always be recovered.

RESTORE ERASED FILES

UNERASE EXEC shows the current status of saved files and supports the following prefix commands, which perform file restores:

- R [fm] – restores the file on mini-disk fm. If fm is not specified, then the file is restored on mini-disk A. If UNERASE finds a file with the same filename on the target disk, the command is cancelled.
- RR [fm] – restores the file on mini-disk fm. If fm is not specified, then the file is restored on mini-disk A. If a file with the same filename already exists on the target mini-disk, it will be replaced.

The PFK settings for the UNERASE panel are:

- PF2 – shows a list of files, sorted by date and time of erasure.
- PF3 – quit.
- PF4 – begin.
- PF5 – end.
- PF7 – backward.
- PF8 – forward.
- PF9 – sorted list of files by name, date, and time of erasure.

PROFILE RESUME

```
/*****************************************************************/
/***                                                  ***         ***/
/*** PROF_RESUME        protected file erase          ***  DG'97  ***/
/***                                                  ***         ***/
/*****************************************************************/
/***   SIZE 00045  VER 1.0 MOD 03  TIME 16:35:45  DATE 03/01/97   ***/
/*****************************************************************/

  DAYS_TO_SAVE = 10    /* the files to be erased after 10 work days */
  EXECIO 1 DISKR $ERASED$ LASTDATE D 1 '(' FINI VAR CHECK_DATE
  IF CHECK_DATE < DATE(C) THEN
  DO
    EXECIO 1 DISKW $ERASED$ LASTDATE D 1 '(' FINI ST DATE(C)
    LISTFILE '*' DIR D '(' STACK
    DO I = 1 TO QUEUED()
      PULL VMID .
      X VMID DIR D '(' PROF UNERCHK
    END
  END
  DO FOREVER
    SLEEP 99 SEC
    AREA = DIAG(8, QUERY R ALL, 64000)
    BGN = INDEX(AREA, '15'X) + 1
    DO FOREVER
      END = INDEX(AREA, '15'X, BGN)
      IF END = 0 THEN
      LEAVE
PARSE VALUE SUBSTR(AREA,BGN,END-BGN) WITH VMID . . . . . . . . FN FT .
      BGN = END + 1
      DISK LOAD '(' NOP NOR OLDD
      IF RC ¬= 0 THEN
      LOGOFF
      SAVE_FN = DATE('J')
      SAVE_FT = TIME('S')RIGHT(I, 3, '0')
      IF RC = 0 THEN
      EXECIO 1 DISKW VMID DIR D '(' FINI                              ,
      ST RIGHT(DAYS_TO_SAVE - 1, 2, '0') SAVE_FN SAVE_FT LEFT(FN, 8) ,
          LEFT(FT, 8) LEFT(DATE('W'), 9) RIGHT(DATE(), 11) TIME()
      COPY FN FT A SAVE_FN SAVE_FT D '(' PA
      IF RC ¬= 0 THEN
      LOGOFF
      W VMID '>>>' FN FT '<<< saved for' DAYS_TO_SAVE 'working days'
      ERASE FN FT A
    END
  END
```

UNERCHK XEDIT

```
/*********************************************************************/
/***                                                        ***    ***/
/*** UNERCHK            protected file erase                ***  DG'97  ***/
/***                                                        ***    ***/
/*********************************************************************/
/***   SIZE ØØØ37  VER 1.Ø MOD Ø2  TIME 11:15:28  DATE Ø4/Ø1/97   ***/
/*********************************************************************/

  EXT '/SIZ'
  MAKEBUF
  ':1'
  CL ':1'
  DO I = 1 TO SIZE.1
    STA 1 1 17
    PULL CHECK_DAYS FN FT .
    IF CHECK_DAYS = Ø THEN
    DO
      ADDRESS CMS ERASE FN FT D
      CR '$$'
    END
    ELSE
    CR RIGHT(CHECK_DAYS - 1, 2, 'Ø')
    N
  END
  DROPBUF
  ':1'
  ALL '/$$'
  IF RC = Ø THEN
  DELET '*'
  EXT '/SIZ/FN'
  IF SIZE.1 = Ø THEN
  DO
    ADDRESS CMS ERASE FNAME.1 DIR D
    QQ
  END
  ELSE
  FF
```

PROFILE EXEC

```
/*********************************************************************/
/***                                                        ***    ***/
/*** PROFILE            protected file erase                ***  DG'97  ***/
/***                                                        ***    ***/
/*********************************************************************/
/***   SIZE ØØØ2Ø  VER 1.Ø MOD Ø1  TIME 16:59:5Ø  DATE Ø3/Ø1/97   ***/
/*********************************************************************/
```

```
REL A
AC 191 D
DEF T338Ø Ø91 3Ø
QUEUE 1
QUEUE REP191
FORMAT Ø91 A
IF RC ¬= Ø THEN
EXIT
COPY PROFILE RESUME D '=' EXEC A '(' OLDDATE
COPY UNERCHK XEDIT D '= = A (OLDDATE'
SET RUN ON
EXEC PROFILE
```

UNERASEP XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** UNERASEP           protected file erase            ***  DG'97  ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE ØØØ54  VER 1.Ø MOD Ø3  TIME 17:ØØ:35  DATE Ø3/Ø1/97   ***/
/*********************************************************************/

EXT '/LS'
PFØ1 NULLKEY
PFØ2 ONLY DMSXMS 4 17
PFØ3 ONLY QQUIT
PFØ4 ONLY ':1'
PFØ5 ONLY BOT
PFØ7 ONLY 6 - LSCREEN.1
PFØ8 ONLY LSCREEN.1 - 6
PFØ9 ONLY DMSXMS 19 35
PF11 NULLKEY
PF1Ø NULLKEY
PF12 NULLKEY
ENT  ONLY CURS SCR 4 1
CMD  OFF
CURL ON 4
LINEN OFF
MSGL ON 4 2 O
NUM  OFF
PRE  NULL
SCAL OFF
SER  OFF
STAY ON
TOFEOF OFF
VER  19 91
PREF SYN R UNERCPY
PREF SYN RR UNERCPY
```

```
  EXT '/SIZ'
  RESER 1 HI 'Saved' RIGHT(SIZE.1, 4, 'Ø') 'files'                    ,
             COPIES(' ', 31)'***  Protect file erase  ***'
  RESER 2 HI  COPIES(' ', 46) '*****  Ver 1.Ø  (C) DG''97  *****'
  RESER 3 NO  COPIES(' ', 6)'Originid'COPIES(' ', 1Ø)'Date of'        ,
             'saving'COPIES(' ', 17)'Days before erase'
  RESER LSCREEN.1-1 NO '   Pfk: 2 Unsort  3 Exit  4 Bgn  5 End'       ,
            ' 7 Ba  8 Fo  9 Sort'
  RESER LSCREEN.1 NO 'Prefix: R [fm] Restore  RR [fm] Replace'
  ':'1
  STACK '*' 1 2
  CL ':'68
  ':'1
  DO I = 1 TO QUEUED()
    PULL DAYS
    CR CENTRE(DAYS, 24, '-')
    N
  END
  ':'1
  CURS SCR 4 1
```

UNERCPY XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** UNERCPY            protected file erase           ***  DG'97  ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE ØØØ56  VER 1.Ø MOD Ø3  TIME 15:Ø3:Ø5  DATE Ø3/Ø1/97   ***/
/*********************************************************************/

  EXT '/CURS'
  IF QUEUED() > Ø THEN
  PULL
  PARSE SOURCE . . . . . UNER_FUNC .
  ARG . . LINE_NO FM .
  IF LENGTH(FM) ¬= Ø THEN
  DO;
     IF LENGTH(FM) ¬= 1 |                                              ,
     VERIFY(FM, 'ABCDEFGHIJKLMNOPQRSTUVWXYZ') ¬= Ø THEN
     DO
       EMSG '--- Invalid mode' FM
       EXIT
     END
     MAKEBUF
     QUERY DISK FM '(' STACK LIFO
     PULL . . . MODE .
     DROPBUF
     IF MODE ¬= 'R/W' THEN
```

```
    DO
      EMSG '--- Disk ' FM 'is read/only'
      EXIT
    END
  END
  ELSE
  FM = A
  ':'LINE_NO
  STACK 1 4 32
  PULL MYN MYT OLDN OLDT .
  ADDRESS CMS
  SET CMSTYPE HT
  STATE OLDN OLDT FM
  SAVE_RC = RC
  IF UNER_FUNC = 'RR' THEN
  SAVE_RC = 1
  IF SAVE_RC = Ø THEN
  MSG = 'Found'
  ELSE
  DO
    COPYFILE MYN MYT U OLDN OLDT FM '(' UNPA OLDDATE REP
    IF RC ¬= Ø THEN
    MSG = 'Failed'
    ELSE
    MSG = 'On' FM
  END
  ADDRESS XEDIT
  CL ':68'
  CR CENTRE(MSG 'at' TIME(), 24, '-')
  PUSH CURS SCR CURSOR.1 1
```

SAVERASE EXEC

```
/*********************************************************************/
/***                                                    ***         ***/
/*** SAVERASE           protected file erase            ***  DG'97  ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE ØØØ59  VER 1.Ø MOD Ø3  TIME 13:48:45  DATE Ø4/Ø1/97   ***/
/*********************************************************************/

  ARG FN_G FT_G FM_G EMPTY
  SET CMSTYPE HT
  IF ¬ (LENGTH(FM_G) = 1 | LENGTH(FM_G) = 2) THEN
  EXIT
  MAKEBUF
  LISTFILE FN_G FT_G FM_G '(' STACK
  IF RC ¬= Ø THEN
  EXIT
```

```
  N_SAVE = QUEUED()
  DO I = 1 TO N_SAVE
    PULL FULL_ID.I
  END
  DROPBUF
  DO N = 1 TO N_SAVE
    PARSE VAR FULL_ID.N FN FT FM .
    STATEW FN FT FM
    IF RC ¬= Ø THEN
    EXIT
    QUERY PUNCH '(' STACK
    PULL
    FILES = QUEUED()
    DO I = 1 TO FILES
      PULL . FILE_ID.I .
    END
    DISK DUMP FN FT FM
    ERASE FN FT FM
    IF FILES > Ø THEN
    DO
      MAKEBUF
      QUERY PUNCH '(' STACK
      PULL
      DO I = 1 TO QUEUED()
        PULL . FILE .
        DO J = 1 TO FILES
          IF FILE = FILE_ID.J THEN
          DO
            FILE_ID.J = FILE_ID.FILES
            FILES = FILES - 1
            ITERATE I
          END
        END
        LEAVE I
      END
      DROPBUF
      PARSE VALUE DIAGRC(8,TRAN PUNCH FILE ERASED RDR,128) WITH CODE .
    END
    ELSE
    PARSE VALUE DIAGRC(8, TRAN PUNCH ALL ERASED RDR, 128) WITH CODE .
    IF CODE = Ø THEN
    ERASE FN FT FM
  END
```

UNERASE EXEC

```
/*********************************************************************/
/***                                                  ***         ***/
/*** UNERASE            restore erased file           ***  DG'97  ***/
```

```
/***                                                        ***         ***/
/*************************************************************************/
/***   SIZE 00026  VER 1.0 MOD 02  TIME 16:00:41  DATE 03/01/97   ***/
/*************************************************************************/

  HI = '1DF8'X
  LO = '1DF0'X
  SET CMSTYPE HT
  REL U
  DIAG(8, DET 119, 128)
  DIAG(8, LINK ERASED 191 119 RR, 128)
  AC 119 U
  STATE USERID() DIR U
  IF RC = 0 THEN
  X USERID() DIR U '(' PROF UNERASEP W 99
  ELSE
  DO
    SET CMSTYPE RT
    SAY '--- Saved files'HI'not found'LO
  END
  SET CMSTYPE HT
  REL U
  DIAG(8, DET 119, 128)
```

PREPARING SAVERASE

To add SAVERASE commands to a particular VM installation, the following steps must be carried out:

1 Update system directory:

- Log-on to VM MAINT.
- Add the entry below to the directory, replacing <bgn> and <vol> with their proper values.

```
USER ERASED ERASEDPW 1M 1M BG
 ACCOUNT 00000000 ERASED
 IPL CMS PARM NOSPROF AUTOCR
 CONSOLE 009 3215
 SPOOL 00C 2540 READER *
 LINK MAINT 190 190 RR
 MDISK 191 3380 <bgn> 200 <vol> WR ALL
```

- Issue command DIRECT against the system directory.

2 Update PROFILE EXEC:

- Log-on to VM AUTOLOG1.

- Add the entry below to the PROFILE EXEC:

```
CP AUTOLOG ERASED ERASEDPW
```

3 Prepare VM ERASED:

- Log-on to VM ERASED.
- FORMAT the 191 mini-disk as mini-disk A.
- Copy to mini-disk A the distribution files PROFILE EXEC, PROFILE RESUME, and UNERCHK XEDIT.
- Log-off VM ERASED.

4 Copy the distribution files – SAVERASE EXEC, UNERASE EXEC, UNERASEP XEDIT, and UNERCPY XEDIT – to a commonly accessible mini-disk, ie VM MAINT's 190 or 319.

5 AUTOLOG VM ERASED – do not start it by LOGON and DISC.

All this make the SAVERASE command ready to use at your site. Next time VM ERASED will be started at IPL time by VM AUTOLOG1.

To change the length of the saving period, modify the value of variable DAYS_TO_SAVE in PROFILE RESUME.

Dobrin Goranov
Information Services Co (Bulgaria)

CMS job staging EXEC

Like many other data centres, we sometimes have to run jobs in our production environment that are submitted by programmers, end users, or outside sources (most of these are jobs that print reports from our production archive database). Since we only allow our scheduler, data centre, and support staff to submit directly into production, we implemented procedures allowing other users to SENDFILE or TRANSFER jobs to a CMS service machine, which is monitored by operators who, after reviewing the jobs, send them into production. To help automate this process, the JCLTRAK EXEC shown below was implemented.

JCLTRAK is a REXX EXEC executed at regular intervals. Its primary function is to move accumulated jobs from the service machine's reader to the A disk, where operators can process them further. At the same time, ageing is done to delete A disk files older than a specified number of days. This allows the jobs to be retained for a period of time in case there are any questions or problems.

New jobs are added to the A disk by doing a QUERY READER and then RECEIVEing each file. The ageing process is accomplished by doing a virtual LISTFILE and then converting the date of each member to Julian format where it can be compared to the current date. Expired members are then DELETEd. A table of days is used to handle leap years and adjust February days accordingly.

```
/*****************************************************************/
/***  Program Name  JCLTRAK                                    ***/
/***                                                           ***/
/***  Function      Purge members residing on the JCL staging   ***/
/***                library that have expired.                 ***/
/***                                                           ***/
/***                Add new members to the JCL staging library ***/
/***                that were SENDFILEd by various users.      ***/
/***                                                           ***/
/***  Parameters    PARM1 = Number of days to retain JCL members ***/
/***                        on disk                            ***/
/***                                                           ***/
/***                PARM2 = Filemode of the JCL staging library ***/
/***                                                           ***/
/*****************************************************************/
```

```
/*********************************************************************/

  SET CMSTYPE HT

  RETAIN   = 5            /***  Default nbr of days to keep       ***/
  FMODE    = A            /***  Default filemode of staging lib   ***/

  PARSE UPPER ARG PARM1 PARM2

  IF PARM1 ¬= '' THEN RETAIN = PARM1
  IF PARM2 ¬= '' THEN FMODE  = PARM2

  M.   = ''  /*******************************************************/
  M.1  = 31  /***  Days-in-month table.  Adjust February days     ***/
  M.2  = 28  /***  if current year is a leap year.                ***/
  M.3  = 31  /*******************************************************/
  M.4  = 3Ø
  M.5  = 31 ; IF SUBSTR(DATE(U),7,2)      = Ø THEN M.2 = 29
  M.6  = 3Ø ; IF SUBSTR(DATE(U),7,2) // 4 = Ø THEN M.2 = 29
  M.7  = 31
  M.8  = 31
  M.9  = 3Ø
  M.1Ø = 31
  M.11 = 3Ø
  M.12 = 31

/*********************************************************************/
/***  Build virtual filelist;  Calculate Julian expiration date   ***/
/*********************************************************************/

 'LISTFILE * * ' FMODE ' (DATE STACK FIFO'

  EXPDATE = (DATE(J) - RETAIN)
  IF SUBSTR(EXPDATE,3,3) > 365 THEN EXPDATE = EXPDATE - 635

/*************************************************************/
/***  Convert filelist date to Julian & compare...        ***/
/*************************************************************/

  DELETED = Ø
  DO UNTIL QUEUED() = Ø

     PARSE PULL LINE
     PARSE VAR LINE FNAME FTYPE FFM . . . . FDATE .
     IF FFM = 'AØ' | FNAME = 'PROFILE' | FNAME = 'JCLTRAK' THEN ITERATE

     FDATE = RIGHT(FDATE,8,Ø)
     WMTH = SUBSTR(FDATE,1,2)
     WDAY = SUBSTR(FDATE,4,2)
     WYR  = SUBSTR(FDATE,7,2)
```

```
     WJUL = WYR * 1000

     DO FOREVER

        IF WMTH = '01' THEN LEAVE  ;  WJUL = WJUL + M.1
        IF WMTH = '02' THEN LEAVE  ;  WJUL = WJUL + M.2
        IF WMTH = '03' THEN LEAVE  ;  WJUL = WJUL + M.3
        IF WMTH = '04' THEN LEAVE  ;  WJUL = WJUL + M.4
        IF WMTH = '05' THEN LEAVE  ;  WJUL = WJUL + M.5
        IF WMTH = '06' THEN LEAVE  ;  WJUL = WJUL + M.6
        IF WMTH = '07' THEN LEAVE  ;  WJUL = WJUL + M.7
        IF WMTH = '08' THEN LEAVE  ;  WJUL = WJUL + M.8
        IF WMTH = '09' THEN LEAVE  ;  WJUL = WJUL + M.9
        IF WMTH = '10' THEN LEAVE  ;  WJUL = WJUL + M.10
        IF WMTH = '11' THEN LEAVE  ;  WJUL = WJUL + M.11
        IF WMTH = '12' THEN LEAVE

     END ; FDATE = WJUL + WDAY

    /*********************************************************/
    /***  Delete expired members...                        ***/
    /*********************************************************/

     IF FDATE < EXPDATE THEN DO
        DELETED = DELETED + 1
       'ERASE' FNAME FTYPE FMODE
     END

  END

  SET CMSTYPE RT
  SAY ' '
  SAY 'JCLTRAK  Number of modules deleted = 'DELETED

/*********************************************************************/
/***  Read new JCL members onto disk...                            ***/
/*********************************************************************/

 ADDED = 0 ; SET CMSTYPE HT

 'Q R * ALL (STACK'
  PARSE PULL ENTRY                    /* DISCARD HEADER RECORD     */

 DO UNTIL QUEUED() = 0
    PARSE PULL ENTRY
    PARSE VAR ENTRY . SPID . TYPE . . . . . FID .
    IF TYPE ¬= 'PUN' THEN ITERATE
   'RECEIVE' SPID '= =' FMODE '(REPL NOLOG NEWDATE'
    IF RC ¬= 0 THEN LEAVE
    ADDED = ADDED + 1
```

```
END ; SET CMSTYPE RT

SAY 'JCLTRAK  Number of modules added   = 'ADDED
SAY ' '

EXIT
```

Steve Bernard
Senior Systems Programmer
Arkansas Farm Bureau (USA)

Electronic bulletin board – part 3

This month we continue the code for a bulletin board system.

```
DISPLAY:
  'CONWAIT'
  'DESBUF'
  subjx  = '!"' || subj || '!@'
  flashx = '!"' || flash || '!@'
  cronx  = '!"' || cron || '!@'
  profx  = '!"' || prof || '!@'
  fnx    = '!"' || fn || '!@'
  ftx    = '!"' || ft || '!@'
  xxx   = DATE(W) '      '
  yyy   = 'Electronic News - PROFS Bulletin Board'
  zzz   = '     ' DATE(USA)
  'COMMAND SET RESERVED  1 N  !@' xxx ' !%' yyy ' !@' zzz
  if fullscreen = 'YES'
     then do
          'COMMAND SET CURL ON 3'
          xxx = COPIES('-',79)
          'COMMAND SET RESERVED  2 GREEN NONE N' xxx
          'COMMAND SET RESERVED  3 OFF'
          'COMMAND SET RESERVED  4 OFF'
          'COMMAND SET RESERVED  5 OFF'
          'COMMAND SET RESERVED  6 OFF'
          'COMMAND SET RESERVED  7 OFF'
          'COMMAND SET RESERVED  8 OFF'
          'COMMAND SET RESERVED  9 OFF'
          'COMMAND SET RESERVED 10 OFF'
          'COMMAND SET RESERVED 11 OFF'
          'COMMAND SET RESERVED 12 OFF'
          'COMMAND SET RESERVED 13 OFF'
          'COMMAND SET RESERVED 14 OFF'
          end
     else do
```

```
          'COMMAND SET CURL ON 15'
          'COMMAND SET RESERVED  2 BLUE NONE N  '
          xxx  = '  Supply the information below for "'desc'"'
          'COMMAND SET RESERVED  3 BLUE NONE N' xxx
          'COMMAND SET RESERVED  4 GREEN NONE N  '
          xxx = 'Subject:' || subjx || 'Broadcast? (Y/N)' || flashx
          'COMMAND SET RESERVED  5 GREEN NONE N' xxx
          'COMMAND SET RESERVED  6 GREEN NONE N  '
    xxx = 'To reference a!%PROFS draft document!@supply a CRON number:'
          'COMMAND SET RESERVED  7 GREEN NONE N' xxx cronx
          xxx  = '                    !%-- or --!@  '
          'COMMAND SET RESERVED  8 GREEN NONE N' xxx
    xxx = 'To reference a!%PROFS personal document!@supply a filename:'
          'COMMAND SET RESERVED  9 GREEN NONE N' xxx profx
          xxx  = '                    !%-- or --!@  '
          'COMMAND SET RESERVED 1Ø GREEN NONE N' xxx
    xxx = 'To reference a!%non-PROFS document!@filename and filetype: '
          'COMMAND SET RESERVED 11 GREEN NONE N' xxx fnx ftx
          xxx  = '                    !%-- or --!@  '
          'COMMAND SET RESERVED 12 GREEN NONE N' xxx
    xxx = 'Or, just type !%additional comments!@below...              '
          'COMMAND SET RESERVED 13 GREEN NONE N' xxx
          xxx = COPIES('-',79)
          'COMMAND SET RESERVED 14 GREEN NONE N' xxx
          end
  xxx = COPIES('-',79)
  'COMMAND SET RESERVED -3 BLUE NONE N' xxx
  if func = 'REPLACE'
     then pf_func = 'Repl bulletin'
     else pf_func = 'Add bulletin '
  if fullscreen = 'YES'
  then xxx  = 'PF1='pf_func '2=Add line  3=Del line',
              ' 4=View      5=Calendar   6=HalfScreen'
  else xxx  = 'PF1='pf_func '2=Add line  3=Del line',
              ' 4=View      5=Calendar   6=FullScreen'
  'COMMAND SET RESERVED -2 BLUE NONE HIGH' xxx
  xxx  = 'PF7=Refresh       8=          9=Help',
         '    1Ø=Forward  11=Backward  12=Return '
  'COMMAND SET RESERVED -1 BLUE NONE HIGH' xxx
  'EXTRACT /CURSOR'
  if subj = '_____________________________________________________'
     then 'COMMAND CURSOR SCREEN 5 1Ø'
     else if CURSOR.3 = -1
             then 'CURSOR SCREEN' CURSOR.1 CURSOR.2
             else 'CURSOR FILE' CURSOR.3 CURSOR.4
return
INIT:
  if old_bull = '...'
     then subj = '_____________________________________________________'
     else subj = old_bull
  cron  = '____________'
  prof  = '________'
```

```
  fn    = '________'
  ft    = '________'
  flash = 'N'
  fullscreen = 'NO'
  'TOP'
  'DELETE *'
  'COMMAND ADD 17'
  'TOP'
return
SETUP:
  'COMMAND SET PFØ1 QQUIT'
  'COMMAND SET PFØ2 QQUIT'
  'COMMAND SET PFØ3 QQUIT'
  'COMMAND SET PFØ4 QQUIT'
  'COMMAND SET PFØ5 QQUIT'
  'COMMAND SET PFØ6 QQUIT'
  'COMMAND SET PFØ7 QQUIT'
  'COMMAND SET PFØ8 QQUIT'
  'COMMAND SET PFØ9 QQUIT'
  'COMMAND SET PF1Ø QQUIT'
  'COMMAND SET PF11 QQUIT'
  'COMMAND SET PF12 QQUIT'
  'COMMAND SET AUTOSAVE OFF'
  'COMMAND SET SHADOW OFF'
  'COMMAND SET FULLREAD ON'
  'COMMAND SET CASE M I'
  'COMMAND SET CMD OFF'
  'COMMAND SET SCALE OFF'
  'COMMAND SET MSGLINE ON 2'
  'COMMAND SET PRE OFF'
  'COMMAND SET CTLCHAR ! ESCAPE '
  'COMMAND SET CTLCHAR % PROTECT HIGH'
  'COMMAND SET CTLCHAR @ PROTECT NOHIGH'
  'COMMAND SET CTLCHAR " NOPROTECT HIGH'
  'COMMAND SET CTLCHAR * NOPROTECT NOHIGH'
  'COMMAND TOP'
return
EXIT:
  'QQUIT'
  exit
```

EBBFLASH EXEC

```
/*  System     : EBBNEWS                                                 */
/*  EXEC name  : EBBFLASH                                                */
/*  Invoked by : EBBSTART EXEC                                           */
/*  Function   : This EXEC sends news flashes to the bulletin board      */
/*               administrators for new bulletins.                       */
  hi = '1DE8'X
  lo = '1D6Ø'X
  blnk = lo || ' '
```

```
msg1 = hi || 'PROFS Bulletin Board News...' || lo
msg2 = lo || 'The following bulletins have been added or replaced:'
usr=USERID()
'DESBUF'
'EXECIO * DISKR EBBUSER AUTH A0 (FINIS ZONE 1 8 LOCATE /'usr'/'
if rc > 0
   then exit
if QUEUED() <> 2
   then exit
parse pull recnum; parse pull record
parse var record . uinit upswd .
'DESBUF'
'GETFMADR 200'
parse pull . user_mode user_vaddr .
'CP LINK' usr '191' user_vaddr 'RR' upswd
'ACC' user_vaddr user_mode
'EXECIO * DISKR $EBBNEWS $CONTROL' user_mode '(STEM' ctlrec.
'FINIS $EBBNEWS $CONTROL' user_mode
count = 0
i = 0
do ctlrec.0
   i = i + 1
   board = SUBSTR(ctlrec.i,1,8)
   bdate = SUBSTR(ctlrec.i,10,8)
   btime = SUBSTR(ctlrec.i,19,5)
   bcomp = bdate || btime
   'EXECIO * DISKR' board 'EBBNEWS A (STEM' item.
   j = 1
   do until j > item.0
      if j > 2
         then do
              idate = SUBSTR(item.j,101,8)
              itime = SUBSTR(item.j,75,5)
              icomp = idate || itime
              if icomp > bcomp
                 then do
                      if count = 0
                         then do
                              'EXECIO * CP (STRING MSGNOH' usr blnk
                              'EXECIO * CP (STRING MSGNOH' usr msg1
                              'EXECIO * CP (STRING MSGNOH' usr blnk
                              'EXECIO * CP (STRING MSGNOH' usr msg2
                              'EXECIO * CP (STRING MSGNOH' usr blnk
                              end
                      msg3 = SUBSTR(item.j,1,79)
                      'EXECIO * CP (STRING MSGNOH' usr msg3
                      count = count + 1
                      end
              end
      j = j + 1
   end
end
```

```
  'REL' user_vaddr '(DET'
  exit
```

EBBGOTO XEDIT

```
/*  System     : EBBNEWS                                                 */
/*  Macro name : EBBGOTO                                                 */
/*  Invoked by : EBBCURR XEDIT macro                                     */
/*  Function   : This macro lets the user "GO TO" a specific board       */
/*               from the list of all or new bulletins on all boards.    */
  'GLOBALV SELECT EBBNEWS GET naddr nmode'
EXTRACT:
  'COMMAND EXTRACT /CURSOR/LSCREEN/SIZE/LINE'
  cursscrn = CURSOR.1
  cursfile = CURSOR.3
  screenl  = LSCREEN.1
  filel    = SIZE.1
  saveline = LINE.1
CHECK:
  if cursscrn > screenl  then signal MSG
  if cursscrn < 2        then signal MSG
  if cursfile > filel    then signal MSG
  if cursfile < 1        then signal MSG
GETFN:
  'COMMAND :' cursfile
  'COMMAND EXTRACT /CURLINE'
  'COMMAND :' saveline
  strg = CURLINE.3
  if strg = ' ' then exit
  fn    = SUBSTR(strg,81,8)
  ft    = SUBSTR(strg,90,8)
  type  = SUBSTR(strg,99,1)
  board = SUBSTR(strg,112,8)
EXECUTE:
  'SET CMSTYPE HT'
  'STATE' board 'EBBNEWS' nmode
  saverc = rc
  'SET CMSTYPE RT'
  if saverc <> 0
     then do
          'COMMAND EMSG No Bulletin Board found for' board
          exit
          end
  'EXEC EBBBULL $OKAY$' board
  queue 'EMSG NOTE: Your display may not be current now.'
  exit
MSG:
  'COMMAND EMSG Place the cursor next to an item to GO TO',
               'its Board and press PF1'
  exit
```

EBBHELP EXEC

```
/*  System     : EBBNEWS                                                */
/*  EXEC name  : EBBHELP                                                */
/*  Rewritten from its original state to not use the CMS facility       */
/*                        (because it is too slow) and to provide       */
/*                        help screens specificly for each panel.       */
/*  Invoked by : EBBMENU, EBBBULL, EBBADD, EBBCREEN, and EBBCURR        */
/*               XEDIT macros                                           */
/*  Function   : This EXEC displays help screens for each bulletin      */
/*               board screen.                                          */
parse upper arg screen .
  hi = '1DE8'X
  lo = '1D6Ø'X
  'VMFCLEAR'
  say lo ' '
  say hi,
'Press the ENTER key to hold this screen.  Press CLEAR to return...   '
  say lo ' '
if screen = ''         then signal GENERAL
if screen = 'EBBMENU'  then signal GENERAL
if screen = 'EBBCREEN' then signal EBBCREEN
if screen = 'EBBBULL'  then signal EBBBULL
if screen = 'EBBCURR'  then signal EBBCURR
if screen = 'EBBEDIT'  then signal EBBEDIT
GENERAL:
  say hi 'The PROFS Bulletin Board',
      lo'is a set of programs that work with PROFS '
  say lo,
'to simulate  a real bulletin board.  PROFS documents, CMS files, and '
  say lo,
'"one liners"  can be placed  in one of six bulletin categories.      '
  say lo ' '
  say lo,
'From the  Bulletin Board menu  you may select a specific category by '
  say lo,
'pressing the  appropriate PF key.  To view all the bulletins  on all '
  say lo,
'of the boards  press the PF7 key.  To view  all the bulletins on all '
  say lo,
'the boards  that you  have not seen yet press the PF8 key.   You may '
  say lo,
'"screen" specific boards so that they are not shown when viewing all '
  say lo,
'or new  bulletins.  You may still go directly to a screened board to '
  say lo,
'view it.  To screen a  board from  the menu,  move the  cursor to  a '
  say lo,
'board  such as  "PF4  Vacations and OUT-OF-TOWN Notices"  and  press '
  say lo,
'PF11.  It will show  as ".....Screened".   To  un-screen the  board, '
  say lo,
```

```
'press the PF11 key again.'
  say lo ' '
  say hi 'Most PROFS users can view any item',
      lo 'on any of the boards.  However, '
  say lo,
'only selected persons have authority to add and delete items. If you '
  say lo,
'need add/delete ability then contact the Help Desk at ext. 154.      '
  say lo ' '
exit
EBBCREEN:
  say hi,
'You must move the cursor to a specific board before pressing PF11.   '
  say lo ' '
  say lo,
'The "Screen" function  allows you to shutdown  specific boards  so   '
  say lo,
'they are not shown if "PF7 ALL" or "PF8 NEW" are selected.  Boards   '
  say lo,
'which are "screened" in this fashion are still available if you go   '
  say lo,
'directly into the specific board  but will not be shown or printed   '
  say lo,
'when viewing "ALL" or "NEW" bulletins.                               '
  say lo ' '
  say lo,
'To screen a board,  move the cursor  to a specific board,  such as   '
  say lo,
'"PF4 Vacation and OUT-OF-TOWN notices",  and press  the PF11  key.   '
  say lo,
'Items placed on the board will no longer appear for "ALL" or "NEW"   '
  say lo,
'selections. The board can  be unscreened  by pressing the PF11 key   '
  say lo,
'again.  Screened boards will show as "......Screened" on the menu.   '
  say lo ' '
  say hi 'Please use this feature with care.',
      lo 'Only screen boards which you'
  say lo,
'do not care to see and which do not affect your daily work.          '
  say lo ' '
  exit
EBBBULL:
  say lo,
'You can move  the cursor  to any bulletin that has "more..." next to '
  say lo,
'the subject and  press PF4  or  the ENTER key  to view the reference '
  say lo,
'material for that item. The reference documents are displayed and/or '
  say lo,
'printed using PROFS facilities.                                      '
  say lo ' '
```

```
  say lo,
'The items on each specific bulletin board can be sorted by descending'
  say lo,
'date and time by pressing the PF7 key.  They can be sorted by author '
  say lo,
'by pressing PF8 (...handy if you are deleting your out-dated items). '
  say lo ' '
  say lo,
'To print an entire bulletin board or the list of all the boards then '
  say lo,
'press the PF6 key while the cursor is still at the top of the screen.'
  say lo,
'To print the reference document for an item  place the cursor on the '
  say lo,
'specific line before pressing PF6.'
  say lo ' '
  say lo,
'If you have add/delete authority to the bulletin boards then you can '
  say lo,
'press PF1  from any board  to add items.  Only selected persons have '
  say lo,
'authority to  add and delete items.  If you need  add/delete ability '
  say lo,
'then contact the Help Desk at ext. 154.                              '
  say lo ' '
  exit
EBBCURR:
  say lo,
'You can move  the cursor  to any bulletin that has "more..." next to '
  say lo,
'the subject and  press PF4  or  the ENTER key  to view the reference '
  say lo,
'material for that item. The reference documents are displayed and/or '
  say lo,
'printed using PROFS facilities.                                      '
  say lo ' '
  say lo,
'You may place the cursor  on a board heading or  on one of the items '
  say lo,
'and press PF1 to jump over to the specific board.                    '
  say lo ' '
  say lo,
'To print an entire bulletin board or the list of all the boards then '
  say lo,
'press the PF6 key while the cursor is still at the top of the screen.'
  say lo,
'To print the reference document for an item  place the cursor on the '
  say lo,
'specific line before pressing PF6.'
  say lo ' '
  say lo,
'If you have add/delete authority to the bulletin boards then you can '
```

```
  say lo,
'press PF1  from any board  to add items.  Only selected persons have '
  say lo,
'authority to  add and delete items.  If you need  add/delete ability '
  say lo,
'then contact the Help Desk at ext. 154.                              '
  say lo ' '
  exit
EBBEDIT:
  say hi,
'You have been established as a Bulletin Board administrator.',
      lo '  You'
  say lo,
'can add items  to the  boards  and delete any item you add.  You are '
  say lo,
'responsible for the contents of your bulletins and for deleting your '
  say lo,
'out-dated bulletin items.                                            '
  say lo ' '
  say lo,
'The only required data for a bulletin item  is the subject. You must '
  say lo,
'key the subject first.  Then if you have a  reference document which '
  say lo,
'you would like to  associate with the bulletin, type the appropriate '
  say lo,
'information such as a  document number  or filename and press PF1 to '
  say lo,
'add the bulletin item. You may press PF4 first, to view the document '
  say lo,
'before actually adding it to the bulletin board.                     '
  say lo ' '
  say lo,
'If you want to  elaborate  on an item  but  do not have a  reference '
  say lo,
'document, then  you may type  additional text  at the bottom  of the '
  say lo,
'screen.  The input area  at the bottom of the screen can be enlarged '
  say lo,
'by pressing PF6 for fullscreen mode. The screen mode can be switched '
  say lo,
'back by pressing PF6 again.  Lines can also be added by pressing PF2 '
  say lo,
'and/or deleted by pressing PF3.'
  say lo ' '
  exit
```

EBBHELP TEXT

```
.*
.* ...............FILE NAME = EBBNEWS SCRIPT
```

```
.*                                        last updated:  see FLIST date
.*
.* ..............general document format
:gdoc
.* ..............front matter begin
:frontm
.* ..............title page
:titlep
:title.Electronic News - PROFS Bulletin Board
:date.
:author.P. C. Shumway
:etitlep
.* ..............define heading levels 2 through 4 to indent in table
of contents
.dh 2 tcin 5
.dh 3 tcin 1Ø
.dh 4 tcin 15
.* ..............set page numbers off for table of contents
.pn offno
.* ..............generate table of contents
:tc
.* ..............set page numbers back on
.pn on
.* ..............start body of document
:body
.* ..............set format off - don't want DCF to format print
.fo off
.sv on
:h1 Introduction
:h2 About this manual
        This manual is stored as EBBNEWS SCRIPT and can be viewed and/or
    printed using the SCRIPT/VS facility.  The document you are now
    reading was printed on &date..
:h2 EBBNEWS Definition
         EBBNEWS (Electronic News) is a set of REXX programs and XEDIT
    macros that work with the PROFS system to provide an electronic
    bulletin board. This facility actually provides a SET of bulletin
    boards each of which stores bulletins belonging to the same category
    such as "Safety News" or "Plant Tours".  A bulletin can be a
    sentence or a reference to another document such as an inter-office
    memo created using the PROFS system or a CMS file containing text
    or diagrams.
:h1 System Features
    *    The PROFS Bulletin Board can be selected from a PROFS main menu
         using a PF key or the user can type "EBBNEWS" on a command line.

    *    This facility is menu driven with PF key selection processing.
         The same standard PF keys that are used in PROFS are used in the
         Bulletin Board screens such as PF9 for HELP and PF12 to RETURN.

    *    PROFS users can view all the bulletins on all the boards;  view
```

only those bulletins that they have not yet seen; or view and/or edit specific bulletin boards.

* When viewing either "ALL" or "NEW" bulletins the user can easily jump over to a specific board for editing.

* The items on each specific bulletin board can be sorted by date and time or by author.

* A bulletin can reference a PROFS draft document, a PROFS personal storage document, a CMS file or additional text that elaborates on the subject.

* To view reference material for a bulletin the user need only to move cursor to any bulletin that has "more..." shown next to the subject and press a PF key or the ENTER key.

* The reference documents are displayed and/or printed using the familiar document handling facilities of the PROFS system.

* To print an entire bulletin board or the list of all the boards then the user can press a PF key while the cursor is still at the top of the screen.

* To print the reference document for a bulletin the user need only to move the cursor to a specific line that has "more..." displayed and press a PF key.

* All PROFS users can view any item on any of the bulletin boards but only selected persons have authority to add and delete items.

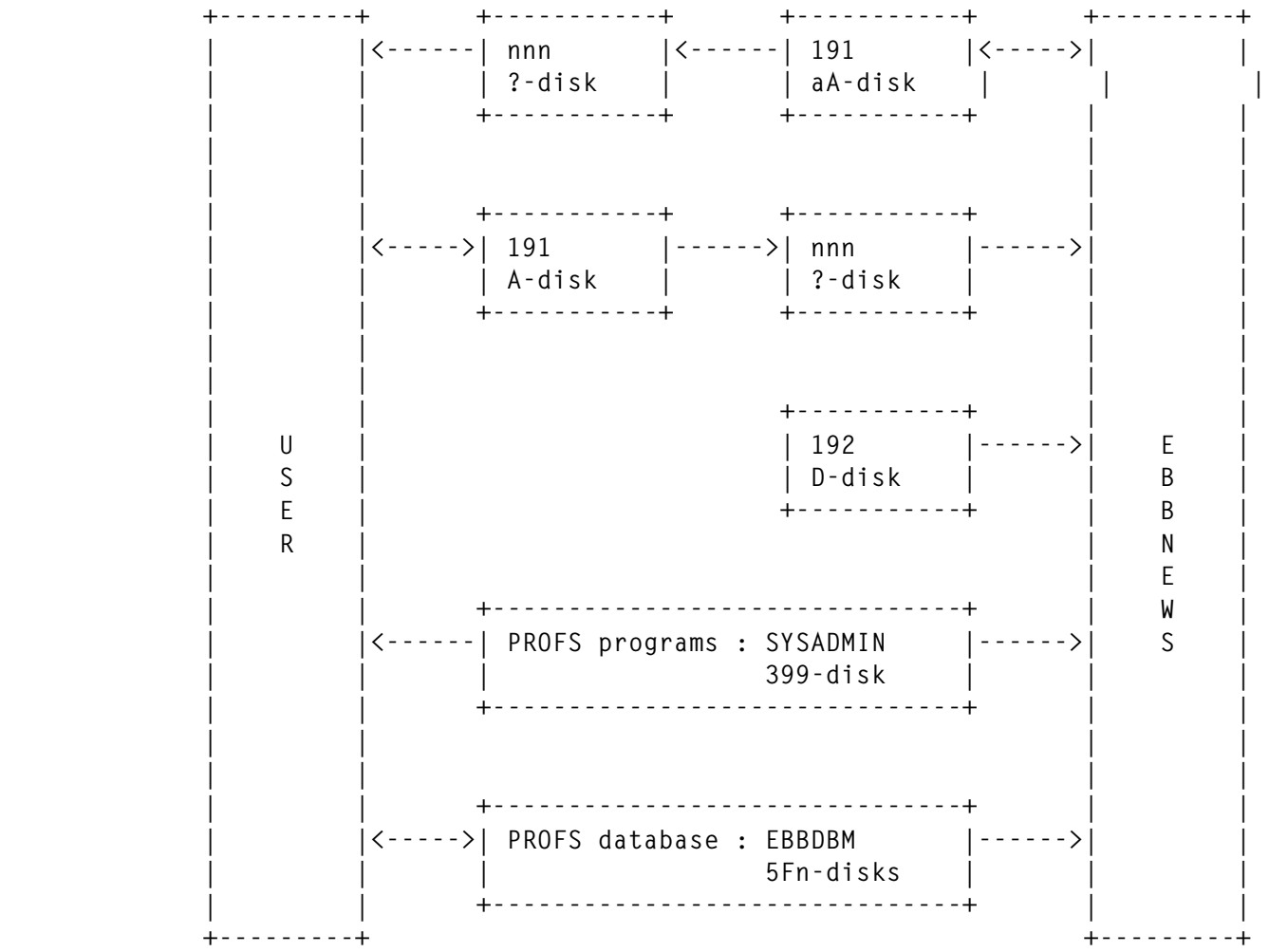
* Any user with add/delete authority to the bulletin boards can press a PF key from a specific board to invoke the ADD screen where the subject is supplied and a reference document can be specified or text typed "on-the-fly" to elaborate on the subject.

* There is no automatic deletion of bulletins. Those users who have add/delete ability must review and maintain their bulletins.
  To delete an item, the cursor is moved to the line on the screen and a PF key is pressed twice.

```
:h1 System Design
:h2 Mini-disk Structure
```

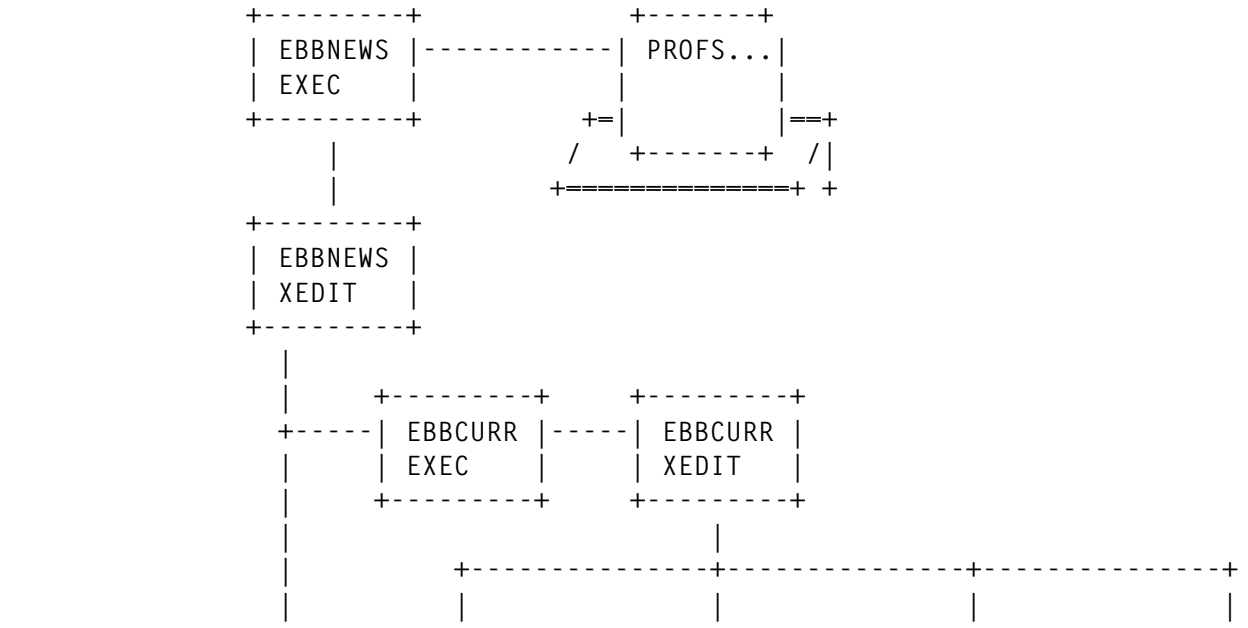
EBBNEWS is a disconnected virtual machine running CMS. The bulletin boards reside on EBBNEWS' 191 A disk. Reference documents for bulletins are also copied to this mini-disk for public viewing. The REXX EXECs and XEDIT macros that make up EBBNEWS actually reside on the PROFS 399 disk. The user authorization file resides on the DataCop 192 security disk.

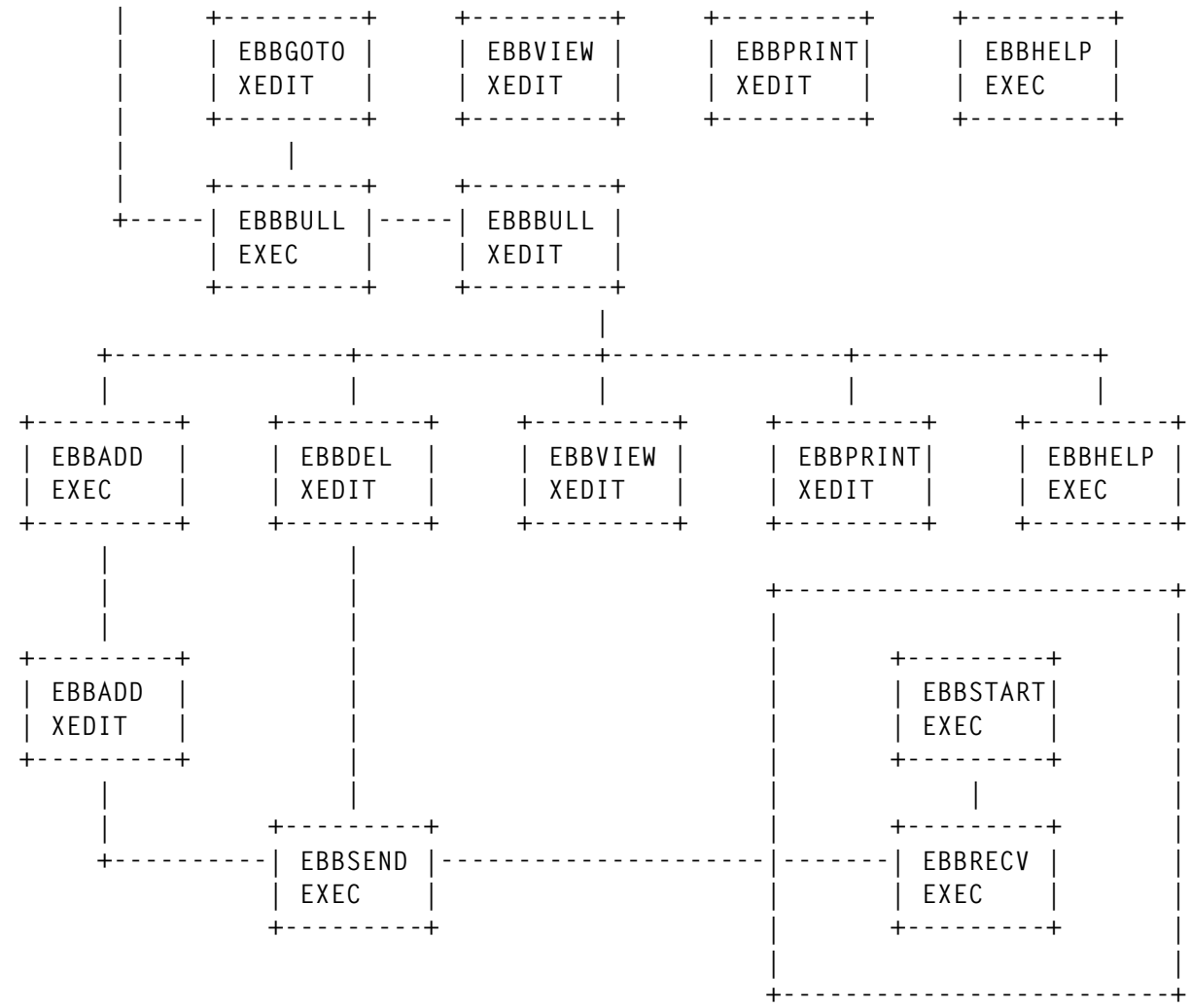
Here is an overview diagram of the EBBNEWS mini-disk structure:



.pa

:h2 Program Flow





.pa
:h2 Interfacing with PROFS

EBBNEWS calls upon PROFS to retrieve documents from the database if the user references a PROFS draft document when adding a bulletin. The PROFS RETRIEVE command is issued from the EBBADD XEDIT macro if a cron number is typed on the screen. The RETRIEVE command creates a script file on the user's A disk using the following naming scheme:

cron number : 87174EBB 0001

file number : D1740001 SCRIPT

All reference documents (including PROFS and non-PROFS documents) are copied to the EBBNEWS A disk as script files. A sequential number is assigned as a filename. The filename and filetype is then stored on the bulletin board record for that item. When the user elects to view or print the reference document for the bulletin, the filename and filetype is read from the bulletin record so the file can be located to be printed or displayed.

PROFS is called upon to display the reference documents stored

on the EBBNEWS A disk. If the reference document was created using PROFS then it will have DCF commands, GML tags, and other formatting data included with the text. The PROFS MEMO command is used to display PROFS-formatted documents while the PROFS OFSMOSCR command is issued to display any document that was created outside the PROFS system.

The PROFS HARDCOPY command is issued to print all the documents since they are stored on the EBBNEWS A disk as script files. This facility determines whether it is a PROFS formatted file or a simple script file for formatting the print. The "CHOOSE A PRINTER" screen is displayed so the user can select print options and route the output to a printer on the network.

```
.pa
:h2 Communications with EBBNEWS
```

Communications with the EBBNEWS machine must always be made using the EBBNEWS EXEC on SYSADMIN's 399 disk. The EBBNEWS EXEC links to the EBBNEWS virtual machine's 191 disk before displaying the menu. To view information on the bulletin boards the EBBNEWS mini-disk is re-accessed for current data. To add or delete items on the bulletin boards (which reside on EBBNEWS 191 disk) the EBBNEWS EXEC invokes the EBBSEND EXEC to communicate with the EBBNEWS virtual machine.

EBBNEWS utilizes the IPF WAKEUP command to control event driven VMCF communications with other (PROFS users') virtual machines. EBBNEWS issues the WAKEUP command to go into a wait state until an external event such as a VMCF message arrives.

The EBBSEND EXEC will also place the PROFS user's virtual machine in a wait state after sending add and delete requests to the EBBNEWS machine. EBBNEWS processes the request and sends back a message with a status code. When the message reaches the PROFS user's virtual machine it wakes up and EBBSEND reads the message and continues processing. This processing is completely transparent to the PROFS user.

As you can see, using the VMCF communications with EBBNEWS provides a single-threaded access mechanism for multiple users' virtual machines to share access to CMS files. If multiple requests arrive for EBBNEWS, they will wait in turn to be read and processed.

The messages are sent to and from EBBNEWS using the Special Message Facility's SMSG command. The 'SET SMSG ON' command is issued from the EBBNEWS's profile to enable the Special Message Facility before issuing
the WAKEUP command.

```
:h1 Operation and Maintenance
:h2 Startup/Shutdown
```

EBBNEWS does not need operator intervention to run. The EBBNEWS virtual machine starts up during a VM IPL through the AUTOLOG machine profile as are the other disconnected service

machines.

If for any reason one must start up EBBNEWS, one of two methods can be employed: simply log on to EBBNEWS and let the PROFILE EXEC execute (the EBBSTART EXEC will issue a WAKEUP command tell you to type '#CP DISC' and press ENTER), or issue an 'AUTOLOG' command to log EBBNEWS on disconnected.

To stop EBBNEWS's execution, log on EBBNEWS and press PA1 (for VM/READ mode) then press ENTER twice. If there is a need to shut-down EBBNEWS quickly then issue a 'FORCE' command to stop execution.

:h2 Adding or Deleting EBBNEWS users

The following procedures describe adding new users to EBBNEWS. Procedures for deleting users should become apparent.

* Users must first be defined in the VM directory and added to the PROFS UAD (User Authorization Directory). See chapter 3 in the "Managing PROFS" guide.

* Users must then be added to the EBBUSER LIST on the EBBNEWS 191 mini-disk. The EBBUSER AUTH file on DataCop's 192 security disk must also be updated. This file is a copy of the EBBUSER LIST file and includes read passwords to the users' 191 mini-disks.

NOTE : Make sure the initials of the users are unique.

:h1 Online Help

EBBNEWS has a summary of its functions available for on-line viewing by pressing the PF9 key from any EBBNEWS display.
It is included here:

The Electronic News Bulletin Board (EBBNEWS) is a set of on-line programs that work with PROFS to simulate a real bulletin board. The bulletins are grouped into categories such as "SAFETY NEWS" and "PLANT TOURS". When you select "PROFS Bulletin Board" from one of your main menus in PROFS you will be shown all the categories (bulletin boards) defined.

From the PROFS Bulletin Board screen select a specific bulletin board by pressing the appropriate PF key. To view all the bulletins on all the boards press the PF7 key. To view all the bulletins on all the boards that you have not seen yet press the PF8 key.

You can move the cursor to any bulletin that has "more..." next to the subject and press PF4 or the ENTER key to view the reference material for that item. The reference documents are displayed and/or printed using the PROFS facilities. If you are viewing either "ALL" or "NEW" bulletins then you can place the cursor next to a board heading or to one of the bulletins and press PF1 to jump over to the specific board.

The items on each specific bulletin board can be sorted by decending date and time by pressing the PF7 key. They can be sorted by author by pressing the PF8 key. You may find this to be a handy way to see

```
  who has posted which  bulletins or if you are  deleting your out-dated
  items.

  To print an entire  bulletin board or the list  of all the boards then
  press the PF6 key  while the cursor is still at the top of the screen.
  To print the  reference document  for an item  place the cursor on the
  specific line before pressing PF6.

  All PROFS users can  view any item on any of the bulletin boards.  But
  only selected persons  have authority to add and delete items.  If you
  need add/delete ability then contact the Help Desk at ext. 154.

  If you have  add/delete authority to the  bulletin boards then you can
  press PF1  from any bulletin board  to add  items.  The only  required
  data for a bulletin is the subject. You may also reference PROFS draft
  documents,  PROFS personal storage  documents,  or CMS files.   If you
  do not have a reference document  but want to elaborate on the subject
  type "Y" and any additional text at the bottom of the screen.

:h1 EXECs and XEDIT macros
     EBBNEWS programs reside on SYSADMIN's 399 mini-disk.  Printed
copies of these EXECs and XEDIT macros should be included in this
document after this page...
      EBBADD   EXEC
      EBBADD   XEDIT
      EBBBULL  EXEC
      EBBBULL  XEDIT
      EBBCURR  EXEC
      EBBCURR  XEDIT
      EBBDEL   XEDIT
      EBBDELX  XEDIT
      EBBGOTO  XEDIT
      EBBHELP  EXEC
      EBBMENU  XEDIT
      EBBNEWS  EXEC
      EBBNEWS  XEDIT
      EBBPRINT XEDIT
      EBBPRTX  XEDIT
      EBBRECV  EXEC
      EBBSEND  EXEC
      EBBSTART EXEC
      EBBVIEW  XEDIT
.* ..............end of general document
:egdoc
```

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

P C Shumway
Systems Analyst (USA)

VM utilities

The following are two useful VM utilities.

MODEFIND EXEC

There are many programs to find an available filemode to access a new disk/directory, but this one is definitely the shortest and simplest.

```
/* Find free filemode                                              */
/*                                    Vadim Rapp, 1987-97          */
call csl $DMSGETFM RC RRC FM$
if rc <> Ø then return -1
else return fm
```

XFLIST EXEC

XFLIST EXEC allows you to FLIST all the files satisfying a filter across all SFS directories. This is useful because, when working with SFS, duplicates tend to appear (versions of the same file in different directories), often it's hard to recall in which directory the particular file is stored, etc.

```
/* FLIST over directories */
arg fn ft fm
address command
$PIPE COMMAND L$ fn ft $(SEARCH | DROP 1 | STEM DIRS.$
if dirs.Ø = Ø then do; say $Nothing found$; exit 28; end
$ERASE $FLIST$ EXEC A$
j = Ø
$SET CMSTYPE HT$
do i=1 to dirs.Ø
  parse var dirs.i fni fti fmi dirf
  if fmi = $-$ then do    /* possible it is accessed on prev.. cycles*/
    $PIPE COMMAND LISTDIR$ dirf $| DROP 1 | SPEC 1 1 | VAR MODE$
    if mode = $-$ then do
      fmi = modefind()
        If fmi=-1 then do
           $SET CMSTYPE RT$
           $You want too much. Alphabet is not enough to access it all.$
           exit -1
        end
      j = j+1
      mode.j = fmi
```

```
      $ACCESS$ dirf fmi
    end
    else fmi = mode
  end
 diri = left(strip(right(dirf,21)),21)
  $PIPE COMMAND L$ fni fti fmi $(DATE | DROP 1 $ ,
  $   | SPEC 1-21 8$ ,
           $/$ || LEFT(right(diri,21),21) || $/ 29 57-7Ø 51$ ,
  $   | >> $FLIST$ EXEC A1 F$
end i
Mode.Ø = j
$SET CMSTYPE RT$
$FLIST (USE $FLIST$ MENU$
if mode.Ø<>Ø
  then $PIPE STEM MODE. | SPEC /REL / 1 1-* NEXT | COMMAND$
```

Vadim Rapp
Systems Officer
First Chicago NBD Corporation (USA)

CMS back-up/restore

There are six functions in the CMS back-up/restore system:

- BF – Back-up Full
- BL – Back-up Limited
- RD – Restore Direct
- RI – Restore Indirect
- LT – ListTape
- LU – ListUser.

When the options for an input field are listed, you need only enter the capital letters of the option, ie if you want the ListTape function just enter LT.

Abbreviations are also accepted, but the shortest abbreviation allowed is from the last capital letter in the option, eg shortest form for ListTape is LISTT.

Errors in a field will be displayed in reverse red. For an explanation of the error, place the cursor in the field and press PF1.

All messages from CMS back-up/restore are self-explanatory.

Only one tape drive (no alternatives) is supported.

ABOUT THE BACK-UP FUNCTION

There are two kinds of back-up – full and limited.

The full back-up takes a back-up of a complete volume.

The limited back-up uses a mix of INCLUDE/EXCLUDE files[1] and timestamps to select the mini-disks that will be backed up.

CMS back-up/restore uses the DDR (DASD Dump Restore) program to back-up the mini-disks. This makes it possible to restore stand-alone from a full back-up[2].

CMS back-up/restore maintains a timestamp for every mini-disk it knows.

These timestamps are kept in a CMS file (CMSBR COMPTBL) on the system's own A disk, and are used when you run a limited back-up to compare with the last update on a mini-disk. If there has been an update on the mini-disk, it will be selected for back-up.

Updating or erasing a file will cause CMS back-up/restore to select that mini-disk. If the last update was an erase, the file that was erased can be restored. See *Indirect restore*.

The timestamps will be updated after a successful back-up.

In case of an unsuccessful back-up, the timestamps will not be updated and you can rerun the back-up.

You can request that the timestamps are *not* updated. This can be used to make a temporary back-up.

Together with the update of the timestamps, an update of user-information is also done. The user-information is also kept in the CMSBR COMPTBL file and is used by the ListUser function.

You should run a full back-up of all the volumes, and limited back-ups on following days (they compare timestamps). The number of days

you run limited back-ups are up to you, but you must remember that:

- A full back-up and all the following limited back-ups (until the next full back-up) are one complete back-up.
- If you have a problem where all the data on a volume is lost, you must restore from a full back-up tape and run a special EXEC (BRRESALL), which will scan and restore from *all* the limited back-up tapes, that follow the full back-up tape[2].

Therefore the period between the full back-ups should not be too long.

Notes:

1 See *Explanation of the INCLUDE/EXCLUDE pairs* below.

2 See *Full DASD restore* below.

BACKUP FULL

Backup Full will back-up a full volume. All cylinders on the specified volume are backed up.

Input-fields

Volume id:

- Enter the volume-id of the volume you want to back up.
- Available volume-ids are displayed on the screen.

Sort back-up list by – this specifies how a list of the mini-disks that were backed up should to be sorted:

- User sorted on user-id.
- VolUser sorted on volume-id/user.
- VolStart sorted on volume-id/start cylinder.

If the field is blank, no list will be created. You can create the same list with the ListTape function. If you use ListTape, the volseq is not printed!

Update compare table – this specifies whether or not the update table will be updated. If you reply 'No', the timestamps will not be updated.

This could be used when you want a temporary back-up. The default is 'Yes'.

BACKUP LIMITED

Backup Limited will back-up selected mini-disks. It creates a back-up of the mini-disks that have been updated since they were last backed up.

Input-fields

The name of an INCLUDE/EXCLUDE pair. You can enter the name of an INCLUDE/EXCLUDE pair in this field. If only one of the files exists a warning is displayed, but the input is accepted. Not specifying a pair is also a valid input.

No matter what is entered in this field, the $ALWAYS$ pair will *always* be read.

Sort backup list by – see explanation in *Backup Full*.

Update compare table – see explanation in *Backup Full*.

Explanation of the INCLUDE/EXCLUDE pairs

The limited back-up uses (besides the timestamps) an INCLUDE/EXCLUDE pair which specifies what volumes and user-ids you wish to back-up.

The filename for these pairs is of your own choice but the filetypes must be INCLUDE and EXCLUDE and the filemode must be A.

There is no limit to how many pairs you can define, but you can only specify one pair when running a back-up.

The format of an INCLUDE/EXCLUDE pair is:

- First parameter – DASD or user-id. If you specify DASD the second parameter must be a volume-id.
- Second parameter – ALL, cuu, or volume-id. If the first parameter is DASD, this must be a volume-id. If the first parameter is a user-id you can specify 'ALL' if you want all the mini-disks belonging

to this user included/excluded, or you can specify the cuu of the mini-disk to be included/excluded.

- Third parameter – blank or FORCE. This parameter is only valid if the first parameter specifies a user-id. It specifies that you want to back-up this user's mini-disk(s) even if there are no updates on them.

All DASD are included by the system.

In the EXCLUDE file you can specify DASD ALL, which will exclude all volume-ids, and then in the INCLUDE file specify what volumes and/or user-ids you wish to back up.

There is a special INCLUDE/EXCLUDE pair ($ALWAYS$) which is always read by the system. Those volumes/user-ids defined in this pair will therefore always be included/excluded when you run a limited back-up. This pair can be used to specify different system-areas ($TEMP$, $TDISK$, ...) that you will have no use for backing up/restoring.

How the system interprets the INCLUDE/EXCLUDE pairs

Entries in the EXCLUDE file that specify volume-ids (and DASD ALL) are overridden by entries in the INCLUDE file that specify volume-ids.

Entries in the EXCLUDE file that specify user-ids are overridden by entries in the INCLUDE file that specify the same user-ids.

Entries that specify user-ids override all entries that specify volume-ids, irrespective of whether they are defined in the INCLUDE or EXCLUDE file.

Example:

You want to back up all the mini-disks on VMPK01 that have been updated, but only MAINT's 191 if there has been an update, and always MAINT's 198.

It is assumed that more of MAINT's mini-disks are on VMPK01.

The EXCLUDE file contains:

```
DASD ALL
MAINT ALL
```

The INCLUDE file contains:

```
DASD VMPKØ1
MAINT 191
MAINT 198 FORCE
```

From the defined INCLUDE/EXCLUDE pair the system will scan the chosen mini-disks to check whether they have been updated or not. If there has been an update (or FORCE has been specified) all the members on the mini-disk will be backed up.

See also the section about the CHECK option.

RESTORE DIRECT

Restore Direct will restore files directly to a specified mini-disk.

Any files on that disk will be erased and only the ones from the back-up will be available.

All files from the back-up will be restored.

Specify user-id/mdisk-cuu to be restored from (from-disk) and user-id/mdisk-cuu to be restored to (to-disk).

Because CMS back-up/restore will restore the entire disk directly, you must make sure that no one has access to the to-disk, and that the size of the from-disk equals the to-disk.

Input-fields

Userid to be restored from – enter the id of the user that owned the mini-disk when the back-up was taken.

Mdisk CUU to be restored from – enter the CUU that the mini-disk had when the back-up was taken.

Userid to be restored to – enter the id of the user that owns the mini-disk on which the restore must be placed.

Mdisk CUU to be restored to – enter the mini-disk CUU of where to place the restore.

Mdisk WRITE password – enter the WRITE password to the mini-disk where the restore must be placed.

RESTORE INDIRECT

With Restore Indirect, the restore is made to a temporary mini-disk and a list of files is displayed. You must then mark the files to be restored. The marked file(s) will be sent to the user-id you specify and must be RECEIVEd.

If the mini-disk being restored is so big that it's impossible to define a temporary mini-disk with the same size, you must use direct restore.

CMS back-up/restore shows all the files that are available for restore.

If the last (or only) update you do on a mini-disk before a back-up is erasing a file, that file will also appear on the list of files that can be restored.

See also *General information about the back-up functions.*

Input-fields

Userid to be restored from – enter the id of the user that owned the mini-disk when the back-up was taken.

Mdisk CUU to be restored from – enter the CUU that the mini-disk had when the back-up was taken.

Userid to be restored to – enter the id of the user that the selected files must be sent to. Default is the same as the mdisk owner.

Full DASD restore

In case of a problem where all the data on a volume is lost, you must restore from a *full* back-up tape.

This is done with DDR from a CMS user or stand-alone.

Use the following cards as input to DDR:

```
INPUT 181 TAPE (SKIP 3
OUTPUT cuu volser
SYSPRINT CONS
RESTORE ALL
```

If there are limited back-up tapes following the full back-up tape, you must 'update' the DASD. To do this:

- Log-on to CMS back-up/restore.
- Attach the DASD to CMS back-up/restore.
- Run BRRESALL.

BRRESALL first asks for the full back-up tape that was used to restore from, and will then ask for the limited back-up tapes that follow the full back-up tape.

BRRESALL restore 'backwards' from the limited back-up tapes, ie you must mount the *newest* limited back-up tape as the first tape to BRRESALL.

When BRRESALL requests a new tape, you must mount the tape that was created before the previous mounted tape.

BRRESALL will keep requesting new tapes until you terminate the program or until all cylinders have been updated (restored).

If BRRESALL is terminated abnormally, you must re-run BRRESALL and mount *all* the tapes again.

LISTTAPE

With the ListTape function you can list the users/mini-disks that are on the back-up tape(s).

This function creates the same list as you can request in the back-up function.

If there is more than one tape, it's only the first that has to be mounted.

Input-fields

Output device – select where you want the output: terminal or printer. If you choose terminal you can re-sort and/or print the list.

Sort output by – specifies how you want the list sorted:

- User sorted on user-id.
- VolUser sorted on volume-id/user.
- VolStart sorted on volume-id/start cylinder.

LISTUSER

The ListUser function can list information about the last two full and five limited back-ups that were taken for the specified user's mini-disk.

The function will display:

- Date and time of back-up.
- Volume-id of the DASD where the mini-disk resided.
- The label of the mini-disk.
- The VOL1 label from the first tape of the back-up.
- The tape volume-sequence-number (volseq).

No tape has to be mounted for this function.

Input-fields

Userid – enter a user-id.

Mdisk CUU – enter the mini-disk CUU of the mini-disk you want the information about. '?' will give you a list of CUUs that belong to the specified user-id.

Check option

The CHECK option is only meaningful for the back-up functions.

To run with the CHECK option, enter CMSBR CHECK, and select a back-up function.

Fill in the fields as required. The compare table will *not* be updated.

When the mini-disk scan is complete, the system exits and an output file (CMSBR CHECKLST) holds the result.

The CHECKLST lists all the scanned mini-disks and the ones that were selected are marked *SELECTED*. If you chose a full back-up, all mini-disks are marked *SELECTED*.

The CHECK option is a useful way to check the result of your INCLUDE/EXCLUDE definitions, or to see what mini-disks reside on a specific volume.

Only the CHECKLST file is created, so you do not need a tape drive when running with this option.

Batch mode

The CMS back-up/restore system also offers a way to run your back-ups in batch mode.

A BATCHRUN file specifies what kind of back-up you want and the input to that back-up.

You can specify as many back-ups as you want in one BATCHRUN file.

The filename for the BATCHRUN files are up to you, but the filetype must be BATCHRUN and the filemode A. There is no limit to how many BATCHRUN files you can define, but you can only specify *one* when you start the system in batch mode.

To start the system in batch mode enter CMSBR BATCH batchrun-filename.

The next section will describe the parameters that can be specified in the BATCHRUN files.

Parameters for BATCHRUN files are shown in Figure 1.

The PRINTLOG and ERASELOG cards are executed immediately when they are met in the input file. You must therefore be careful where and how you place them in the BATCHRUN file. All the other cards are read until a RUN card is met. When the RUN card is met the function begins.

An example of a BATCHRUN file is shown below:


```
R2 NEXTUDIR          GET DASD ADDRESS
```

FUNCTION	Backup Full	
	Backup Limited	No default - must be specified
ID	volumeid	Specify the volume for Backup Full.
	filename	Name of INCLUDE/EXCLUDE pair for Backup Limited. No default - must be specified
SORTLIST	Specifies how to sort the list of backed up mini-disks.	
	User	sorted on user-id.
	VolUser	sorted on volume-id/user
	VolStart	sorted on volume-id/start cylinder.
	(blank)	no list - default
UPDATE	Yes	The compare table is updated - default
	No	The compare table is NOT updated
TAPECUU	cuu	Indicates that the system must attach a tape drive. Must have status FREE. If another tape drive is attached as 181 it will be detached before the specified cuu is attached. Default: any tape drive attached as 181.
TAPEVOL	nnnnnn	If tape does not have a VOL1 label this will be used.
	Default: BATCHØ	
RUN	(no input)	Start the back-up.
PRINTLOG	(no input)	When running in batch mode a log file is written. This keyword specifies that it must be printed.
ERASELOG	(no input)	Erase (clean) the log file.

Figure 1: Parameters for BATCHRUN

```
ERASELOG               Clean log before run starts
FUNCTION   BF          Run a Backup Full
ID   VMSRES            Back up VMSRES
SORTLIST   USER        Sort the list on user-id
UPDATE   YES           Update compare table
TAPECUU   39Ø          Let CMS B/R attach 39Ø
TAPEVOL   112233       In case tape has no VOL1 label
RUN                    Start the back-up
FUNCTION   BL          We also need a Backup Limited
ID   DAILY             INCLUDE/EXCLUDE name
```

```
SORTLIST   USER         Sort the list on user-id
UPDATE   YES           Update compare table
TAPECUU   391          Now use 391 (390 is detached)
TAPEVOL   445566       Just in case
RUN                    Start the back-up
PRINTLOG               Let's see the log
ERASELOG               and clean it.
```

BRADPSW ASSEMBLE

```
*********************************************************************
** BRADPSW - Back-up/Restore Active Directory Pointer Switch       **
*********************************************************************
BRADPSW  CSECT
         REGEQU
         STM   R14,R12,12(R13)
         USING BRADPSW,R12
         LR    R3,R13
         LA    R13,SAVEAREA
         ST    R13,8(R3)
         ST    R3,SAVEAREA+4
         MVC   CUU,8(R1)
         PACK  PACKCUU(5),CUU(9)
         MVC   MODCCW,READCCW          SET MODCCW = READCCW
         BAL   R14,DIAG18              GO READ RECORD 3
         CH    R15,RC13
         BE    OUT
         XC    RC,RC
         CLC   INBUFF+12(4),=F'800'    800 BYTE BLOCK DISK
         BE    OUT                     YES, NOT SUPPORTED
         MVC   WRITECCW+5(3),INBUFF+13 LENGTH OF DISK BLOCK
         CLC   INBUFF+16(4),REC4       DOES ADP POINT TO REC 4
         BE    INSERT5                 YES, GO MAKE IT POINT TO REC 5
         MVC   INBUFF+16(4),REC4       ELSE LET IT POINT TO REC 4
         B     WRITENEW                GO WRITE IT
INSERT5  EQU   *
         MVC   INBUFF+16(4),REC5       SET ADP = REC 5
WRITENEW EQU   *
         MVC   MODCCW,WRITECCW         SET MODCCW = WRITECCW
         BAL   R14,DIAG18              GO WRITE REC 3
*********************************************************************
**                   E N D   O F   P R O G R A M                   **
*********************************************************************
OUT      EQU   *
         L     R13,SAVEAREA+4
         L     R14,12(R13)             RESTORE ADDRESSES
         LM    R0,R12,20(R13)          RESTORE ADDRESSES
         L     R15,RC                  SET RETURNCODE
         BR    R14
*********************************************************************
```

```
DIAG18   EQU   *
         LA    R15,1                   NUMBER OF READ/WRITES IN CCW
         L     R4,PACKCUU              R4 =  CUU
         LA    R5,CCWS                 R5 -> CCWS
         DIAG  R4,R5,X'0018'           DO DIAG X'18'
         ST    R15,RC                  SAVE RC
         BR    R14                     LET CALLER CHECK RC
*********************************************************************
**          C O N S T A N T S   A N D   V A R I A B L E S          **
*********************************************************************
SAVEAREA DS    18F
ONEPAGE  DC    F'4096'
RC       DC    F'0'
REC4     DC    F'4'
REC5     DC    F'5'
PACKCUU  DS    F,XL1
CUU      DS    CL8,XL1
*
RC13     DC    H'13'
*
         DS    0F
         DC    X'0000'
SEEK     DC    X'0000'
CYL      DC    X'0000'                 CYLINDER
TRACK    DC    X'0000'                 TRACK
RECORD   DC    X'03'                   RECORD
*
CCWS     CCW   X'07',SEEK,X'40',6
         CCW   X'23',RECORD,X'40',1
         CCW   X'31',CYL,X'40',5
         CCW   X'08',*-8,X'00',0
MODCCW   CCW   X'00',*,X'00',0         CCW IS MODIFIED DURING EXECUTION
*
READCCW  CCW   X'06',INBUFF,X'20',4096 X'20'=IGNORE INCORR. LENGTH
WRITECCW CCW   X'05',INBUFF,X'00',0    LENGTH IS SET DURING EXECUTION
*
         LTORG
         DS    0F
INBUFF   DS    CL4096
         END
```

BRDDR ASSEMBLE

```
*********************************************************************
** BRDDR - CMS Back-up/Restore run DDR program.                    **
*********************************************************************
BRDDR    CSECT
R0       EQU   0                       WORK
R1       EQU   1                       WORK
R2       EQU   2                       IPARML BASE AND WORK
R3       EQU   3                       WORK
```

```
R1Ø      EQU   1Ø                      TAPE CUU
R11      EQU   11                      CCW POINTER
R12      EQU   12                      PROGRAM BASE
R14      EQU   14                      BAL REGISTER AND RETURN ADDRESS
R15      EQU   15                      RC
         USING BRDDR,R12
         USING NUCON,RØ
         ST    R14,SAVERET
         CLM   R1,B'1ØØØ',=X'FF'       IS IT A NUCXDROP OR ABEND CALL?
         BE    NUCXDROP                YES, BRANCH
         CLI   INIT,C'N'               SHALL I RUN INIT
         BE    INITOK                  NO, BRANCH
         HNDIUCV SET,NAME=CMSBR,EXIT=CONNPEND
         LA    R2,IUCVLIST
         USING IPARML,R2
         IUCV  CONNECT,PRMLIST=(R2),USERID=SCIF,MF=L
         CMSIUCV CONNECT,NAME=CMSBR,PRMLIST=(R2),EXIT=IUCVINT
         MVC   PATHID,IPPATHID
         MVI   INIT,C'N'               DO NOT INIT NEXT TIME THROUGH
         B     OUT                     GO EXIT
         DROP  R2
INITOK   EQU   *
         LH    R1Ø,CUU                 LOAD TAPE CUU
         MVC   FUNCTION,8(R1)          GET THE FUNCTION
         CLC   BACKUP,FUNCTION         IS IT BACK-UP
         BE    STARTDDR                YES, GO START DDR
         CLC   RESTFULL,FUNCTION       IS IT RESTORE FROM A FULL BACK-UP
         BE    REWIND                  YES, GO REWIND TAPE
         MVC   VOLSER,16(R1)           ELSE GET VOLUME ID
         PACK  WORKD,24(4,R1)          AND START CYLINDER
         CVB   R2,WORKD                CONVERT
         ST    R2,CYLNO                ... AND STORE IT
         CLC   BRRESALL,FUNCTION       CALLED FROM EXEC BRRESALL
         BE    NOREW                   YES, DO NOT REWIND TAPE
REWIND   EQU   *
         MVI   TAPECMD,X'Ø7'           REWIND COMMAND
         LA    R11,TAPECMD             POINT TO CCW STRING
         BAL   R14,DIAG2Ø              GO DO DIAGNOSE
NOREW    EQU   *
         LA    R11,TAPEREAD            READ A RECORD
READLOOP EQU   *
         BAL   R14,DIAG2Ø              GO DO DIAGNOSE
         CLC   TAPEBUFF(4),=C'THR '    IS IT A TAPE HEADER
         BE    THR                     YES, GO CHECK IT
         CLC   TAPEBUFF(4),=C'VHR '    IS IT A VOLUME HEADER
         BE    VHR                     YES, GO CHECK IT
         CLC   TAPEBUFF(4),=C'EOV '    IT IT A END OF VOLUME
         BE    EOV                     YES, GO HANDLE END OF VOLUME
         B     READLOOP                GO READ ANOTHER RECORD
THR      EQU   *
         CLC   FUNCTION,RESTFULL       RESTORING FROM A FULL BACK-UP
```

```
         BE    GOBACK                   YES, GO BACK SPACE THE TAPE
         CLC   TAPEBUFF+12(2),CYLNO+2   CORRECT CYLINDER
         BNE   READLOOP                 NO, READ NEXT RECORD
         CLC   VOLSER,TAPVOL            CORRECT VOLUME
         BNE   READLOOP                 NO, READ NEXT RECORD
         B     GOBACK                   ELSE GO BACK SPACE THE TAPE
VHR      EQU   *
         CLC   FUNCTION,RESTFULL        RESTORING FROM A FULL BACK-UP
         BE    GOBACK                   YES, GO BACK SPACE THE TAPE
         MVC   TAPVOL,TAPEBUFF+30       MOVE TAPVOL
         B     READLOOP                 READ NEXT RECORD
EOV      EQU   *
         MVI   TAPECMD,X'0F'            REWIND/UNLOAD COMMAND
         LA    R11,TAPECMD              POINT TO CCW STRING
         BAL   R14,DIAG20               GO DO DIAGNOSE
         CLC   BRRESALL,FUNCTION        CALLED FROM EXEC BRRESALL
         BE    RESALL                   YES, DO NOT EXEC BRNEWTAP
         LA    R1,BRNEWTAP
         ICM   R1,B'1000',=X'00'
         SVC   202                      EXEC BRNEWTAPE
         DC    AL4(1)
         B     REWIND                   GO REWIND
RESALL   EQU   *
*        LA    R2,L'EOVTEXT
         LINEWRT DATA=(EOVTEXT,L'EOVTEXT)
         B     REWIND                   GO REWIND
*********************************************************************
GOBACK   EQU   *
         MVI   TAPECMD,X'27'            BACK SPACE THE TAPE SO IT IS ...
         LA    R11,TAPECMD              ... POSITIONED FOR DDR
         BAL   R14,DIAG20               GO DO DIAGNOSE
STARTDDR EQU   *
         LA    R1,DDRLIST
         ICM   R1,B'1000',=X'00'
         SVC   202                      EXECUTE DDR
         DC    AL4(1)
OUT      EQU   *
         L     R14,SAVERET              RESTORE RETURN ADDRESS
         BR    R14                      AND EXIT
*********************************************************************
NUCXDROP EQU   *
         HNDIUCV CLR,NAME=CMSBR         CLEAR IUCV TRAP
         B     OUT
*********************************************************************
CONNPEND EQU   *
         BR    R14
*********************************************************************
**                      S U B R O U T I N E S                      **
*********************************************************************
DIAG20   EQU   *
         XR    R15,R15                  CLEAR R15
```

```
         DIAG  R1Ø,R11,X'ØØ2Ø'          DO I/O
         BCR   1Ø,R14                   CC=Ø OR 2 - RETURN
         LA    R15,1ØØØ(,R15)           ADD 1ØØØ TO RC
         B     OUT                      AND EXIT PROGRAM
*************************************************************************
IUCVINT  EQU   *
         USING *,R12
         LR    R12,R15                  SET UP BASE
         XR    R15,R15                  CLEAR R15
         STM   RØ,R15,SAVEREGS          SAVE REGISTERS
         USING IPARML,R2
         CLI   IPTYPE,X'Ø2'             CONNECTION COMPLETE?
         BER   R14                      YES, JUST RETURN
         XC    IUCVLIST,IUCVLIST
         LA    R2,IUCVLIST
         LA    R3,255                   MAX BYTES TO BE RECEIVED
         IUCV  RECEIVE,PATHID=PATHID,PRMLIST=(R2),                      *
               BUFFER=IUCVINP,BUFLEN=(3)
         L     R1,IPBFLN1F              # OF BYTES NOT RECEIVED
         SR    R3,R1                    SUBTRACT FROM MAX BYTES
         SH    R3,=H'8'                 SUBTRACT ANOTHER 8 (USERID)
         FSWRITE 'CMSBR LOG A',BUFFER=IUCVINPX,BSIZE=(3),RECFM=V
         CLC   BACKUP,FUNCTION          RUNNING BACK-UP
         BNE   RETURN                   NO, GO TO RETURN
         CLC   ENDOFVOL,IUCVINPX        END OF VOLUME REACHED
         BNE   RETURN                   NO, GO TO RETURN
NEWTAPE  EQU   *
         LA    R1,BRNEWTAP
         ICM   R1,B'1ØØØ',=X'ØØ'
         SVC   2Ø2                      EXEC BRNEWTAP
         DC    AL4(1)
RETURN   EQU   *
         LM    RØ,R15,SAVEREGS          RESTORE REGISTERS
         BR    R14                      AND RETURN
*************************************************************************
WORKD    DS    D
IUCVLIST DC    XL4Ø'ØØ'
*
TAPEREAD CCW   X'Ø2',TAPEBUFF,X'2Ø',36 READ ONLY 36 BYTES
TAPECMD  CCW   X'Ø7',X'ØØØØØ1',X'6Ø',1 TAPE CMD - Ø7=REW  27=BSR
         CCW   X'Ø3',X'ØØØØØ1',X'6Ø',1 NOP
         CCW   X'Ø4',TAPEBUFF,X'2Ø',36 SENSE
*
SAVEREGS DS    16F
SAVERET  DS    F
CYLNO    DS    F
*
PATHID   DS    H
CONSADR  DS    H
CUU      DC    X'Ø181'
*
```

```
         DS    ØF
DDRLIST  DC    CL8'DDR'
         DC    CL8'INPUT'
         DC    CL8'DDR'
         DC    CL8'A1'
         DC    2F'-1'
*
BRNEWTAP DC    CL8'EXEC'
         DC    CL8'BRNEWTAP'
         DC    2F'-1'
*
INIT     DC    CL1'Y'
TAPVOL   DS    CL6
VOLSER   DS    CL6
SCIF     DC    CL8'*MSG'
CMSBR    DC    CL8'CMSBR'
FUNCTION DS    CL8
RESTFULL DC    CL8'RESTFULL'           RESTORE FROM A   FULL  BACK-UP
RESTLIM  DC    CL8'RESTLIM'            RESTORE FROM A LIMITED BACK-UP
BRRESALL DC    CL8'BRRESALL'           CALLED FROM BRRESALL
BACKUP   DC    CL8'BACKUP'             RUNNING BACK-UP
ENDOFVOL DC    C'END OF VOLUME'
EOVTEXT  DC    C'End of tape - Mount next tape'
TAPEBUFF DS    CL36
IUCVINP  DS    CL255
IUCVINPX EQU   IUCVINP+8
         LTORG
BRDDR    CSECT
         NUCON
         COPY  IPARML
         END
```

BRDIRSCN ASSEMBLE

```
*************************************************************************
** BRDIRSCN - CMS Back-up/Restore DIRectory SCaN                       **
** ------------------------------------------------------------------- **
** INPUT    - VOLSER OR "ALL"                                          **
** OUTPUT   - MDISKS THAT MATCH THE INPUT   (RETURNED IN THE STACK)  **
*************************************************************************
BRDIRSCN CSECT
RØ       EQU   Ø                       WORK
R1       EQU   1                       WORK
R2       EQU   2                       WORK
R3       EQU   3                       WORK
R4       EQU   4                       CUU
R5       EQU   5                       CCW POINTER
R8       EQU   8                       UDEVBLOK AND UMACBLOK BASE
R9       EQU   9                       UDIRBLOK BASE
R1Ø      EQU   1Ø                      PROGRAM BASE
R11      EQU   11                      PROGRAM BASE
```

```
R12      EQU   12                      PROGRAM BASE
R13      EQU   13                      SAVEAREA POINTER
R14      EQU   14                      BAL
R15      EQU   15                      RETURNCODE
         STM   R14,R12,12(R13)
         USING BRDIRSCN,R1Ø,R11,R12
         LR    R1Ø,R12
         LR    R11,R12
         A     R11,ONEPAGE
         A     R12,ONEPAGE
         A     R12,ONEPAGE
         LR    R3,R13
         LA    R13,SAVEAREA
         ST    R13,8(R3)
         ST    R3,SAVEAREA+4
         MVC   REQVSER(6),8(R1)
         L     R4,CUU
         LA    R5,RDDISK
         BAL   R14,READBLOK            GO DO FIRST READ (CCHHR = ØØØØ3)
         ICM   R2,B'1111',INBUFF1+52   DIRECTORY START ADDRESS
NEXTDIR  EQU   *
         BAL   R14,GETBLK1             GO GET THE BLOCK
         LA    R9,INBUFF1              DATA IN BUFFER 1 (ALWAYS)
         USING UDIRBLOK,R9
         LH    R1,UDIRDISP             GET DISPLACEMENT
         A     R1,=A(INBUFF1)          ADD BUFFER ADDRESS TO IT
         ST    R1,LASTSTRT             AND SAVE IT
         MVC   NEXTUDIR(4),UDIRDASD    REMEMBER NEXT UDIRBLOK
CHEKNDIR EQU   *
         C     R9,LASTSTRT             END OF DIRECTORY BLOCK
         BNL   UDIRCHEK                YES, GO CHECK FOR END OF DIR.
         A     R9,=A(UDIRSIZE*8)       MOVE POINTER
         TM    UDIRFLG1,UDIRPRF        IS IT A PROFILE
         BO    CHEKNDIR                YES, GO GET NEXT
         XC    UDIRUSER(8),MASK        UNMASK USERID
         LA    R2,UDIRBLOK
         MVC   STKLINE(8),UDIRUSER     AND MOVE IT
         MVC   DISP(2),UDIRDISP        SAVE DISPLACEMENT
         L     R2,UDIRDASD             GET DASD ADDRESS
         BAL   R14,GETBLKX             GO READ IT
         USING UMACBLOK,R8             R8 = USED BUFFER ADDRESS
         AH    R8,DISP                 ADD DISPLACEMENT
         L     R2,4(R8)                GET DASD ADDRESS
         LTR   R2,R2                   NO MORE
         BZ    CHEKNDIR                NO, GO GET NEXT USER
         MVC   DISP(2),2(R8)           SAVE DISPLACEMENT
         DROP  R8
```

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

Michael Plannthin (Denmark)

IBM has introduced its VisualAge 2000 offering, aimed at helping to make applications year 2000-compatible. It's a cross-platform, cross-language application development that combines IBM and non-IBM tools. The initial focus is on COBOL. The tools now available provide the ability to find the problematic dates automatically and change the system to interpret the date correctly. And users can choose to make the date application modifications on a workstation or on the platform of their choice.

For automating the process of finding the problematic date fields and fixing them, there's VisualAge for COBOL V1.2 on the OS/2 platform. To fix the problem on a workstation and then run the fixed code on a mainframe, there's the year 2000 cross-platform compiler, available on VM, OS/390, MVS, and VSE.

The new suite also offers a number of tools for testing and debugging, which include ISPF, the Debug Tool (compiler feature), and VisualAge for COBOL, Test.

For further information contact your local IBM representative.

* * *

Serena Software has announced that its COMPAREX software can now be used for year 2000 conversion testing. Comparex compares the contents of any two libraries' directories, files, or databases, and pinpoints any differences or similarities in a range of reports. As well as suporting VM/CMS, COMPAREX supports MVS, VSE, and OS/2.

For further information contact:
Serena Software International, 500 Airport Blvd, 2nd Floor, Burlingame, CA 94010-1904, USA
Tel: (415) 696 1800.

* * *

IBM has announced a new version of its RAMAC virtual storage array, promising up to twice the performance of the original system. The company also announced new capabilities for its Snapshot duplication software, so that RAMAC can work with nearly all System/390 data types.

Snapshot and the virtual array will be better integrated with the DFSMS storage management software by enhancing the Concurrent Copy function of DFSMS for easier use of Snapshot. Snapshot supports VM/ESA as well as OS/390, and makes use of RAMAC's virtual disk architecture.

The performance boost of the new RAMAC Virtual Array 2 Storage Turbo comes from a new shared-memory design. Also, the number of logical channels has increased from 32 to 128, allowing more System/390 processors to share data in Parallel Sysplex environments. This improvement will also be made available for all previous models of the Virtual Array.

IBM also announced the Scalable Array Storage 2 now uses Ultrastar 2XP 9.1GB drives. And addressable volumes have increased from from 256 to 512.

For further information contact your local IBM representative.