128 VM update

April 1997

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.

Displaying disk information

QDISK EXEC is a utility that displays information about your accessed and unaccessed CMS disks. It is intended as a replacement for the standard CMS QUERY DISK command, and has many useful features. Here are the main ones:

- QDISK memorizes your ‘normally accessed’ disks the first time it is run. This enables it to draw your attention to any disks that you do not usually have accessed (‘foreign’ disks). Information about your usually-accessed disks is saved locally onto your A disk. This means that each QDISK user can have his own personalized settings.
- QDISK uses colour to enable you to see instantly disks that are read/write and disks that you do not usually have accessed. Disk information is shown using blue and red to denote read-only and read/write status (respectively), and reverse video to draw your attention to foreign disks. For example, a standard system mini-disk accessed read-only would be displayed in blue, while someone else’s disk which you have accessed in write mode would be shown in reverse video red.
- QDISK cross-references your accessed disks with your linked disks, and shows those that are linked but not accessed. Full details are shown for each of these unaccessed disks, including disk label, size, status, and usage statistics. Empty disks and disks with errors (for example, unformatted ones) are all handled and reported on correctly. You would normally have to access the disks yourself to find out this information.

Before using QDISK to display disk information, you must initialize it by typing:

```
QDISK INIT
```

This causes the program to take a snapshot of the disks you currently have accessed, and save information about them to your A disk. These are then treated as your ‘usual’ disks; it is therefore best to use this option just after you have logged-on and before you have linked to any other users’ disks. QDISK INIT can be re-run at any time.

Thereafter, you can just type QDISK to display information about your disks. (It's even more convenient when assigned to a PF key.)

QDISK EXEC

```
/*====================================================================*/
/*                                                                    */
/* Enhanced QUERY DISK display, highlighting disks we don't           */
/* normally have accessed, and disks that we have linked/defined,     */
/* but have not accessed.                                             */
/*                                                                    */
/*====================================================================*/
address COMMAND
arg parm .
if parm = 'INIT' then
do
  call BUILD_DISK_LIST
  say 'Your currently accessed disks have been memorized.'
  say 'QDISK is now ready for use.'
  say
  exit Ø
end
/*--------------------------------------------------------------------*/
/* Check that our control file exists...                              */
/*--------------------------------------------------------------------*/
'STATE QDISK CONTROL A'
if rc <> Ø then
do
  say 'QDISK was unable to locate the QDISK CONTROL file.'
  say
  say 'Before using QDISK, you must prepare it for use by typing' ,
      '"QDISK INIT"'
  say
  exit 99
end
call READ_DISK_LIST                          /* Read the control file */
/*--------------------------------------------------------------------*/
/* Colours and highlighting.  Change these to whatever you want.      */
/*--------------------------------------------------------------------*/
heading_colour_1 = 'YELLOW UNDERLIN'    /* First section heading      */
heading_colour_2 = 'PINK   UNDERLIN'    /* Second section heading     */

my_rw_colour     = 'RED    NONE'        /* 'Normal' disk - read/write */
my_ro_colour     = 'BLUE   NONE'        /* 'Normal' disk - read/only  */

other_rw_colour  = 'RED    REVVIDEO'    /* Other disk - read/write    */
other_ro_colour  = 'BLUE   REVVIDEO'    /* Other disk - read/only     */

not_acc_colour   = 'TURQ   NONE'       /* Disks linkd but not accessed*/
```

```
blank_colour     = 'BLUE   NONE'        /* Colour used for spacing    */

/*--------------------------------------------------------------------*/
/* Save the current screen colour and highlight settings.             */
/*--------------------------------------------------------------------*/

'PIPE CP QUERY SCREEN | locate /VMOUT/ | var screen'

parse var screen . 'VMOUT' old_vmout_1 old_vmout_2 .
old_vmout = old_vmout_1 old_vmout_2

/*--------------------------------------------------------------------*/
/* Display the accessed disks.                                        */
/*--------------------------------------------------------------------*/

'VMFCLEAR'
'PIPE CMS QUERY DISK | stem disk_line.'
call COLOUR heading_colour_1
say copies(' ',79) || 'Ø1'x
accessed. = Ø
do i = 1 to disk_line.Ø
  parse var disk_line.i label cuu . mode .
  four_digit_cuu = right(cuu,4,'Ø')
  accessed.four_digit_cuu = 1            /* Flag for later DASD check */
  call DETERMINE_COLOUR
  call COLOUR colour
  disk_line.i = left(strip(disk_line.i),8Ø,'Ø1'x)
  say disk_line.i
end
/*--------------------------------------------------------------------*/
/* Show disks which are defined/linked but not accessed.              */
/*--------------------------------------------------------------------*/
call COLOUR blank_colour
say
'PIPE CP QUERY VIRTUAL DASD | stem dasd.'
shown_heading = Ø
do i = 1 to dasd.Ø
  parse var dasd.i . cuu .
  four_digit_cuu = right(cuu,4,'Ø')
  if accessed.four_digit_cuu then iterate   /* Accessed, so ignore it */
  if ¬shown_heading then
  do
    /*----------------------------------------------------------------*/
    /* Before processing the first unaccessed disk, obtain a          */
    /* temporary disk mode to use, and write the screen heading.      */
    /*----------------------------------------------------------------*/
    'GETFMADR'
    pull . fm . check .
    if check <> '' then
    do
      say 'There are no more free disk modes for QDISK to use!'
```

```
    say 'Unable to display the list of disks which are defined but',
        'not accessed.'
    leave
  end

  call COLOUR heading_colour_2
  say left('Disks that are not currently accessed:',79) || 'Ø1'x
  shown_heading = 1
  call COLOUR not_acc_colour
end

'SET CMSTYPE HT'

/*-----------------------------------------------------------------*/
/* NOTE:                                                           */
/*                                                                 */
/* Remove the "(MODEØ" option from the following line if your CMS  */
/* machine is not permitted to access file mode Ø files.  The line */
/* should read: 'PIPE command ACCESS' cuu fm '| var access_error'  */
/*-----------------------------------------------------------------*/

'PIPE command ACCESS' cuu fm '(MODEØ | var access_error'
arc = rc
select
  /*---------------------------------------------------------------*/
  /* Accessed okay, so extract disk info and release it...         */
  /*---------------------------------------------------------------*/

  when arc=Ø then
  do
    'PIPE CMS QUERY DISK' fm '|',
         'take last 1 |',
         'var disk_info'
    disk_info = overlay('-  ',disk_info,13)   /* Remove temp. fmode */
    'RELEASE' fm
  end

  /*---------------------------------------------------------------*/
  /* If we get RC=28 from the access, then the disk is R/O and     */
  /* contains no files.  In this case, we must use CP QUERY to     */
  /* get the disk information instead.                             */
  /*---------------------------------------------------------------*/

  when arc = 28 then
  do
    cuu = right(cuu,3,'Ø')
    'PIPE CP QUERY VIRTUAL' cuu '| var cpinfo'
    parse var cpinfo . . dev_type pack_id rw_status disk_size .

    dev_type  = strip(dev_type)
    pack_id   = strip(pack_id)
```

```
      disk_size = strip(disk_size)
      rw_status = strip(rw_status)

      disk_info = 'empty  ' || cuu || '  -   ' || ,
                  rw_status || right(disk_size,5) dev_type ,
                  '   -         Ø          Ø-ØØ' || ,
                  '          -          -'
      mode = rw_status
    end

    /*-------------------------------------------------------------------*/
    /* Other access error... show the error message in place of the      */
    /* disk information.                                                 */
    /*-------------------------------------------------------------------*/

    otherwise
    do
      cuu = right(cuu,3,'Ø')
      disk_info = '------ ' || cuu || '  -   Error:',
                  access_error
    end
  end
  'SET CMSTYPE RT'
  disk_info = left(strip(disk_info),8Ø,'Ø1'x)
  say disk_info
end
say
/*-----------------------------------------------------------------------*/
/* All done.  Restore screen colours and exit.                           */
/*-----------------------------------------------------------------------*/
call COLOUR old_vmout
exit Ø
/*-----------------------------------------------------------------------*/
/* This routine creates the QDISK CONTROL file on your A disk. The       */
/* file contains the labels of your own disks and disks which you        */
/* normally have accessed.                                               */
/*-----------------------------------------------------------------------*/

BUILD_DISK_LIST:
'PIPE CMS QUERY DISK |',
        'drop first 1 |',                   /* Ignore heading line  */
        'specs 1-6 1 |',                    /* Keep only the label  */
        '> QDISK CONTROL A fixed 6'         /* Write control file   */
if rc <> Ø then
do
  say 'Error occurred while creating QDISK CONTROL on your A disk.'
  say
  exit 99
end
return
/*-----------------------------------------------------------------------*/
```

```
/* Read the QDISK CONTROL file so we can identify our own disks.      */
/*--------------------------------------------------------------------*/

READ_DISK_LIST:
'PIPE < QDISK CONTROL A |',
        'join * / / |',
        'var normal_disks'
if rc <> Ø then
do
  say 'QDISK was unable to read file QDISK CONTROL from your A disk.'
  say 'Type "QDISK INIT" to build a new control file.'
  say
  exit 99
end
return
/*--------------------------------------------------------------------*/
/* Determine which colour to use to display a line of disk information*/
/*--------------------------------------------------------------------*/

DETERMINE_COLOUR:
if find(normal_disks,label) > Ø then normal = 1
                                else normal = Ø
if mode = 'R/W' then
do
  if  normal then colour = my_rw_colour
  if ¬normal then colour = other_rw_colour
end
if mode = 'R/O' then
do
  if  normal then colour = my_ro_colour
  if ¬normal then colour = other_ro_colour
end
return
/*--------------------------------------------------------------------*/
/* Subroutine to change screen colour and highlighting. 'CONWAIT' is  */
/* necessary to keep colour changes and text output in sync.          */
/*--------------------------------------------------------------------*/

COLOUR:
arg new_colour
if new_colour <> prev_colour then
do
  'CONWAIT'
  'EXECIO Ø CP (STRING SCREEN VMOUT' new_colour
  prev_colour = new_colour
end
return
```

Paul Harrison
Senior Systems Programmer (UK)

Events REXX EXEC

One very useful REXX EXEC that I have found is the 'Events' EXEC. This EXEC will display the events that are due for the next three days or as per your specifications. From now on you will never have to remember an event because this EXEC will always remind you of it, year after year, well ahead of time.

The EXEC can be run in VM as well as MVS. If you are running under VM, call this REXX EXEC as the last EXEC within your PROFILE EXEC. If you are running under MVS, suitably modify your LOGON procedure or type EVENTS in TSO.

This EXEC reads an 'event' file and then decides whether an event is due for display. An event file contains all your events. If the first character of a line within this file is not numeric then that line is considered to be a comment. Columns 1-8 contain the date due, in the format DDMMYYYY. If you ignore 'YYYY' then the event will be repeated every year. (Good for birthdays, renewals, getting new passwords, etc.) You can just give 'DD' and ignore 'MMYYYY'. This will repeat the event every month. Columns 10-12 are the number of days before the current date that you would like to see the event displayed. For example, every year you would like to display 'Renew *VM Update* subscription' at least 20 days before its expiry. You would set this field to 20. If columns 10-12 are blank or non-numeric, the default days – currently set to three – will be used.

An example of the event file and the corresponding display is shown below.

EVENT FILE

```
* ---------------- Events Log ---------------------- *
* Col  1-8   Date in DDMMYYYY Format                 *
* Col 1Ø-12  Reminder Days; Space=Default Days       *
* Col 14-53  Description Of The Events               *
* ------------------------------------------------- *
*                   Temporary                        *
* ------------------------------------------------- *
<-Date-> Dys <------------ Description ------------->
*                                                    *
```

```
Ø2Ø51996 1Ø  Move Xxxxxxx From Test To Prod
Ø3Ø41996     Install New Version Of Xxxxxxxx
31121996 1Ø  Apply For Holiday
12Ø11997 2Ø  Renew Passport
* ------------------------------------------------ *
*                   Permanent                      *
* ------------------------------------------------ *
<-Date-> Dys <------------ Description ------------>
*                                                  *
3Ø Ø1     3Ø  Renew VM Update Subscription
2Ø Ø2     15  Call ------ For Password
18Ø3     2Ø  Subscription Expires
Ø3Ø4         Call Sameena
Ø2Ø5     1Ø  Renew MVS Update Subscription
2812     2Ø  xxxxxx Birthday
```

For the events listed above, Figure 1 shows the events display that you will see when you log on.

```
*-----------------------------------------------------*
*       Events due for the next Three Days            *
*            Today's Date is 27/12/96                 *
*-------------+---------------------------------------*
* Date Due    |      Event                            *
*=============+=======================================*
* 31-Dec-1996 | Apply For Holiday                     *
*-------------+---------------------------------------*
* 12-Jan-1997 | Renew Passport                        *
*-------------+---------------------------------------*
* 28-Dec-1996 | xxxxxx Birthday                       *
*-----------------------------------------------------*
```

Figure 1: Event display

Please modify the filenames and the default number of days – defined as nDefDays and xDefDays – suitably. Don't define the nDefDays to be less than three – if you are working five days a week, you may miss some events! Also please ensure that xDefDays corresponds to nDefDays, ie the description of the number (eg three) and the number (eg 3) match.

EVENTS EXEC

```
/* ------------------------------------ */
/* REXX to display events on a particular  */
/* day                                  */
/* ------------------------------------ */
/* Naming conventions used:             */
/* REXX commands and functions have the    */
/*      first letter in upper case      */
/* TSO commands in upper case           */
/* User data (variables/constants/labels)  */
/* have a combination of upper & lower case*/
/*     and                              */
/* Alphanumeric user data starts with x    */
/* Numeric user data      starts with n    */
/* Switches / flags       start  with s    */
/* ------------------------------------ */
/* Author:- Moyeen Ahmed Khan           */
/* ------------------------------------ */
             /* ---------------------------------------- */
             /* Default number of days from which the      */
             /* event will be displayed. Modify suitably   */
             /* ---------------------------------------- */
nDefDays=3                                  /* Numerals */
xDefDays='Three'                            /*Character */

xWhereAmI=Address()
If xWhereAmI='TSO' | xWhereAmI='MVS' Then Call DfltsForMVS
  Else Call DfltsForVm

nJ=0
nDays.1=000
nDays.2=031
nDays.3=059
nDays.4=090
nDays.5=120
nDays.6=151
nDays.7=181
nDays.8=212
nDays.9=243
nDays.10=273
nDays.11=304
nDays.12=334

nMax.=31
nMax.2=28
nMax.4=30
nMax.6=30
nMax.9=30
nMax.11=30
```

```
sDueInd='N'

xMnthList='Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec'

nDaysInYr=Date(D)
nCrntYr=Substr(Date(S),1,4)

Call Years2Days nCrntYr
nDaysSoFor=nDaysInYr+nTotal

Do Forever
 Subcmd = 'R'
 'EXECIO 1 DISKR' xInpName
 Select
  When Rc=Ø Then Parse Pull xData
  When Rc=2 Then Leave
  Otherwise Do
   Say '*Error* Unable To Read ' xInput
   Say '        Read RC='Rc
   Exit
   End
  End
 Parse Var xData 1 nEventDD  3 nEventMM 5 nEventYY 9 nEventDays 14
xEventDetls
 If \Datatype(nEventDd,'W') Then Iterate
 If nEventDd=Ø              Then Iterate
 If \Datatype(nEventYy,'W') Then nEventYy=nCrntYr
 If \Datatype(nEventDays,'W') Then nEventDays=nDefDays
 If \Datatype(nEventMm,'W') Then nEventMm=nCrntMm
 If (nEventMm<1) | (nEventMm>12) Then nEventMm=nCrntMm
 nMm=Strip(nEventMm,L,'Ø')
 If nMm=2 & nEventYy//4=Ø Then Do
   nMax.2=29
   If nEventYy//1ØØ=Ø Then nMax.2=28
   If nEventYy//4ØØ=Ø Then nMax.2=29
   End
 If nEventDD>nMax.nMm Then nEventDD=nMax.nMm
 nColmn=((nMm-1)*4)+1
 xMnthName=Substr(xMnthList,nColmn,3)
 Call Years2Days nEventYy
 nDueDays=(nDays.nMm+nEventDd)+nTotal
 nLeapDay=Ø
 If nEventMm>2 & nEventYy//4=Ø Then Do
   nLeapDay=1
   If nEventYy//1ØØ=Ø Then nLeapDay=Ø
   If nEventYy//4ØØ=Ø Then nLeapDay=1
   End
 nDueDays=nDueDays+nLeapDay
 nDifDays=nDueDays-nDaysSoFor
```

```
 xEventPassed=(nDifDays<0)
 xEventNotDue=(nDifDays-nEventDays)>0
 Select
  When xEventPassed Then Iterate
  When xEventNotDue Then Iterate
  Otherwise Do
   sDueInd='Y'
   nJ=nJ+1
   xDispDd.nJ=nEventDd
   xDispMm.nJ=xMnthName
   xDispYy.nJ=nEventYy
   xDispEvent.nJ=Left(Strip(xEventDetls),40)
   End
  End
 End
'EXECIO * DISKR' xInpName '(FINIS'
If sDueInd='N' Then Do
  Exit                    /* If you want, you can call another EXEC, */
  End                     /*  in case there are no events to display */

If xWhereAmI='TSO' | xWhereAmI='MVS' Then Do
  xMsgst=Msg("OFF")
  'FREE DD(xINPUT)'
  xResetMsg=Msg(xMsgSt)
  'CLRSCRN'
  End
 Else Do
  'VMFCLEAR'
  End

xTitle.=''
xTitle.1='*-------------------------------------------------------*'
xTitle.2='*       Events due for the next' xDefDays 'Days'
xTitle.3="*            Today's Date is"  Date(e)
xTitle.4='*-------------+-----------------------------------------*'
xTitle.5='* Date Due    |      Event                               *'
xTitle.6='*=============+=========================================*'

xTitle.2=Overlay('*',xTitle.2,Length(xTitle.1))
xTitle.3=Overlay('*',xTitle.3,Length(xTitle.1))
nLength=Length(xTitle.3)

Say
Do nI=1 By 1 While xTitle.nI¬=''
 Say Center(xTitle.nI,78)
 End
Do nI=1 By 1 While nI<=nJ
 xEventDue='*' xDispDd.nI'-'xDispMm.nI'-'xDispYy.nI '|'  xDispEvent.nI
 xEventDue=Overlay('*',xEventDue,nLength)
 Say Center(xEventDue,78)
```

```
 If nI=nJ Then Say Center(xTitle.1,78)
   Else Say Center(xTitle.4,78)
 End
Exit

Years2Days:
Arg nValue
nBaseYr=1994
If (nValue-nBaseYr)<Ø Then Do
 Say '*Error* Events Program Does Not Support Events Before' nBaseYr
 Say 'Program Aborted'
 Exit
 End
nTotal=Ø
Do nI=nBaseYr By 1 While nI<nValue
  nTotal=nTotal+365
  If nI//4=Ø Then Do
                                  /* Divisible By 4, Leap Year */
    nTotal=nTotal+1
                                  /* Divisible By 1ØØ, No Leap Year */
    If nI//1ØØ=Ø Then nTotal=nTotal-1
                                  /* Divisible By 4ØØ, Leap Year */
    If nI//4ØØ=Ø Then nTotal=nTotal+1
   End
 End nI
Return

DfltsForVm:                     /* ---------- Note --------------- */
xFname='MYEVENTS'               /* Please modify To suit your stds */
xFtype='YEAR'                   /* ---------- Note --------------- */
xFmode='A'
xInput=xFname xFtype xFmode
Set Cmstype Ht
State xInput
If Rc¬=Ø Then Do
  Set Cmstype Rt
  Say '*Error* Your file' xInput 'does not exist'
  Say 'Rexx program aborted'
  Exit 9999
  End
Set Cmstype Rt
xInpName=xInput
Return

DfltsForMVS:
/* ------------------------------------------------------------- */
/*    Please modify to suit your stds                            */
/*  Your input can be a flat file or a PDS member                */
/* -----------------Examples------------------------------------ */
/* xInput='MYEVENTS.YEARLY.LOG'                                  */
```

```
/* xInput='MY.PDS.DATASET(EXAMPLE)'                                  */
/* ---------------------------------------------------------------- */
xInput='MYSF.PDS.DATASET(EXAMPLE)'      /* <---------- Modify Here */
xSetMsg=Msg('OFF')
xAvail=SYSDSN(xInput)
xMsg=Msg(xSetMsg)
If xAvail¬='OK' Then Do
  Say '*Error* Your file' xInput 'does not exist'
  Say 'REXX program aborted'
  Exit 9999
  End
xMsgst=Msg("OFF")
'FREE DD(xINPUT)'
'ALLOCATE DD(xINPUT) DSN('xInput') SHR'
nRcSave=Rc
xRstMsg=Msg(xMsgst)
If nRcSave¬=Ø Then Do
   Say '*Error* Unable To Alloc' xInput
   Say 'Alloc Return Code Is ' nRcSave
   Exit
   End
xInpName='XINPUT'
Return
```

Moyeen Ahmed Khan (Canada)

XEDIT extensions – continued

This month we continue with the XEDIT extensions that are designed mainly for applications programmers and for users who have any XEDIT execution experience. The product provides a powerful set of additional editing functions, which considerably improve programmer productivity in the XEDIT environment.

XEPF12 XEDIT

```
/*********************************************************************/
/***                                                        ***    ***/
/*** XEPF12         XEDIT  Extensions                        ***    ***/
/***                                                        ***    ***/
/*********************************************************************/
/***   SIZE 00013  VER 1.0 MOD 00  TIME 16:01:52                    ***/
/*********************************************************************/
  EXT '/CURS'
  IF CURSOR.1 = 24 THEN
  MACRO XEBLK
  ELSE
  CUR S 12 40
```

XEUNDO XEDIT

```
/*********************************************************************/
/***                                                        ***    ***/
/*** XEUNDO         XEDIT  Extensions                        ***    ***/
/***                                                        ***    ***/
/*********************************************************************/
/***   SIZE 00022  VER 1.0 MOD 00  TIME 16:07:07                    ***/
/*********************************************************************/
  EXT '/COL/CURS/LI/LR'
  IF CURSOR.3 < 0 THEN
  EXIT
  ADDRESS CMS GLOBALV SELECT XEDIT GET OLD_LINE OLD_LINE_SET OLD_NUM
  IF OLD_LINE_SET ¬= 'Y' THEN
  EXIT
  OLD_LINE_SET = 'N'
  ':'OLD_NUM
  CL ':1'
  CD LRECL.1
  IF OLD_LINE ¬= ' ' THEN
  CR OLD_LINE
  ':'LINE.1
  CL COLUMN.1
```

XEBLK XEDIT

```
/*********************************************************************/
/***                                                          ***  ***/
/*** XEBLK          XEDIT  Extensions                         ***  ***/
/***                                                          ***  ***/
/*********************************************************************/
/***   SIZE ØØ254  VER 1.Ø MOD ØØ  TIME 14:37:39              ***  ***/
/*********************************************************************/
  CMDLINE      OFF
  SCALE        ON 2
  CTLCHAR '['  ESCAPE
  CTLCHAR '\'  PROTECT
  CTLCHAR ']'  NOPROTECT
  EXT'/FN/FT/FM/RECF/LR/TR/W'
  EXT'/PF1/PF2/PF3/PF4/PF5/PF6/PF7/PF8/PF9/PF1Ø/PF11/PF12'
  DO I_N = 1 TO 12
    PF || I_N   ONLY X
  END
  B_MRK = Ø
  E_MRK = Ø
  SWITCH = 1
  ROW = 12
  POS = 7
PROC_MARK:
  RESERVED 24 HI '>>>'                                              ,
  '1 Mrk 2 Umrk 3 Esc 4 Top 5 Bot 6 CurL 7 Ba 8 Fo 9 CurR'          ,
  '1Ø Le 11 Ri 12 Proc'
  DO FOREVER
    CUR S ROW POS
    READ N N T
    PULL TAG PFK B C D E
    EXT'/CURS/LI'
    ROW = CURSOR.1
    POS = CURSOR.2
    IF TAG = 'PRF' THEN
    ':'C
    ELSE
    IF TAG = 'PFK' THEN
    DO
      IF PFK < 4 THEN
      DO
        IF PFK = 1 & CURSOR.3 > Ø THEN
        DO
          IF B_MRK = Ø THEN
          DO
            ':'CURSOR.3 - 1
            B_MRK = CURSOR.3
            B_COL = CURSOR.4
            I COPIES('=', B_COL - 1)'|>'
```

```
  END
  ELSE
  IF E_MRK = Ø & ABS(B_MRK - CURSOR.3 - 1) > 1                ,
         & CURSOR.4 > 1 & ABS(CURSOR.4 - B_COL) > 1 THEN
  DO
    ':'CURSOR.3
    E_MRK = CURSOR.3 + 1
    E_COL = CURSOR.4
    I '1'
  END
  IF B_MRK > Ø & E_MRK > Ø THEN
  DO
    IF B_MRK > E_MRK THEN
    DO
      TEMP = B_MRK + 1
      B_MRK = E_MRK
      E_MRK = TEMP
    END
    IF B_COL > E_COL THEN
    DO
      TEMP = B_COL
      B_COL = E_COL
      E_COL = TEMP
    END
    ':'B_MRK
    CL ':'1
    CD TRUNC.1
    CR COPIES('=', B_COL - 1)'|>'
    CL ':'E_COL
    CR '|'COPIES('=', TRUNC.1 - E_COL)
    ':'E_MRK
    CL ':'1
    CD TRUNC.1
    CR COPIES('=', B_COL - 1)'|'
    CL ':'E_COL - 1
    CR '<|'COPIES('=', TRUNC.1 - E_COL)
  END
END
ELSE
DO
  IF B_MRK > Ø THEN
  DO
    ':'B_MRK
    DEL
    B_MRK = Ø
  END
  IF E_MRK > Ø THEN
  DO
    ':'E_MRK - 1
    DEL
```

```
          E_MRK = Ø
        END
        IF PFK = 3 THEN
        SIGNAL EXIT
      END
      ':'LINE.1
      ITERATE
    END
    IF PFK = 4 THEN
    DO
      IF SWITCH < Ø THEN
       ':'MAX(1, B_MRK)
      ELSE
      TOP
      SWITCH = - SWITCH
    END
    IF PFK = 5 THEN
    DO
      IF SWITCH < Ø THEN
      ':'MAX(1, E_MRK)
      ELSE
      BOT
      SWITCH = - SWITCH
    END
    IF PFK = 6 THEN
    DO
      POS = POS - 1Ø
      IF POS < 1 THEN
      POS = 8Ø
    END
    IF PFK = 7 THEN
    '-2Ø'
    IF PFK = 8 THEN
    2Ø
    IF PFK = 9 THEN
    DO
      POS = POS + 1Ø
      IF POS > 8Ø THEN
      POS = 1
    END
    IF PFK = 1Ø THEN
    LE 4Ø
    IF PFK = 11 THEN
    RI 4Ø
    IF PFK = 12 THEN
    LEAVE
  END
END
CURS S 12 4Ø
DO FOREVER
```

```
RESERVED 24 HI                                                          ,
'1 InsA  2 ExcA  3 Esc  4 CpyR  5 MveR  6 CpyI 7 MveI'                  ,
'8 Del 9 Ins 1Ø Exc 11 Fil'
READ N T
PULL TAG PFK B C D E
IF TAG = 'PFK' THEN
DO
  IF PFK = 3 THEN
  LEAVE
  EXT'/CURS'
  IF PFK < 3 THEN
    IF B_MRK = Ø THEN
  LEAVE
  ELSE
  DO
    ':'B_MRK
    DEL
    IF E_MRK > Ø THEN
    DO
      ':'E_MRK - 1
      DEL
    END
    EXT'/SIZ'
    B_MRK = Ø
    E_MRK = SIZE.1 + 1
  END
  IF PFK > 3 & PFK < 12 THEN
    IF B_MRK = Ø & E_MRK = Ø THEN
  LEAVE
  IF PFK > 3 & PFK < 6 THEN
    IF CURSOR.4 >= B_COL & CURSOR.4 <= E_COL THEN
      IF CURSOR.3 >= B_MRK & CURSOR.3 <= E_MRK THEN
  LEAVE
  IF PFK > 9 THEN
  DO
    IF PFK = 1Ø THEN
    PFK_CH = 'A'
    IF PFK = 11 THEN
    PFK_CH = 'B'
  END
  ELSE
  PFK_CH = PFK
  IF PFK < 12 THEN
  DO
    REQ = TRANSLATE(PFK_CH, 'IECMCMDIEF', '12456789AB')
    IF ¬ (REQ = 'C' | REQ = 'M' | REQ = 'D') THEN
    DO
      IF REQ = 'F' THEN
      MSG = COPIES(' ', 9)                                              ,
      'Enter char -> []_[\   Enter to proc  PF12 Esc'
```

```
          ELSE
          MSG =                                                          ,
          'Enter number of cols -> []____[\   Enter to proc  PF12 Esc'
          RESERVED 24 HI MSG
          DO FOREVER
            CUR S 24 26
            READ N T
            PULL TAG PFK_IN B C D E
            IF TAG = 'PFK' & PFK_IN = 12 THEN
            SIGNAL PROC_MARK
            IF TAG = 'ETK' & QUEUED() > Ø THEN
            DO
              PULL . . . CHAR
              CHAR = SUBSTR(CHAR, 1, MAX(1, INDEX(CHAR, '_') - 1))
              IF REQ = 'F' THEN
              LEAVE
              IF DATATYPE(CHAR, 'N') THEN
              DO
                E_COL = MIN(CHAR, WIDTH.1 - B_COL + 1) + B_COL - 1
                LEAVE
              END
            END
          END
        END
        IF PFK > 3 & PFK < 6 THEN
        CHAR = 'R'
        IF PFK > 5 & PFK < 8 THEN
        CHAR = 'I'
        RECF F
        LRE WIDTH.1
        TR '*'
        XEHMA FNAME.1 FTYPE.1 FMODE.1 RIGHT(WIDTH.1, 6, 'Ø') REQ CHAR,
              RIGHT(B_MRK, 6, 'Ø') RIGHT(E_MRK, 6, 'Ø')               ,
              RIGHT(B_COL, 6, 'Ø') RIGHT(E_COL, 6, 'Ø')               ,
              RIGHT(CURSOR.3, 6, 'Ø') RIGHT(CURSOR.4, 6, 'Ø')
        RECF RECFM.1
        LRE LRECL.1
        TR TRUNC.1
        IF PFK < 3 THEN
        DO
          B_MRK = Ø
          E_MRK = Ø
          LEAVE
        END
      END
    END
  END
  ':'LINE.1
  SIGNAL PROC_MARK
EXIT:
```

```
  CMDLINE      BOT
  SCALE        OFF
  DO I_N = 1 TO 12
    INTERPRET PF || I_N PF || I_N'.1' PF || I_N'.2'
  END
```

XECMS XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** XECMS          XEDIT  Extensions                    ***         ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE 00017  VER 1.0 MOD 00  TIME 15:59:03                    ***/
/*********************************************************************/
  ADDRESS CMS
  QUERY PF03 '(' STACK
  PARSE PULL SET_PF03
  MAKEBUF
  QUERY TERM '(' STACK
  PULL . LINEND .
  DROPBUF
  SET PF03 IMM SET SET_PF03 LINEND RETURN
  ADDRESS XEDIT CMS
```

XE1011ST XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** XE1011ST       XEDIT  Extensions                    ***         ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE 00014  VER 1.0 MOD 00  TIME 15:57:34                    ***/
/*********************************************************************/
  PF10 LE 40
  PF11 RI 40
  PREF OFF
  TR '*'
  RIGHT 1
  V OFF 1 73
```

XE1011RS XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** XE1011RS       XEDIT  Extensions                    ***         ***/
/***                                                    ***         ***/
/*********************************************************************/
```

```
/***   SIZE 00012  VER 1.0 MOD 00  TIME 15:56:47                  ***/
/*********************************************************************/
  PF10 ONLY MACRO XEPF10
  PF11 ONLY MACRO XEPF11
  PREF NULL
  V OFF 1 73
```

XEHLP XEDIT

```
/*********************************************************************/
/***                                                    ***         ***/
/*** XEHLP          XEDIT  Extensions                    ***         ***/
/***                                                    ***         ***/
/*********************************************************************/
/***   SIZE 00026  VER 1.0 MOD 00  TIME 16:25:12                  ***/
/*********************************************************************/
  DO I = 1 TO 12
    PF || I ONLY QQUIT
  END
  ENT  ONLY    QQUIT
  CMDLINE      OFF
  CURLINE      ON  3
  CURS         S 24 80
  MSGLINE      OFF
  NUMB         OFF
  PREFIX       OFF
  SCALE        OFF
  SERIAL       OFF
  STAY         ON
  TOFEOF       OFF
  RESER 1 HI '>>> XEDIT/E Help <<<'COPIES(' ', 28)                   ,
                                          '***  XEDIT  Extensions  ***'
  RESER 2 HI COPIES(' ', 48)'*****  Ver 1.0 (C) DG''95  *****'
  TOP
```

XEHMA ASSEMBLE

```
**********************************************************************
****                                                    ***         ****
**** XEHMA          XEDIT  Extensions                    ***         ****
****                                                    ***         ****
**********************************************************************
****   SIZE 00237  VER 1.0 MOD 00  TIME 13:59:30                  ****
**********************************************************************
*                                                                    *
XEHMA    CSECT
         USING *,12
         ST    14,RG14
```

```
         MVC   FSCBFN(8),8(1)
         MVC   FSCBFT(8),16(1)
         MVC   FSCBFM(2),24(1)
         PACK  DOUBLE(8),32(6,1)
         CVB   11,DOUBLE
         ST    11,FSCBSIZE
         MVC   OPER,40(1)
         MVC   FILLIN,48(1)
         PACK  DOUBLE(8),56(6,1)
         CVB   3,DOUBLE
         LA    3,1(3)
         PACK  DOUBLE(8),64(6,1)
         CVB   2,DOUBLE
         SR    2,3
         PACK  DOUBLE(8),72(6,1)
         CVB   4,DOUBLE
         PACK  DOUBLE(8),80(6,1)
         CVB   5,DOUBLE
         SR    5,4
         LA    5,1(5)
         MVI   PROCTARG,C'N'
         CLI   OPER,C'C'
         BE    SETFLG
         CLI   OPER,C'M'
         BNE   GETMEM
SETFLG   EQU   *
         MVI   PROCTARG,C'Y'
         PACK  DOUBLE(8),88(6,1)
         CVB   6,DOUBLE
         PACK  DOUBLE(8),96(6,1)
         CVB   7,DOUBLE
GETMEM   EQU   *
         DMSFREE DWORDS=12000
         LR    8,1
         LA    9,4000
         SLL   9,3
         LR    10,9
         AR    9,8
         AR    10,9
         CLI   OPER,C'I'
         BNE   GET
         LR    9,8
         LR    7,4
GET      EQU   *
         MVC   FSCBCOMM(8),=CL8'DMSXFLRD'
         MVI   PROCORIG,C'N'
         ST    3,FSCBAITN
         ST    8,FSCBBUFF
         LA    0,EXTPLIST
         LA    1,FSCB
```

```
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
         CLI   PROCTARG,C'Y'
         BNE   SKIPTARG
         ST    6,FSCBAITN
         ST    9,FSCBBUFF
         LA    Ø,EXTPLIST
         LA    1,FSCB
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
SKIPTARG EQU   *
         CLI   OPER,C'C'
         BE    COPY
         CLI   OPER,C'M'
         BNE   CHECKD
COPY     EQU   *
         CLI   FILLIN,C'R'
         BE    REPLACE
INSERT   EQU   *
         LR    Ø,9
         LR    1,7
         LR    14,1Ø
         LR    15,1
         MVCL  14,Ø
         LR    1,11
         SR    1,5
         LR    Ø,9
         AR    Ø,7
         BCTR  Ø,Ø
         LR    14,1Ø
         AR    14,7
         AR    14,5
         BCTR  14,Ø
         LR    15,1
         MVCL  14,Ø
         LR    14,1Ø
         CLI   OPER,C'I'
         BNE   COPYMOVE
         LR    Ø,1Ø
         SR    1,1
         ICM   1,8,=X'4Ø'
         B     ERASE
REPLACE  EQU   *
         LR    14,9
COPYMOVE EQU   *
         LR    Ø,8
         AR    Ø,4
         BCTR  Ø,Ø
```

```
         LR    1,5
         AR    14,7
         BCTR  14,Ø
         LR    15,1
         MVCL  14,Ø
         CLI   OPER,C'C'
         BE    PUT
EXCLUDE  EQU   *
         MVI   PROCORIG,C'Y'
         LR    Ø,8
         AR    Ø,4
         LR    14,Ø
         AR    Ø,5
         LR    1,11
         SR    1,4
         SR    1,5
         BCTR  Ø,Ø
         BCTR  14,Ø
         LR    15,1
         MVCL  14,Ø
         LR    Ø,8
         AR    Ø,11
         SR    Ø,5
         SR    1,1
         ICM   1,8,=X'4Ø'
         LR    14,Ø
         LR    15,5
         MVCL  14,Ø
         B     PUT
CHECKD   EQU   *
         MVI   PROCORIG,C'Y'
         SR    1,1
         CLI   OPER,C'D'
         BNE   CHECKF
         ICM   1,8,=X'4Ø'
         B     DELETE
CHECKF   EQU   *
         CLI   OPER,C'F'
         BNE   CHECKI
         ICM   1,8,FILLIN
DELETE   EQU   *
         LR    Ø,8
ERASE    EQU   *
         AR    Ø,4
         BCTR  Ø,Ø
         LR    14,Ø
         LR    15,5
         MVCL  14,Ø
         B     PUT
CHECKI   EQU   *
```

```
         CLI   OPER,C'I'
         BE    INSERT
CHECKE   EQU   *
         CLI   OPER,C'E'
         BE    EXCLUDE
PUT      EQU   *
         MVC   FSCBCOMM(8),=CL8'DMSXFLWR'
         CLI   PROCORIG,C'Y'
         BNE   CHKTARG
         ST    3,FSCBAITN
         CLI   OPER,C'I'
         BE    CNGIBUF
         ST    8,FSCBBUFF
         B     WRITEI
CNGIBUF  EQU   *
         ST    1Ø,FSCBBUFF
WRITEI   EQU   *
         LA    Ø,EXTPLIST
         LA    1,FSCB
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
CHKTARG  EQU   *
         CLI   PROCTARG,C'Y'
         BNE   SKIWTARG
         ST    6,FSCBAITN
         CLI   FILLIN,C'R'
         BNE   CNGOBUF
         ST    9,FSCBBUFF
         B     WRITEO
CNGOBUF  EQU   *
         ST    1Ø,FSCBBUFF
WRITEO   EQU   *
         LA    Ø,EXTPLIST
         LA    1,FSCB
         ICM   1,8,=X'Ø2'
         SVC   2Ø2
         DC    AL4(1)
SKIWTARG EQU   *
         LA    3,1(3)
         LA    6,1(6)
         BCT   2,GET
         LR    1,8
         DMSFRET DWORDS=12ØØØ,LOC=(1)
         L     14,RG14
         BR    14
OPER     DS    C
FILLIN   DS    C
PROCORIG DS    C
PROCTARG DS    C
```

```
RG14     DS    F
DOUBLE   DS    D
EXTPLIST EQU   *
         DC    A(COMMVERB)
         DC    A(Ø)
         DC    A(Ø)
         DC    A(Ø)
COMMVERB DC    CL8'SUBCOM'
         DS    ØF
FSCB     EQU   *
FSCBCOMM DC    CL8' '
FSCBFN   DC    CL8' '
FSCBFT   DC    CL8' '
FSCBFM   DC    CL2' '
FSCBITNO DC    H'Ø'
FSCBBUFF DC    A(Ø)
FSCBSIZE DC    F'Ø'
FSCBFV   DC    CL1'F'
FSCBFLG  DC    X'2Ø'
FSCBNOIT DC    H'1'
FSCBNORD DC    AL4(Ø)
FSCBAITN DC    AL4(1)
FSCBANIT DC    AL4(1)
FSCBWPTR DC    A(Ø)
FSCBRPTR DC    A(Ø)
         END   XEHMA
```

XEHLP XEHLP

```
19 records starting with /PFK settings/
PFK settings
```

	Cursor in command line	Cursor in file area
PA2	No action	Undo
PF1	Help	New line
PF2	Save	Cursor in start of line
PF3	Quit	Cursor in end of line
PF4	Switch Top/current line	Copy left from cursor
PF5	Switch Bot/current line	Copy right from cursor
PF6	Left 4Ø	Schange 9
PF7	Up 21	Cursor to previous word
PF8	Down 21	Cursor to next word
PF9	Right 4Ø	Tabf
PF1Ø	Up 5	Cursor 1Ø columns to left
PF11	Down 5	Cursor 1Ø columns to right
PF12	Box processing	Centre cursor

XEINSTL EXEC should be used to install XEDIT/E. XEINSTL EXEC performs the following actions:

- Assemble XEHMA
- LOAD XEHMA and GENMOD.

All EXECs should be on any accessible disk. PROFILE XEDIT may be renamed – in this case its name should be explicitly passed as a parameter when XEDIT calls it.

Dobrin Goranov
Systems Programmer
Information Services (Bulgaria)

Tag files

Our installation has about 200 REXX procedures, and many of them use centrally-stored parameters.

Before I wrote the following functions for ‘tag files’, I had many of these parameter files, each with a different structure, and whoever changed a parameter value had to be very cautious not to violate the format of the file. And for each of these parameter files, there was at least one access procedure. As the number of procedures and parameter files grew, I decided to make an effort once and for all to simplify my life in the future.

‘Tag files’ offer the following benefits:

- A simple, consistent format
- Very readable parameter files with comments
- A single access procedure.

So what is a tag? In this context, it is simply a key/value pair or

something like a global, external associative array. The key is a single word, up to 38 characters long, and the value is a string of words with no length limitation.

The exact syntax for a tag file is very simple and is described in TAGFR EXEC, which is the access procedure.

Some of our applications make heavy use of tag files, and the files contain many comments, so the access procedure was sometimes not very fast. To speed things up, I wrote TAGFC EXEC, which reads a tag file and ‘compiles’ it into a compact format which can be read by TAGFR many times faster. TAGFR expects tag files with a filetype of ‘tagfile’ to be in compiled format; files with a different filetype are read the old slow way, which is OK for small files and avoids the overhead of having to compile the tag file after every change.

If many procedures access the same tag file, it is a good idea to encapsulate the call to TAGFR in a separate, application-specific access function, which could look like the one shown below.

```
/* sample application-specific access function */
arg tagkey
return tagfr(‘example tagfile u’  tagkey)
```

This way, the name of the tag file is known only to a single function.

TAGFR EXEC

The tag file access function.

```
/* ================================================================== */
/* Name        :  TAGFR    EXEC                                       */
/* ================================================================== */
/* Application :  tag files                                           */
/*                                                                    */
/* Usage       :  Function                                            */
/*                                                                    */
/* Arguments   :  fn ft fm tagname                                    */
/*                                                                    */
/* Result      :  tagvalue                                            */
/*                (a string of blank-delimited words)                 */
/*                                                                    */
/* Function    :  read a tag value from a tag file                    */
/*                                                                    */
/* syntax rules for tag files:                                        */
```

```
/*                                                                    */
/* - Each tag entry consists of a tag key and the corresponding value.*/
/* - The tag key must be the first word on a line and must end        */
/*   with a colon. It must not be longer than 38 characters and       */
/*   cannot contain any spaces. It is not case sensitive.             */
/* - The tag value consists of any words following the tag key,       */
/*   up to the next tag or to the end-of-file. It can span            */
/*   any number of lines. The tag value is returned as a string       */
/*   with single-space-delimited words.                               */
/* - Anything after /* is ignored as comment.                       */*/
/* - Blank lines are ignored.                                         */
/*                                                                    */
/* If the tagfile has a filetype of 'TAGFILE', tagfr assumes the      */
/* file was compiled with the tag file compiler tagfc and reads       */
/* it accordingly.                                                    */
/* ================================================================== */

arg fn ft fm tagname

if ft = 'TAGFILE'
then
   do
     'pipe <' fn ft fm '| locate 1-4Ø /'tagname'/ | stem f.'
     if f.Ø > Ø
     then
        return space(strip(substr(f.1,41)),1)
     else
        return ''
   end
'pipe < ' fn ft fm '| stem inrecs.'
takeit=Ø
tagvalue=''
do i=1 to inrecs.Ø
   parse value inrecs.i with text '/*' .     /* ignore comments */
   if strip(text) = '' then iterate
   /* detect start of tag value */
   if translate(word(text,1)) = tagname':'
   then
      do
        parse value text with . ':' text
        takeit=1
      end
   /* detect end of tag value = next tag */
   if takeit & right(word(text,1),1) = ':' then leave
   if takeit then tagvalue=tagvalue text
end
return space(tagvalue,1)
```

TAGFC EXEC

The tag file compiler.

```
/* ================================================================== */
/* Name        :  TAGFC    EXEC                                       */
/* ================================================================== */
/* Application :  TagFiles                                            */
/*                                                                    */
/* Usage       :  Procedure                                           */
/*                                                                    */
/* Arguments   :  ifn ift ifm ofn oft ofm                             */
/*                                                                    */
/* Result      :  -                                                   */
/*                                                                    */
/* Function    :  TagFile Compiler                                    */
/*                                                                    */
/* The tag file identified by ifn ift ifm is compiled and written     */
/* to the outputfile identifed by ofn oft ofm.                        */
/*                                                                    */
/* The output file has one record per tag entry and can be read       */
/* by TAGFR much more efficiently than a raw tag file.                */
/* ================================================================== */
arg ifn ift ifm ofn oft ofm

ifid=ifn ift ifm
ofid=ofn oft ofm
say 'TAGFC: tag file compiler'
out.Ø=Ø
'pipe < ' ifid '| stem inrecs.'
tagkey=''
do i=1 to inrecs.Ø
   parse value inrecs.i with text '/*' .     /* ignore comments */
   if strip(text) = '' then iterate
   /* detect start of tag value */
   if right(word(text,1),1)=':'
   then
      do
        call write_out
        tagkey=translate(strip(word(text,1),'T',':'))
        tagvalue=strip(substr(text,pos(':',text)+1))
      end
   else
      tagvalue=tagvalue strip(text)
end
call write_out
'pipe stem out. | >' ofid
say 'TAGFC: input file' ifid 'compiled to' ofid
say 'TAGFC: number of tags:' out.Ø
```

```
say 'TAGFC: processing completed'
return
write_out:
if tagkey ¬= ''
then
'pipe literal' left(tagkey,40) || strip(tagvalue) '| stem out. append'
return
```

SAMPLE TAG FILE

```
/* ================================================================== */
/* sample tag file                                                    */
/*                                                                    */
/* ================================================================== */
/* comment line                                                       */
this_is_a_tag_key: this is the value of the tag  /* and a comment */
another.tag: part one of the tag value
             part two of the tag value
```

SAMPLE EXEC

```
/* ================================================================== */
/* sample EXEC to demonstrate the tag file reader                     */
/* ================================================================== */
tagfile='tagsamp text a'
'vmfclear'
say 'demonstration of tag reader'
say tagfr(tagfile 'this_is_a_tag_key')
say tagfr(tagfile 'another.tag')
```

DEMONSTRATION

```
demonstration of tag reader
this is the value of the tag
part one of the tag value part two of the tag value
```

Electronic bulletin board

The rapid growth rate of the Internet and the World Wide Web demonstrate the interest and enthusiasm people have for communicating with each other using computers. An electronic bulletin board accessible by all employees to share information is perhaps one of the greatest software tools any programmer can deliver to a company. VM and PROFS provide a very suitable environment for such an application.

The bulletin board system presented here was developed in 1988 and is still in use today. Its success is due in part to the people who use it on a daily basis to keep each other informed. It is also very easy for everyone to use and requires little effort to specify users who have add/change/delete authority. It requires no routine 'cleanup' maintenance. People can add bulletins 'on the fly' as typed text, or they can attach memos which they have created using PROFS. They can post any CMS file as a bulletin. Many of our users load their department's status reports on the bulletin board instead of printing many copies and stuffing the boxes in the mail room.

This system consists of around two dozen CMS REXX EXECs and XEDIT macros, and requires that a disconnected virtual machine be defined in the VM directory for it to service the user community. If you do not already have a good electronic bulletin board at your company, you may be missing out on one of the best ways of giving your people the power to communicate effectively.

The following is a sample screen session of the 'Electronic Bulletin Board'.

From a command prompt type EBBNEWS to invoke EBBNEWS EXEC. This EXEC could also be made available from one of PROFS' menu pages. The EXEC displays the bulletin board menu screen – see Figure 1.

To view all the bulletins on all the boards press the PF7 key – see Figure 2.

```
 Monday          Electronic News - PROFS Bulletin Board          Ø5/Ø2/88

Press one of the following PF keys.

PF1  From The Desk of... the Company President
PF2  Safety News
PF3  Vistors and Plant Tours
PF4  Vacations and OUT-OF-TOWN Notices
PF5  FYI - General Information
PF6  Suggestions and Ideas to share



PF7  View All Bulletins    PF8  View New Bulletins    PF9  Help    PF12  End
-------------------------------------------------------------------------------
```

Figure 1: Bulletin board menu screen

To page forward through the bulletins press PF10 – see Figure 3.

To process a particular bulletin board, select the board from the first screen or move the cursor to a line in the board and press PF1 – see Figure 4.

Items with more information available such as a reference document or additional text will show with a ‘more...’ next to the subject.

In the example above, to view the additional information about the Chicago PROFS users who were moved to the Corporate computer, move the cursor down to the line and press PF4 or ‘enter’ – see Figure 5.

In the example above the text was typed in ‘on the fly’ and a script file was created containing only the text.

PF12 will return you to the ‘FYI’ bulletin board. To see how to add a bulletin from there, you can press PF1 to call up the ADD screen – see Figure 6.

The subject is typed along with any reference document information.

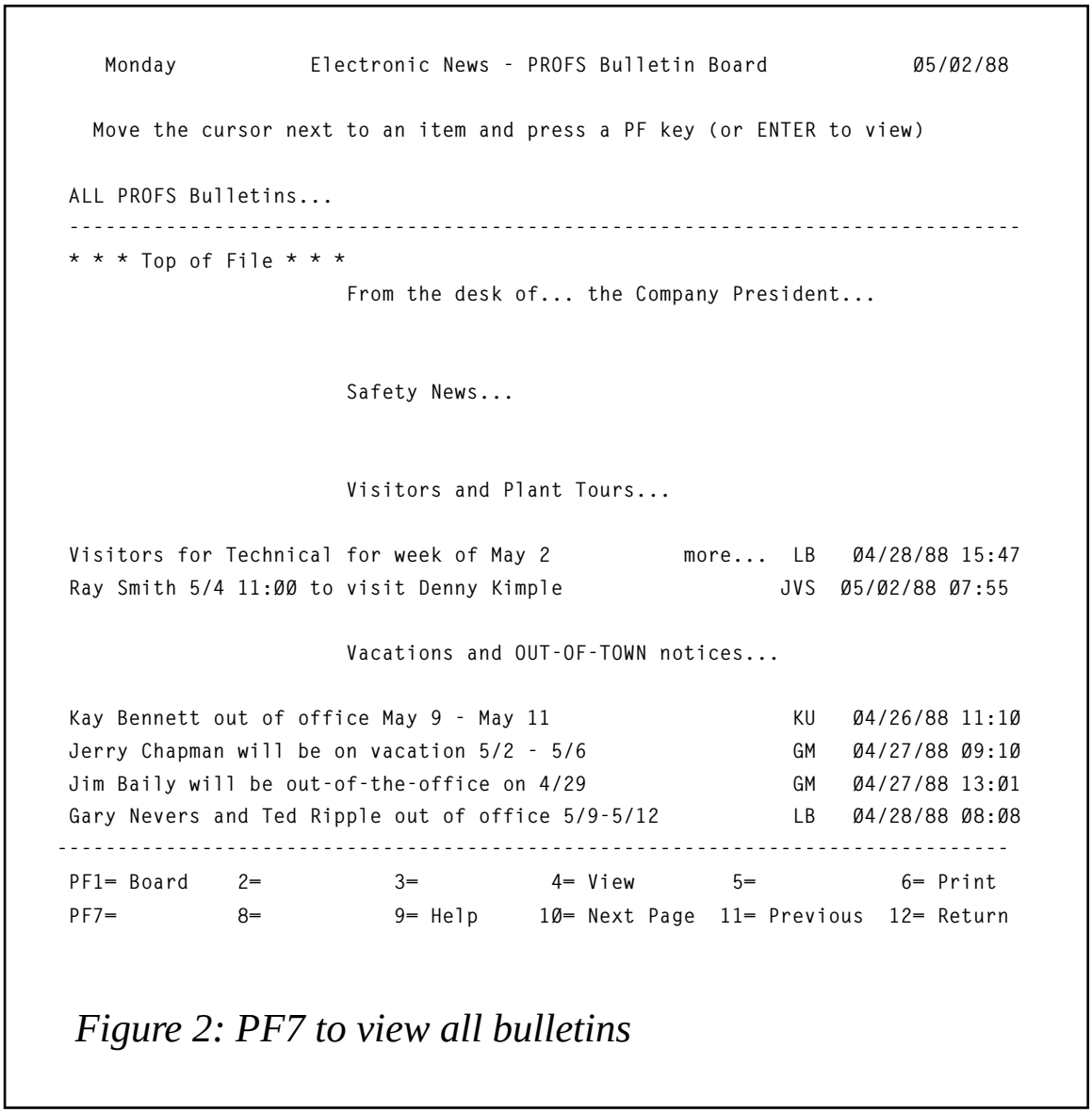

Figure 2: PF7 to view all bulletins

You can view the document before adding it to the bulletin board (in this case the 'FYI' board) by pressing the PF4 key. Once you view the document and return here, you can press PF1 to add the item to the bulletin board.

EBBNEWS has a summary of its functions available for on-line viewing by pressing the PF9 key from any EBBNEWS display. These functions are:

- The EBBonite News Bulletin Board (EBBNEWS) is a set of on-line programs that work with PROFS to simulate a real bulletin

```
   Monday          Electronic News - PROFS Bulletin Board          Ø5/Ø2/88

  Move the cursor next to an item and press a PF key (or ENTER to view)

ALL PROFS Bulletins...
--------------------------------------------------------------------------------
 Gary Nevers and Ted Ripple out of office 5/9-5/12          LB   Ø4/28/88 Ø8:Ø8
 Denny Kimple out of town 5/2 & 5/3                         JVS  Ø4/29/88 Ø8:31
 Bob Angelica on vacation 5/2-5/6                           JVS  Ø4/29/88 Ø8:32
 Frank Spittler out of office 4/29 & week of 5/2            LB   Ø4/29/88 1Ø:11
 Bill Jackson on vacation 5/4-5/6                           JVS  Ø5/Ø2/88 Ø7:52
 Roy Woodson will be out of the office 5/6/88.              IC   Ø5/Ø2/88 14:Ø9

                       FYI - General Information...

 New Hire - Danalynn Brandon                        more...  MW   Ø4/25/88 1Ø:Ø8
 J. Johnson on vacation 6/24-7/1                             JVS  Ø4/25/88 11:47
 BABY SHOWER - LUCY SMITH                           more...  JO   Ø4/26/88 1Ø:3Ø
 Salaried Picnic postponed until Friday, May 13th            GM   Ø4/27/88 Ø8:36
 Chicago Profs users are moved.                     more...  SWT  Ø4/27/88 Ø8:53
 Retirees                                           more...  TW   Ø4/27/88 11:45
 Gloria Masters on vacation 6/24 - 7/1                       GM   Ø4/27/88 13:42

                       Suggestions and Ideas to share...
--------------------------------------------------------------------------------
 PF1= Board    2=          3=           4= View       5=            6= Print
 PF7=          8=          9= Help     1Ø= Next Page  11= Previous  12= Return
```

Figure 3: Paging through bulletins

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

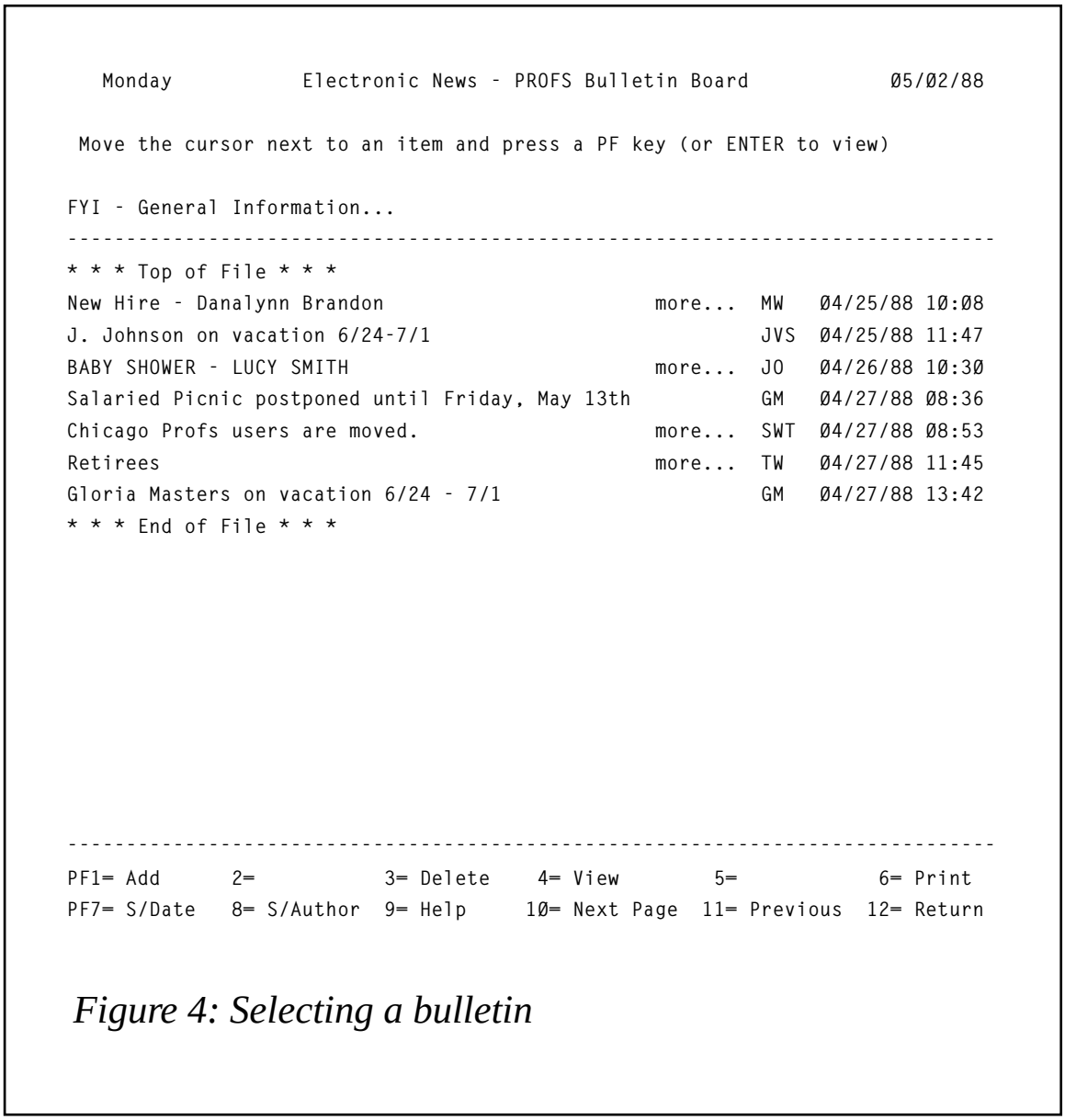

Figure 4: Selecting a bulletin

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

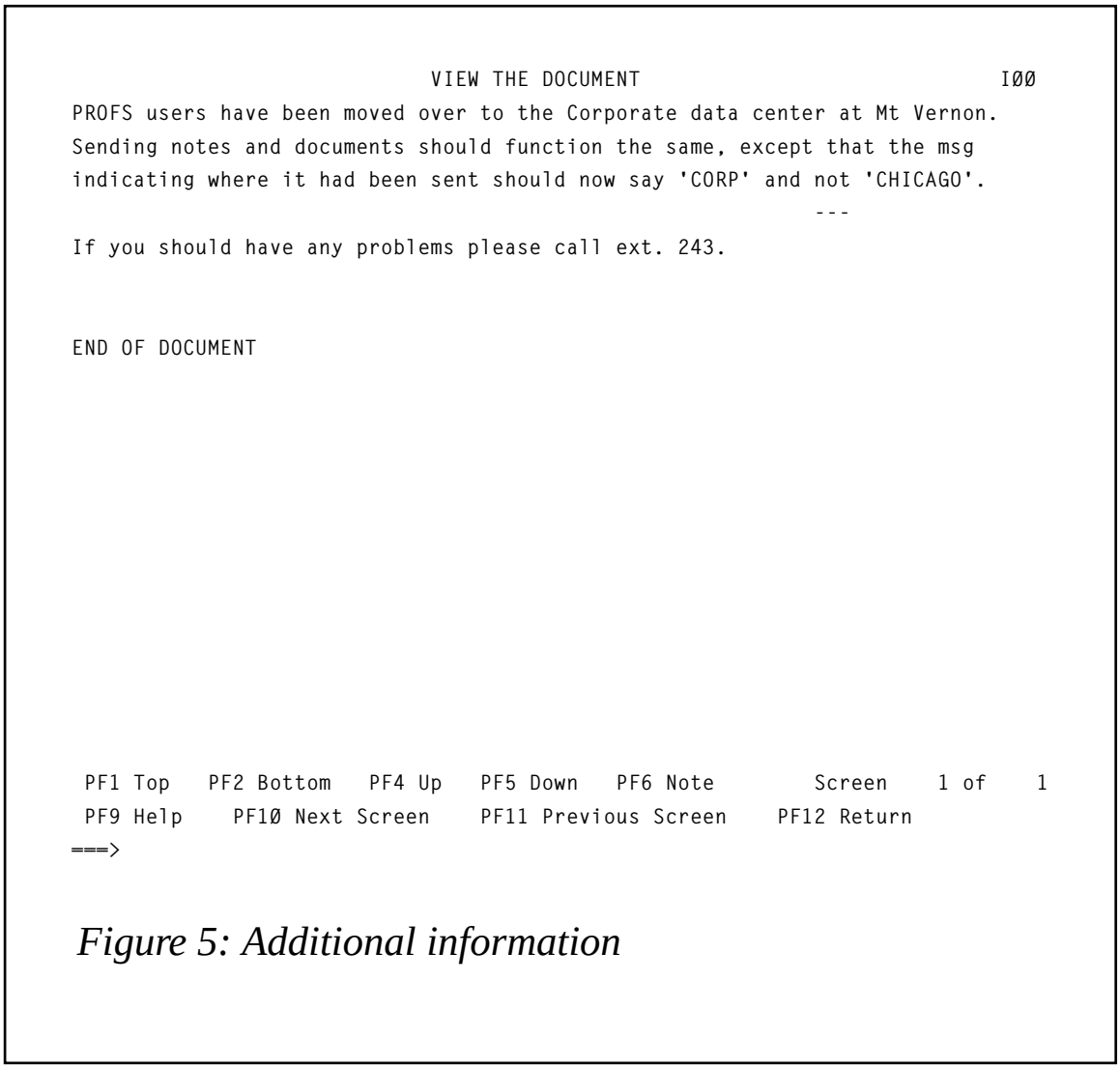

Figure 5: Additional information

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 users need add/delete ability then they can contact the Help Desk.

```
   Wednesday          Electronic News - PROFS Bulletin Board          Ø5/Ø4/88

 Supply the information below for "FYI - General Information..."

Subject:  ____________________________________________________

To reference a PROFS draft document supply a CRON number:  ____________

To reference a PROFS personal document supply a filename:  ________

To reference a non-PROFS document filename and filetype:   ________   ________

Or if you do not have a reference document and wish to elaborate on the
   subject then type additional comments below: (Y/N)  N

--------------------------------------------------------------------------------
* * * Top of File * * *









* * * End of File * * *
--------------------------------------------------------------------------------
PF1= Update Bulletin Board   PF2= Reset     PF4= View           PF12= Return
```

Figure 6: The add screen

- 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.

INSTALLATION INSTRUCTIONS

Installation instructions for EBBNEWS are shown below.

Define a virtual machine in your directory.

```
USER EBBNEWS xxxx 3M 4M BG
IPL CMS
CONSOLE ØØ9 3215
SPOOL ØØC 254Ø READER *
SPOOL ØØD 254Ø PUNCH B
SPOOL ØØE 14Ø3 A
LINK MAINT 319 319 RR
LINK MAINT 19Ø 19Ø RR
LINK MAINT 19E 19E RR
LINK SYSADMIN 399 399 RR
MDISK 191 3375 nnn ØØ2 xxxxxx MR RNEWS
MDISK 192 3375 nnn ØØ1 xxxxxx MR Rxxxx
```

The 192 mini-disk is for the EBBUSER AUTH file. This file has the list of people who can update the bulletin board with the read passwords to their A disks, in order to copy reference documents.

Create the EBBUSER AUTH file using the following format.

The USERIDxx names must be the user-ids from the VM directory:

```
|...+....1....+....2....+....3
USERIDØ1 ART  Rxxxxx
USERIDØ2 BA   Rxxxxx
USERIDØ3 GM   Rxxxxx
USERIDØ4 IC   Rxxxxx
USERIDØ5 JO   Rxxxxx
```

You must also set up an EBBUSER LIST file on EBBNEWS' A disk. EBBUSER LIST is the same as above but without the passwords. This file is used for the 'user to initials' verification; only let people add/replace/delete their own bulletins. The back-up user-ids can also update in case the user is not available.

```
|...+....1....+....2....+....3
USERIDØ1 ART BACKUPØ1 BACKUPØ2 BACKUPØ3
USERIDØ2 BA  BACKUPØ1 ........ ........
USERIDØ3 GM  BACKUPØ1 BACKUPØ2 ........
USERIDØ4 IC  ........ ........ ........
USERIDØ5 JO  BACKUPØ1 BACKUPØ2 BACKUPØ3
```

Set up a PROFILE EXEC and XEDIT profile on the EBBNEWS A disk.

PROFILE EXEC

```
|...+....1....+....2....+....3
/*                                                                 */
/* System     : EBBNEWS                                           */
/* EXEC name  : PROFILE                                           */
/* Invoked by : at LOGON or AUTOLOG time                          */
/* Function   : This EXEC accesses the required system mini-disks */
/*              and starts EBBNEWS by invoking the EBBSTART EXEC. */
/*                                                                 */
'acc 192 D'              /* -- EBBUSER AUTH on security disk     -- */
'acc 399 B'              /* -- EBBNEWS EXECS on PROFS disk       -- */
'acc 319 P'              /* -- FEATURES ( GDDM, ISPF, SOME APL ) -- */
'cp set PF3 IMMED PROFILE'
'cp set PF1Ø RETRIEVE'
queue 'EBBSTART'
exit
```

PROFILE XEDIT

```
|...+....1....+....2....+....3
/*                                                                 */
/* System     : EBBNEWS                                           */
/* Macro name : PROFILE                                           */
/* Invoked by : XEDIT                                             */
/* Function   : Standard XEDIT session                            */
/*                                                                 */
'SET CASE M'
'SET WRAP ON'
'SET STAY ON'
exit
```

Set up the bulletin boards on the EBBNEWS A disk. EBBNEWS uses the first line in each board for displaying the description on the screen. You can set up the boards for different purposes.

We use six bulletin boards, assigned to PF keys 1 to 6.

```
PRES EBBNEWS
    |...+....1....+....2....+....3
    From the desk of... the President

SAFETY EBBNEWS
    |...+....1....+....2....+....3
    REPORTS and Safety News

VISITOR EBBNEWS
    |...+....1....+....2....+....3
    Visitors and Plant Tours
```

```
FYI EBBNEWS
    |...+....1....+....2....+....3
    FYI - General Information

VACATION EBBNEWS
    |...+....1....+....2....+....3
    Vacations and OUT-OF-TOWN notices

IDEAS EBBNEWS
    |...+....1....+....2....+....3
    Suggestion Box... Ideas to share
```

Load the following EXECs and XEDIT macros to your SYSADMIN 399 disk. You may need to go through these programs and change the hardcoded references and descriptions of the boards to whatever you use.

```
EBBADD    EXEC       EBBBULL  EXEC
EBBCHECK  EXEC       EBBCURR  EXEC
EBBFLASH  EXEC       EBBHELP  EXEC
EBBNEWS   EXEC       EBBRECV  EXEC
EBBREPL   EXEC       EBBSEND  EXEC
EBBSTART  EXEC       INITGL   EXEC

EBBBULL   XEDIT      EBBCAL   XEDIT
EBBCURR   XEDIT      EBBDEL   XEDIT
EBBDELX   XEDIT      EBBEDIT  XEDIT
EBBGOTO   XEDIT      EBBMENU  XEDIT
EBBPRINT  XEDIT      EBBPRTX  XEDIT
EBBREPLX  XEDIT      EBBVIEW  XEDIT
```

Log-on to EBBNEWS and execute INITGL EXEC to initialize the global variable called EBBNEWS for the document counter. Log off.

Autolog the EBBNEWS virtual machine.

EBBADD EXEC

```
/*                                                                  */
/*  System     : EBBNEWS                                           */
/*  EXEC name  : EBBADD                                            */
/*  Invoked by : EBBBULL xedit macro                               */
/*  Function   : This EXEC adds detail information to the Bulletin */
/*               Board.  The user is prompted for either the cron  */
/*               number or a filename/filetype or data entered is  */
/*               saved as $TEMP$ $DATA$.  The ADD command is sent  */
/*               to EBBNEWS for adding to the list and his A disk. */
```

```
/*                                                                    */
  parse arg okay board desc
  if okay <> '$OKAY$'
     then do
          say 'This EXEC can only be invoked through the EBBNEWS EXEC'
          exit 99
          end

  'GLOBALV SELECT EBBNEWS GET naddr nmode'
  'SET CMSTYPE HT'
  'ACC' naddr nmode
  saverc = rc
  'SET CMSTYPE RT'
  if saverc then exit rc

  desbuf
  usr = USERID()
  'EXECIO * DISKR EBBUSER LIST' nmode,
                               '(FINIS ZONE 1 8 LOCATE /'usr'/ SKIP'
  if rc > Ø
     then do
     queue 'EMSG You are not authorized to add to the Bulletin Board.'
          exit
          end

  'CP SET SMSG ON'
  'smsg EBBNEWS ::: '
  if rc = 45
     then do
  queue 'EMSG EBBNEWS is not logged on. Check with computer operations'
          'CP SET SMSG OFF'
          exit
          end
     else
     if rc = 57
        then do
  queue 'EMSG EBBNEWS is not receiving. Check with computer operations'
             'CP SET SMSG OFF'
             exit
             end
     else
     if rc > Ø
        then do
  queue 'EMSG EBBNEWS is not available. NOTIFY OPERATIONS IMMEDIATELY'
             'CP SET SMSG OFF'
             exit
             end
  'CP SET SMSG OFF'

  'VMFCLEAR'
```

```
  'SET CMSTYPE HT'
  'STATE $EBBNEWS SCRIPT A'
  if rc = Ø then 'ERASE $EBBNEWS SCRIPT A'
  'STATE $EBBNEWS $SUBJ$ A'
  if rc = Ø then 'ERASE $EBBNEWS $SUBJ$ A'
  'STATE $EBBNEWS $JUNK$ A'
  if rc = Ø then 'ERASE $EBBNEWS $JUNK$ A'
  'SET CMSTYPE RT'

  queue 'ADD' board '........ ........ ... :::' desc
  'XEDIT $EBBJUNK SCRIPT A (PROFILE EBBEDIT'

  'ERASE $EBBNEWS $SUBJ$ A'
  'ERASE $EBBNEWS $JUNK$ A'
  'VMFCLEAR'
  queue 'QQUIT'
  exit
```

EBBBULL EXEC

```
/*                                                                    */
/*  System     : EBBNEWS                                             */
/*  EXEC name  : EBBBULL                                             */
/*  Invoked by : EBBNEWS XEDIT macro                                 */
/*  Function   : This EXEC displays the specific Bulletin Board      */
/*               for the category selected on the menu.              */
/*                                                                    */
  parse arg okay board .
  if okay <> '$OKAY$'
     then do
          say 'This EXEC can only be invoked through the EBBNEWS EXEC'
          exit 99
          end
  if board = '' then exit 99

  'GLOBALV SELECT EBBNEWS GET naddr nmode'
  'SET CMSTYPE HT'
  'ACC' naddr nmode
  saverc = rc
  'SET CMSTYPE RT'
  if saverc then exit rc

  'EXECIO * DISKR' board 'EBBNEWS' nmode '(FINIS STEM' item.
  last = item.Ø
  idate = SUBSTR(item.last,1Ø1,8)
  itime = SUBSTR(item.last,75,5)

  'DESBUF'
  'EXECIO * DISKR $EBBNEWS $CONTROL A (LOCATE /'board'/'
```

```
  saverc = rc
  'FINIS $EBBNEWS $CONTROL A'
  if saverc <> Ø
     then do
  queue 'EMSG ERROR reading the control file on your A disk. rc=' saverc
          exit
          end
  if QUEUED() <> 2
     then do
          queue 'EMSG ERROR locating' board 'on the control file'
          exit
          end
  parse pull recnum; parse pull record
  rest = SUBSTR(record,25,55)
  parse var recnum recnum .

  board = SUBSTR(board,1,8)
  newrec = board idate itime rest
  'EXECIO 1 DISKW $EBBNEWS $CONTROL A' recnum 'F 8Ø (STRING' newrec
  saverc = rc
  'FINIS $EBBNEWS $CONTROL A'
  if saverc <> Ø
     then do
  queue 'EMSG ERROR updating control file on your A disk. rc=' saverc
          exit
          end

  'VMFCLEAR'
  'XEDIT' board 'EBBNEWS' nmode '(PROFILE EBBBULL'

  'VMFCLEAR'
  exit
```

EBBBULL XEDIT

```
/*                                                                  */
/*  System     : EBBNEWS                                           */
/*  Macro name : EBBBULL                                           */
/*  Invoked by : EBBBULL EXEC                                      */
/*  Function   : This macro formats a PROFS Bulletin Board for     */
/*               a specific category of items.                     */
/*                                                                  */
'COMMAND SET AUTOSAVE OFF'
'COMMAND SET MSGMODE OFF'
'COMMAND SET SCOPE ALL'
'COMMAND SET CASE M I'
'COMMAND SET CMDLINE OFF'
'COMMAND SET CURLINE ON 5'
'COMMAND SET MSGLINE ON 2'
```

```
'COMMAND SET PREFIX OFF'
'COMMAND SET SCALE OFF'
'COMMAND SET WRAP ON'
'COMMAND SET STAY ON'
'COMMAND SET SHADOW OFF'
'COMMAND SET VERIFY 1 8Ø'
'COMMAND SET COLOR  *        BLUE   NONE    HIGH'
'COMMAND SET COLOR  CURLINE  GREEN  NONE  NOHIGH'
'COMMAND SET COLOR  FILEAREA GREEN  NONE  NOHIGH'
'COMMAND SET COLOR  MSGLINE  RED    NONE    HIGH'
'COMMAND :1'
'COMMAND EXTRACT /FNAME/CURLINE'
desc = CURLINE.3
'COMMAND :1'
'COMMAND DELETE 2'
'COMMAND SET LINEND OFF'
'COMMAND SET ENTER EBBVIEW'
'COMMAND SET PFØ1 EXEC EBBADD $OKAY$' FNAME.1 desc
'COMMAND SET PFØ2 EBBREPLX' FNAME.1 desc
'COMMAND SET PFØ3 EBBDEL' FNAME.1 'PASS1' desc
'COMMAND SET PFØ4 EBBVIEW'
'COMMAND SET PFØ5 EBBCAL'
'COMMAND SET PFØ6 EBBPRINT'
'COMMAND SET PFØ7 TOP#SORT * D 66 67 69 7Ø 72 73 75 79#CUR S 1 1'
'COMMAND SET PFØ8 TOP#SORT * A 61 63#CUR S 1 1'
'COMMAND SET PFØ9 EXEC EBBHELP EBBBULL'
'COMMAND SET PF1Ø FORWARD'
'COMMAND SET PF11 BACKWARD'
'COMMAND SET PF12 QQUIT'
'COMMAND SET LINEND ON #'
'COMMAND SET CTLCHAR ! ESCAPE'
'COMMAND SET CTLCHAR % PROTECT HIGH'
'COMMAND SET CTLCHAR @ PROTECT NOHIGH'

xxx   = DATE(W) '      '
yyy   = 'Electronic News - PROFS Bulletin Board'
zzz   = '       ' DATE(USA)
'COMMAND SET RESERVED  1 N  !@' xxx ' !%' yyy ' !@' zzz
'COMMAND SET RESERVED  2 BLUE NONE N  '

xxx  = '  Move the cursor next to an item and press a PF key',
       '(or ENTER to view) '
'COMMAND SET RESERVED  3 BLUE NONE N' xxx
'COMMAND SET RESERVED  4 BLUE NONE N  '

xxx = desc
'COMMAND SET RESERVED  5 WHITE NONE HIGH' xxx
xxx = COPIES('-',79)
'COMMAND SET RESERVED  6 BLUE NONE N' xxx
```

```
xxx = COPIES('-',79)
'COMMAND SET RESERVED -3 BLUE NONE N' xxx
xxx  = ' PF1= Add      2= Replace   3= Delete    4= View     ',
       '  5= Calendar   6= Print '
'COMMAND SET RESERVED -2 BLUE NONE HIGH' xxx
xxx  = ' PF7= S/Date   8= S/Author  9= Help     1Ø= Next Page',
       ' 11= Previous  12= Return'
'COMMAND SET RESERVED -1 BLUE NONE HIGH' xxx

'COMMAND SET MSGMODE ON'
'COMMAND CURSOR SCREEN 1 1'
'COMMAND TOP'
exit
```

EBBCAL XEDIT

```
/*                                                                    */
/*  System     : EBBNEWS                                             */
/*  Macro name : EBBCAL                                              */
/*  Invoked by : EBBBULL, EBBCURR XEDIT macros                       */
/*  Function   : This macro displays the current month calendar      */
/*               (with scrolling to other months) during XEDIT       */
/*               sessions (while view all bulletins or a specific    */
/*               board)... copied from previously written code       */
/*                                                                    */

INIT:
  'EXTRACT /RESERVED */CMDLINE/MSGLINE'

  'COMMAND SET CMDLINE OFF'
  'COMMAND SET MSGLINE OFF'
  'COMMAND SET CTLCHAR ! ESCAPE'
  'COMMAND SET CTLCHAR % PROTECT HIGH'
  'COMMAND SET CTLCHAR @ PROTECT NOHIGH'
  hi = '!%'
  lo = '!@'

  usr  = USERID()
  full = DATE(S)
  year = SUBSTR(full,1,4)
  dte  = DATE(USA)
  mm   = SUBSTR(dte,1,2)
  dd   = SUBSTR(dte,4,2)
  yy   = SUBSTR(dte,7,2)
  mth  = DATE(M)
  jul  = DATE(J)
  jda  = SUBSTR(jul,3,3)

  pcal = mm || Ø1 || yy
```

```
  call CALCJUL
  call CALCDAY
  call SETMO
  call DISPLAY

READ:
  'DESBUF'
  'COMMAND READ ALL TAG'
  do QUEUED()
     parse pull strg
     parse var strg action pfkey .
     if action = 'PFK'
        then do
             'DESBUF'
             signal PFKEY
             end
  end
  signal RESET

PFKEY:
  if pfkey =  1
     then do
          mm = mm - 1
          dd = Ø1
          if mm < 1
             then do
                  mm   = 12
                  yy   = yy - 1
                  year = year - 1
                  end
          end
  else
  if pfkey =  2
     then do
          mm = mm + 1
          dd = Ø1
          if mm > 12
             then do
                  mm   = Ø1
                  yy   = yy + 1
                  year = year + 1
                  end
          end
  if pfkey <> 1 & pfkey <> 2
     then signal RESET
  if LENGTH(mm) = 1 then mm = Ø || mm
  if LENGTH(yy) = 1 then yy = Ø || yy
  call CALCMTH
  pcal = mm || dd || yy
  call CALCJUL
```

```
  call CALCDAY
  call SETMO
  call DISPLAY
  signal READ

RESET:
  'COMMAND SET RESERVED +Ø2 OFF'
  'COMMAND SET RESERVED +Ø3 OFF'
  'COMMAND SET RESERVED +Ø4 OFF'
  'COMMAND SET RESERVED +Ø5 OFF'
  'COMMAND SET RESERVED +Ø6 OFF'
  'COMMAND SET RESERVED +Ø7 OFF'
  'COMMAND SET RESERVED +Ø8 OFF'
  'COMMAND SET RESERVED +Ø9 OFF'
  'COMMAND SET RESERVED +1Ø OFF'
  'COMMAND SET RESERVED +11 OFF'
  'COMMAND SET RESERVED +12 OFF'
  'COMMAND SET RESERVED +13 OFF'
  'COMMAND SET RESERVED +14 OFF'
  'COMMAND SET RESERVED +15 OFF'
  'COMMAND SET RESERVED +16 OFF'
  'COMMAND SET RESERVED +17 OFF'
  'COMMAND SET RESERVED +18 OFF'
  'COMMAND SET RESERVED +19 OFF'
  'COMMAND SET RESERVED +2Ø OFF'
  'COMMAND SET RESERVED +21 OFF'
  'COMMAND SET RESERVED +22 OFF'

  ix = 1
  do RESERVED.Ø
     'COMMAND SET RESERVED' RESERVED.ix
     ix = ix + 1
     end

  'COMMAND SET CMDLINE' CMDLINE.1
  'COMMAND SET MSGLINE' MSGLINE.1 MSGLINE.2 MSGLINE.3 MSGLINE.4

exit

DISPLAY:
  am = 'am'
  tim = SUBSTR(TIME(),1,5)
  hour = SUBSTR(tim,1,2)
  if LENGTH(hour) = 1
     then hour = ' ' || hour
     else if hour > 12
             then do
                  hour = hour - 12
                  am = 'pm'
                  if LENGTH(hour) = 1 then hour = ' ' || hour
```

```
                tim = hour || SUBSTR(tim,3,3)
                end
           else if hour = 12 then am = 'pm'

'COMMAND SET RESERVED +Ø2 BLUE NONE N '
'COMMAND SET RESERVED +Ø3 BLUE NONE N '
xxx = '<ENTER> = Reset,  PF1 = Previous month,  PF2 = Next month.'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +Ø4 BLUE NONE H' xxx
'COMMAND SET RESERVED +Ø5 BLUE NONE N '
xxx  = '+------------------------------------------+'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +Ø6 BLUE NONE N' xxx
xxx  = '|                                          |'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +Ø7 BLUE NONE N' xxx

xxx  = usr || '   Time : ' || tim am
xxx  = CENTER(xxx,46)
xxx  = '|' || xxx || '|'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +Ø8 BLUE NONE N' xxx
xxx  = '|                                          |'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +Ø9 BLUE NONE N' xxx

xxx  = mth || ' ' || year
xxx  = CENTER(xxx,46)
xxx  = '|' || xxx || '|'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +1Ø BLUE NONE N' xxx
xxx  = '|                                          |'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +11 BLUE NONE N' xxx

xxx  = 'Sun  Mon  Tue  Wed  Thu  Fri  Sat'
xxx  = CENTER(xxx,46)
xxx  = '|' || xxx || '|'
xxx  = CENTER(xxx,8Ø)
'COMMAND SET RESERVED +12 BLUE NONE N' xxx
```

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

P C Shumway
Systems Analyst (USA)

VM news

Macro 4 has announced the availability of the VM version of EnterWEB, a 3270 Web browser, which follows the November launch of an MVS version. The product enables users of 3270 terminals and PCs with 3270 emulation to browse textual information from the Internet and intranets.

The software's been designed so that all 3270 users can access the Internet from their existing terminals, without the need to invest in PC technology, while security and auditing facilities ensure that Internet access is limited to authorized users and monitored at all times. Users can also access any TCP/IP network application directly from a 3270 terminal using the Telnet support in EnterWEB.

Users access the product by dialling the EnterWEB CMS machine and connect to the Internet via TCP/IP.

For further information contact:
Macro 4, The Orangery, Turners Hill Road, Worth, Crawley, W Sussex, RH10 4SS, UK.
Tel: (01293) 886060.
Macro 4, 35 Waterview Blvd, PO Box 292, Parsippany, NJ 07054-0292, USA.
Tel: (201) 402 8000.

* * *

Lotus has announced the Lotus Calendar Connector for OfficeVision (LCCOV), a server-based connector that provides calendar interoperability in OfficeVision/VM, OfficeVision/MVS, OfficeVision/400, Time and Place/2, and the calendaring and scheduling functionality of the Lotus Domino 4.5 server. LCCOV provides coexistence between host-based and LAN-based calendaring and scheduling applications, helping to migrate OfficeVision's six million users to Domino 4.5-based calendaring and scheduling solutions.

IBM and Lotus have designated Domino 4.5 as the platform of choice for OfficeVision customers' future calendaring and scheduling needs. Hence the new calendar connector is designed to take full advantage of the Domino 4.5 server platform's attributes. Once in place, Notes users can share free-and-busy time information, schedule meetings and exchange meeting notices with OfficeVision/VM, OfficeVision/MVS, OfficeVision/400 and Time and Place/2 calendar users. OfficeVision and Domino calendar users receive real-time access to other users' availability regardless of their location in the enterprise.

The two companies have also announced the expanded beta of OfficeVision/VM-to-Notes and OfficeVision/400-to-Notes tools, for migrating host files, including address book distribution lists, mail logs and calendars, at their own pace.

The calendar connector costs $4,995.

For further information contact:
Lotus Development, 55 Cambridge Parkway, Cambridge, MA 02142-1295, USA.
Tel: (617) 577 8500.
Lotus Development (UK) Ltd, Lotus Park, The Causeway, Staines, Middlesex, TW18 3AG, UK.
Tel: (01784) 455445.

* * *

Xephon is holding a conference in London on 12-13 May called *VM Update '97*.

For further information contact Xephon at any of the addresses shown on page 2.