

# 143

# VM

*July 1998*

---

## **In this issue**

- 3 Identifying over-used directories
  - 7 An extended DATE function
  - 20 REXX tracking system re-visited –  
part 4
  - 29 Call for papers
  - 30 A professional copy tool
  - 39 VM-related newsgroups and  
mailing lists
  - 52 VM news
- 

© Xephon plc 1998

# update

# VM Update

---

## Published by

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

## North American office

Xephon/QNA  
1301 West Highway 407, Suite 201-405  
Lewisville, TX 75067  
USA  
Telephone: 940 455 7050

## 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).

## Subscriptions and back-issues

A year's subscription to *VM Update*, comprising twelve monthly issues, costs £175.00 in the UK; \$265.00 in the USA and Canada; £181.00 in Europe; £187.00 in Australasia and Japan; and £185.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1990 issue, are available separately to subscribers for £14.50 (\$22.50) each including postage.

## Editor

Robert Burgess

## Disclaimer

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

## 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.

## 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*.

---

© Xephon plc 1998. All rights reserved. None of the text in this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior permission of the copyright owner. Subscribers are free to copy any code reproduced in this publication for use in their own installations, but may not sell such code or incorporate it in any commercial product. No part of this publication may be used for any form of advertising, sales promotion, or publicity without the written permission of the publisher. Copying permits are available from Xephon in the form of pressure-sensitive labels, for application to individual copies. A pack of 240 labels costs \$36 (£24), giving a cost per copy of 15 cents (10 pence). To order, contact Xephon at any of the addresses above.

*Printed in England.*

# Identifying over-used directories

The SFSPACE EXEC was written to give our SFS administrators the ability to identify those directories that are, shall we say, over-used, ie where many users have write authority to an SFS tree. In our case, we used a single tree to allow for personnel changes over time and just added or removed a user's authority to see a tree.

This EXEC will scan an SFS root for a list of directories and then total up the number of blocks in use and give a file count. This information can be displayed at the terminal or placed into a file sorted by directory structure, block count, or file count (via a prompt to the user).

## USAGE

```
SFSPACE <filepool:>DIRID
```

If the filepool is not specified, then the default filepool will be used (SET FILEPOOL command). If there is no default filepool, the EXEC will give 0 blocks and 0 files. Any report will contain only headers.

Formatting errors will occur if a single directory has over 999,999 files or blocks in use. Summary information (prior to the menu) will still be correct.

## SFSPACE

```
*/
Default_FileId = 'SFSPACE RESULTS A'      /* Default output file name */
Top.0 = 2                                  /* Header for display/file */
Top.1 = 'Blocks Files   Directory Name '
Top.2 = '—— ————'
Parse arg Root .

If Pos('.',Root) <= Length(Root) then do
    Root = Root||'.'
End

'PIPE CMS LISTDIR' Root '( ALL',
'| Drop 1 ',
'| Spec 3-* 1 ',
'| Stem Data.'
```

```

Do Lp = 1 to Data.Ø
  'Pipe CMS ACC' Data.Lp 'Z',
    '| Stem Acc_Data.'
  Temp_Acc = 'Z'
  Select

    When Subword(Acc_Data.1,1,1) = 'DMSACRØ59E' then do
      Parse Var Acc_Data.1 . . . . . Temp_Acc .
    End
    When Substr(Subword(Acc_Data.1,1,1),1,1) = 'DMS' then do
      Say 'Access error' Data.Lp
      Exit 2
    End /* Access error */
    Otherwise NOP
  End /* Select */

  'Pipe (Endchar $)',
    'CMS Listfile * *' Temp_Acc '( ALLOC',
    '| Drop 1',
    '| a: Fanout',
    '| Spec W 7 1 ',
    '| REXX Summ ',
    '| Var Total',
    '$',
    'a:',
    '| Count Lines',
    '| Var Cur_File '
  Data.Lp = Right(Total,6,' ') Right(Cur_File,6,' ') Data.Lp
End

'PIPE (Endchar $) Stem Data.',
  '| a: Fanout',
  '| Spec w 1 1',
  '| REXX Summ ',
  '| Var Grand_Total',
  '$',
  'a:',
  '| Spec w 2 1 ',
  '| REXX Summ ',
  '| Var Total_Files'

'VMFCLEAR'
Say 'Total blocks used by root directory' Root 'is' Grand_Total

Say 'Total number of files (Starting at directory' root 'in tree) is'
Total_Files
Say
Say 'Enter your choice:'
Say ' 1) Review on-line (in directory sequence)'
Say ' 2) Review on-line sorted (descending) by blocks in use'
Say ' 3) Review on-line sorted (descending) by number of files in

```

```

directory'
Say
Say ' 4) Place list in file (in directory sequence)'
Say ' 5) Place list in file sorted (descending) by blocks in use'
Say ' 6) Place list in file sorted (descending) by number of files in
directory'
Say
Say '(File options allow you to specify file-id or use default of'
Default_Fileid'))'
Say
Say "Enter choice 1-6      (Default Enter is '1')"
```

```

Pull Ans .
If Ans = '' then Ans = 1
If Ans < 1 | Ans > 6 then do
    Say 'Invalid reply' Ans 'Defaulting to review on-line'
    Ans = 1
End

If Ans = 4 | Ans = 5 | Ans = 6 then do
    Hit = 0
    Say 'Please enter a file-id to write results (Enter defaults to'
Default_Fileid'))'
    Pull Ans2
    If Ans2 = '' then do
        Fileid = Default_Fileid
        Hit = 1
    End
    Else do
        Fileid = Ans2
    End
    If ¬Hit then do
        'Pipe cms Statew' Fileid '|' Stem State_Data. '
        Rcsv = Rc
    End
    Else do
        Rcsv = 28 /* Fake file not found because of default file-id */
    End
    Select
        When Rcsv = 28 Then do      /* File not found or invalid file-id -
use default */
            If Substr(State_Data.1,7,4) = '070E' then
                Fileid = Default_Fileid
            End /* Rc 28 */
        When Rcsv = 0 then do
            'Pipe Stem State_Data. | Console'
            Say 'File will be replaced with <Enter> Any other reply to use
default file-id'
            Pull Ans3
            If Ans3 = '' then NOP
            Else Fileid = Default_Fileid
        End /* Rc = 0 */
    End
End

```

```

When Rcsv = 24 | Rcsv = 20 then do
    'Pipe Stem State_Data. | Console'
    Say 'Invalid file-id or file mode - Using default file-id'
    Fileid = Default_Fileid
End /* Rc 24 or RC 20 */
Otherwise do /* All other RCs */
    'Pipe Stem State_Data. | Console'
    Fileid = Default_Fileid
End /* Otherwise */
End /* Select */
End /* If Ans = 4,5,6 */

```

```

Select
    When Ans = 1 then do
        Say Top.1
        Say Top.2
        'PIPE Stem Data. | Console '
    End /* 1 */
    When Ans = 2 then do
        Say Top.1
        Say Top.2
        'PIPE Stem Data. | Sort 1.6 D | Console '
    End /* 2 */
    When Ans = 3 then do
        Say Top.1
        Say Top.2
        'PIPE Stem Data. | Sort 8.6 D | Console '
    End /* 3 */
    When Ans = 4 then do
        'Pipe Stem Top. | >' Fileid
        'PIPE Stem Data. | >>' Fileid
    End /* 4 */
    When Ans = 5 then do
        'Pipe Stem Top. | >' Fileid
        'PIPE Stem Data. | Sort 1.6 D | >>' Fileid
    End /* 5 */
    When Ans = 6 then do
        'Pipe Stem Top. | >' Fileid
        'PIPE Stem Data. | Sort 8.6 D | >>' Fileid
    End /* 6 */
    Otherwise NOP
End /* Select */
'PIPE CMS REL Z'
Exit

```

*Editor's note: please address any comments on this EXEC to the author at Ronla01@cai.com.*

---

*Lawrence E. Rondot*  
*Computer Associates International (USA)*

© Xephon 1998

---

# An extended DATE function

## INTRODUCTION

Although I often read about a requirement for a function to manipulate dates in REXX, I had never felt the need for it, even though I had already done some Julian conversion routines. However, my need for such a function arose when I needed ‘yesterday’s date’ to build JCL automatically. At this point I decided to do some kind of general-purpose date manipulation function, and hence DATEFUNC was born. It can be called either as a command or as a function, depending on your requirements.

DATEFUNC handles dates from the beginning of our era to the end of the tenth millennium (although it’s probably too early to start worrying about year 10000 problems!). Where leap years are concerned, it follows the Gregorian calendar convention whereby a year has 366 days if it’s divisible by 4, but is not a leap year when divisible by 100, unless divisible by 400. This means 1600, 2000, and 2400 are leap years, but 1800 and 1900 are not.

Before looking at practical examples of its use, let’s take a look at the ‘formal’ specifications.

## DATEFUNC SPECIFICATIONS

DATEFUNC has three formats, each with a different finality. The format selection is automatic, depending on the parameters passed. All parameters are optional and must be separated by commas, since they are positional.

The first format (format a) is:

```
DATEFUNC date_in , fmt_date_in , shift , fmt_date_out
```

which returns date\_in, added by shift (number of days) where:

- ‘Date\_in’ – defaults to today.
- ‘Shift’ – defaults to zero.

- ‘Fmt\_date\_in’ and ‘fmt\_date\_out’ – are the input and output date formats. Their default is specified by the contents of variables ‘default\_fmt\_in’ and ‘default\_fmt\_out’. Currently, these are ‘YYYYMMDD’ for input date and ‘WWWWWWWWWW/YYYY/MM/DD’ for output.

Both input and output formats can be specified as a string formed by the combination of the symbols below in such a way that they match positions with the input or output date. The input format must specify a date completely (just ‘D’ or ‘M’ won’t be enough).

The output format can be any string containing any text, using the following symbols:

- ‘DD’ or ‘DDD’ for day (day of month or day of year – Julian style).
- ‘MM’ or ‘MMM’ or ‘MMMMMMMMMM’ (numeric month, three-letter month, or the full month name ).
- ‘YY’ or ‘YYYY’ for the year.
- ‘WWWWWWWWWW’ for the full weekday name.

For example:

```
DATEFUNC("900101,YYMMDD") ==> Monday 1990/01/01
DATEFUNC(".,.,Today is WWWWWWWWW") ==> Today is Saturday
```

Formats can also be specified using REXX’s DATE() options:

- ‘N’ for Normal – the same as ‘DD MMM YYYY’.
- ‘J’ for Julian – the same as ‘YYDDD’.
- ‘O’ for Ordered – the same as ‘YY/MM/DD’.
- ‘S’ for Standard – the same as ‘YYYYMMDD’.
- ‘U’ for USA – the same as ‘MM/DD/YY’.
- ‘E’ for Europe – the same as ‘DD/MM/YY’.
- ‘B’ for Base – the number of days since 01/01/0001.
- ‘C’ for Century – the number of days since the beginning of the



current century, including day one of the century.

- ‘D’ for days – the number of days in the current year, including today.
- ‘M’ for month – the full month name.
- ‘W’ for weekday – the full day name.

For example:

```
DATEFUNC("21056,C,,U") ==> 08/25/57
```

Note: the input date default format is ‘YYYYMMDD’, an eight-number string. However, since Julian date conversion is quite useful, and a Julian date is a five-digit string, a five-digit input date will be assumed to be Julian.

For example:

```
DATEFUNC(90001) ==> Monday 1990/01/01
```

is the same as:

```
DATEFUNC("90001,J") ==> Monday 1990/01/01
```

or:

```
DATEFUNC("90001,YYDDD") ==> Monday 1990/01/01
```

The second format (format *b*):

```
DATEFUNC date1-date2 , format_of_both_dates
```

returns the difference in days between two dates. This format is assumed when there is a minus sign in the first parameter. Both dates must be in the same format, optionally specified by the second parameter, with the same rules as in format *a*.

‘Date1’ can be omitted, defaulting to today. The returned number is unsigned, which means that ‘date2-date1’ is the same as ‘date1-date2’. For example:

```
DATEFUNC(-19650901) ==> 11815  
DATEFUNC("991231-980101,YYMMDD") ==> 729
```

The third format (format *c*):

```
DATEFUNC year
```

returns 1 if the year is a leap year, or 0 if it is not.

This format is selected if there is only one parameter and it is a four-digit figure (two digits are insufficient).

As I have said, DATEFUNC can work either as a command or as a function, 'saying' or returning the results. However, when there is an error, DATEFUNC issues a small message appropriate to the situation, when it is called as a command. As a function, any error will return as '-1.'

As a command, you can also access on-line help by means of DATEFUNC ? or DATEFUNC HELP.

## PRACTICAL EXAMPLES

Here are some examples of a few useful things you can do with DATEFUNC. Don't forget that you can use it as a function from within another EXEC:

```
DATEFUNC("parameters")
```

or as a command:

```
DATEFUNC parameters
```

All the examples below assume it is used as a function:

- Get yesterday's date in standard format:

```
DATEFUNC(",,-1,S") ==> 19980502
```

- Get the month number for three days ago:

```
DATEFUNC(",,-3,MM") ==> 04
```

- Get 3 January 2467 in DD-MM-YYYY format and find the day of the week:

```
DATEFUNC("24670301,,,DD-MM-YYYY is WWWWWWWW") ==> 03-01-2467 is  
Tuesday
```

- Get today's day of the week as a number (ie find out today's base date and divide by 7). This returns a number, where 0 is Monday, 1 is Tuesday, and so on:

```
DATEFUNC(",,,B")//7 ==> 5 /* 5 means Saturday */
```

- How old was Paul McCartney when *Sgt Pepper's Lonely Hearts Club Band* was released (in days)?:

```
DATEFUNC("19420618-19670601") ==> 9114
```

- And in years? (Do an integer divide):

```
DATEFUNC("19420618-19670601")%365 ==> 24
```

- If the Eiffel Tower took 795 days to build and was opened on 31 March 1889, when did its construction begin?:

```
DATEFUNC("31/Mar/1889,DD/MMM/YYYY,-795,N") ==> 26 Jan 1887
```

- How old is it in days? (Note that today is assumed by default):

```
DATEFUNC(-18890331) ==> 39845
```

- How much time left until the World Wide Crash?:

```
DATEFUNC(-20000101) ==> 581
```

- Find whether 2046 is a leap year (0 means no, 1 means yes):

```
DATEFUNC(2046) ==> 0
```

- And 1720?:

```
DATEFUNC(1720) ==> 1
```

Another way to do this is to ask for 29 February for a given year. If you get a result of -1 there is an error (ie an invalid date), which means the year is normal, otherwise, the year is a leap year:

```
DATEFUNC(20460229) ==> -1
```

Format the output with your own text. Specify month names and week days with the nine symbols necessary for the longest names (Wednesday and September). If the names are shorter, the function automatically adjusts the string. Note that the fourth parameter is never put into upper-case, unlike the others. Because of that, you should specify the symbols in upper-case, otherwise they will be treated as text. In this example, the input date defaults to Julian, because it has five digits:

```
DATEFUNC("55070,,,I was born on a WWWWWWWW of MMMMMMMM in Anywhere")
==> I was born on a Friday of March in Anywhere /* not true! */
```

## A FINAL NOTE ABOUT PARAMETER SPECIFICATION

In the examples shown above, almost all parameters appear in quotes. This ensures that they are passed correctly to the program – although the quotes are not always necessary, depending on the number of parameters specified, the call method, and the environment. As a matter of security, you may choose to use quotes to ensure that the entire string is passed.

## DATEFUNC REXX

```
/** REXX *****/
/*
/* DATEFUNC - Date manipulation utility. Works as command or
/* function. Parameters must have one of these formats:
/*
/* Format a: date_in, format_date_in, shift, format_date_out
/* returns date_in shifted by shift in format_date_out
/*
/* Format b: date1-date2, format_of_both_dates
/* returns number of days between date1 and date2
/*
/* Format c: year
/* returns 1 if year is leap, 0 otherwise
/*
/*****/

signal on error                /* avoid crash with */
signal on syntax              /* invalid input      */
parse arg aaa
aaa = translate(aaa,"","'")    /* get rid of quotes */
aaa = translate(aaa,"","'")    /* and parse args    */
parse var aaa datin,"fmt_in","shift","fmt_out
upper fmt_in                  /* avoid problems in */
datin = space(datin,0)         /* comparisons and   */
fmt_in = space(fmt_in,0)       /* length mismatch,  */
shift = space(shift,0)         /* except for fmt_out*/
fmt_out= strip(fmt_out,"T")    /* strip trailing bls*/

default_fmt_out = "WWWWWWWWW YYYY/MM/DD " /* default out format */
default_fmt_in  = "YYYYMMDD"              /* default in format  */

if datin="?"|datin="help"|datin="HELP" then signal helpe
parse source . calltype .      /* find out how called*/
base_cent = 693594             /* = date(b)-date(c) */
days_cent = 36524             /* days of a century  */
call month_table               /* memorize months     */
call week_table                /* memorize weekdays   */
```

```

if fmt_out="" then fmt_out=default_fmt_out      /* conv REXX defaults */
fmt_out = find_format(fmt_out)                 /* to standard format */

if shift = "" then shift = 0                   /* default shift      */
if datin = "" then do                          /* default date is    */
    fmt_in = "S"                                /* today in Standard  */
    datin = date(S)                             /* format             */
end

if pos("-",datin) > 0 then do                   /* datin has 2 dates  */
    p1 = pos("-",datin) -1                     /* separate them      */
    p2 = pos("-",datin) +1
    datin1 = substr(datin,1,p1)
    datin2 = substr(datin,p2)
    if pos("-",datin2) > 0 then signal syntax /* 2 minus sign,error */
    if datin1 = "" then datin1 = "TODAY"        /* implicit today     */
end
else do
    datin1 = ""                                /* datin has 1 date   */
    datin2 = datin
end
if length(datin2)=4 & datatype(datin2,"W"),    /* 4-digit = year,    */
& fmt_in = "" then do                          /* if no format_in    */
    fmt_out = leap_year(datin2)                 /* see if year is leap*/
    call saida                                 /* and exit           */
end
if length(datin2)=5 & datatype(datin2,"W"),    /* 5-digit = julian,  */
& fmt_in = "" then fmt_in = "J"                /* if no format_in    */

fmt_in = find_format(fmt_in)                   /* translate date(X)  */
days_in = process_input(datin2)               /* process datin2     */
if datin1="" then do                          /* if datin1 exists,  */
    if datin1 = "TODAY" then do                /* set implicit to    */
        fmt_in = "YYYYMMDD"                   /* today              */
        datin1 = date(S)
    end
    shift = process_input(datin1)               /* process datin1 and */
    fmt_out = abs(days_in-shift)                /* find difference in */
end                                              /* days & return it   */
else do
    call calculate_output                       /* only datin2, do    */
    call process_output                        /* whatever and return*/
end

saida:
if calltype = "COMMAND" then say fmt_out
else return fmt_out

exit

```

```

/*****
/*                               Subroutines                               */
*****/

/*****
/*      Process input format: select from three basic models:          */
/*      year-month-day type, base type, or century type. If the first */
/*      model is selected, then analyse it in detail                    */
/*      Returns the number of days since 1/1/0                          */
*****/
process_input:
arg dat_in
select
when pos("YY",fmt_in) > 0 then do                                     /* year-mth-day type */

    if pos("YY",fmt_in) > 0 then do                                     /* there is a year */
        if pos("YYYY",fmt_in) > 0 then do                             /* get input year in */
            yy = substr(dat_in,pos("YYYY",fmt_in),4) /* yyyy or yy format */
        end
        else do
            yy = substr(dat_in,pos("YY",fmt_in),2)
            yy = "19"yy
        end
        leap_in = leap_year(yy)                                       /* determine if year */
        call month_days_total leap_in                                /* is leap and set */
        call month_days_total leap_in                                /* month table */
        end                                                         /* accordingly */

    if pos("DD",fmt_in) > 0 then do                                     /* there is a day */
        if pos("DDD",fmt_in) > 0 then do
            ddd_in = substr(dat_in,pos("DDD",fmt_in),3)
            do z = 12 to 1 by -1
                if month_dtot.z < ddd_in then do
                    mm = z
                    dd = ddd_in - month_dtot.z
                    leave z
                end
            end z
        end
        else do
            dd = substr(dat_in,pos("DD",fmt_in),2)
            dd = strip(dd,"L","0")
        end
    end

    if pos("MM",fmt_in) > 0 then do                                     /*there is a month*/
        if pos("MMM",fmt_in) > 0 then do
            mmm_in = substr(dat_in,pos("MMM",fmt_in),3)
            do z = 1 to 12
                month = left(month_txt.z,3)
                upper month
            end
        end
    end
end

```

```

        if mmm_in = month then mm = z
    end
end
else do
    if pos("MM",fmt_in) > 0 then do
        mm = substr(dat_in,pos("MM",fmt_in),2)
        mm = strip(mm,"L","0")
    end
end
end

if dd>29 & mm=2 then call error -3          /* validate input */
if dd>28 & mm=2 & leap_in=0 then call error -2
if dd>30 & (mm=4|mm=6|mm=9|mm=11) then call error -3
if dd>31 then call error -3

ddd = month_dtot.mm + dd                  /* abs day in year */
days = yy*365 + yy%4 - yy%100 + yy%400 + ddd - leap_in
end                                       /* end when */

when pos("BB",fmt_in) > 0 then do        /* REXX date(B) fmt */
    days = dat_in + 366                  /* convert to my own*/
end

when pos("CC",fmt_in) > 0 then do        /* REXX date(C) fmt */
    days = dat_in + 366 + base_cent     /* convert to base */
end

otherwise nop
end                                       /* end select */
return days

/*****
/*    Days_in (days from 1/1/0) is shifted as requested and the
/*    result translated to year-month-day. Leap years are included.
*****/
calculate_output:
days_out = days_in + shift
if days_out < 0 then call error -4      /* date before 1/1/0 */
wk = days_out//7
wkday = week.wk
yy_out = days_out%365.2425
leap_out = leap_year(yy_out)
ddd_out=days_out-yy_out*365-yy_out%4+yy_out%100-yy_out%400+leap_out
if ddd_out = 366 & leap_out = 0 then do
    leap_out = 1
    ddd_out = 1
    yy_out = yy_out+1
end
call month_days_total leap_out

```

```

do z = 12 to 1 by -1
  if month_dtot.z < ddd_out then do
    mm_out = z
    dd_out = ddd_out - month_dtot.z
    leave z
  end
end
yyy = right(yy_out,4,"0")
yy  = right(yyy,2)
mm  = mm_out
dd  = dd_out
ddd = right(ddd_out,3,"0")
mm0 = right(mm,2,"0")
dd0 = right(dd,2,"0")
return

/*****
/*      Process output format overlaying symbols with their values      */
*****/
process_output:

if pos("DDD",fmt_out) > 0 then,
  fmt_out = overlay(ddd,fmt_out,pos("DDD",fmt_out))
if pos("DD",fmt_out) > 0 then,
  fmt_out = overlay(dd0,fmt_out,pos("DD",fmt_out))
if pos("MMMM",fmt_out) > 0 then do
  p1 = pos("MMMM",fmt_out)
  q1 = length(space(month_txt.mm,0))
  q2 = length(month_txt.mm) - q1
  p2 = p1 + q1
  fmt_out = overlay(month_txt.mm,fmt_out,p1)
  fmt_out = delstr(fmt_out,p2,q2)
end
if pos("MMM",fmt_out) > 0 then,
  fmt_out = overlay(month_txt.mm,fmt_out,pos("MMM",fmt_out),3)
if pos("MM",fmt_out) > 0 then,
  fmt_out = overlay(mm0,fmt_out,pos("MM",fmt_out))
if pos("YYYY",fmt_out) > 0 then,
  fmt_out = overlay(yyy,fmt_out,pos("YYYY",fmt_out))
if pos("YY",fmt_out) > 0 then,
  fmt_out = overlay(yy,fmt_out,pos("YY",fmt_out))

if pos("WW",fmt_out) > 0 then do
  p1 = pos("WW",fmt_out)
  q1 = length(space(wkday,0))
  q2 = length(wkday) - q1
  p2 = p1 + q1
  fmt_out = overlay(wkday,fmt_out,p1)
  fmt_out = delstr(fmt_out,p2,q2)
end

```



```

if pos("BB",fmt_out) > 0 then do          /* base format output */
    p1 = pos("BB",fmt_out)
    days_out = days_out - 366             /* conv to REXX date(B) */
    days_out = right(days_out,7)         /* 7 chars for 99991231 */
    fmt_out = overlay(days_out,fmt_out,p1)
end

if pos("CC",fmt_out) > 0 then do          /* century format output */
    p1 = pos("CC",fmt_out)
    days_out = days_out // days_cent     /* only one century */
    days_out = days_out - yyyy%400       /* -1 day each leap cent */
    if days_out=0 then days_out = days_cent /* 31.12.99 case */
    if yyyy//400=0 then days_out=days_out+1
    days_out = right(days_out,5)         /* 5 chars for 991231 */
    fmt_out = overlay(days_out,fmt_out,p1)
end

return

/***** leap_year returns 1 if year is leap, 0 otherwise *****/
/*          leap_year returns 1 if year is leap, 0 otherwise          */
/*****
leap_year:
    arg ano .
    bisexto = 0
    if ano//4 = 0 then do
        if ano//100 = 0 then do
            if ano//400 = 0 then bisexto = 1
        end
    else do
        bisexto = 1
    end
end
return bisexto

/*****
find_format:
    parse arg fmt1
    if length(fmt1)=1 then upper fmt1
    select
        when fmt1 = "" then fmt = default_fmt_in
        when fmt1 = "N" then fmt = "DD MMM YYYY" /* date(Normal) */
        when fmt1 = "J" then fmt = "YYDDD"      /* date(Julian) */
        when fmt1 = "O" then fmt = "YY/MM/DD"   /* date(Ordered) */
        when fmt1 = "S" then fmt = "YYYYMMDD"   /* date(Sorted) */
        when fmt1 = "U" then fmt = "MM/DD/YY"   /* date(USA) */
        when fmt1 = "E" then fmt = "DD/MM/YY"   /* date(European) */
        when fmt1 = "B" then fmt = "BB"         /* date(Base) */

```

```

        when fmt1 = "C" then fmt = "CC"           /* date(Century)          */
        when fmt1 = "W" then fmt = "WW"          /* date(Weekd) out only */
        when fmt1 = "D" then fmt = "DDD"         /* date(Day)   out only */
        when fmt1 = "M" then fmt = "MMMM"        /* date(Month) out only */
        otherwise fmt = fmt1
    end
return fmt

/*****

month_table:
    month_txt.1 = "January  "
    month_txt.2 = "February "
    month_txt.3 = "March    "
    month_txt.4 = "April    "
    month_txt.5 = "May      "
    month_txt.6 = "June     "
    month_txt.7 = "July     "
    month_txt.8 = "August   "
    month_txt.9 = "September"
    month_txt.10 = "October  "
    month_txt.11 = "November "
    month_txt.12 = "December "
return

month_days_total:
    arg leap
    month_dtot.1 = 0
    month_dtot.2 = 31
    month_dtot.3 = 59 + leap
    month_dtot.4 = 90 + leap
    month_dtot.5 = 120 + leap
    month_dtot.6 = 151 + leap
    month_dtot.7 = 181 + leap
    month_dtot.8 = 212 + leap
    month_dtot.9 = 243 + leap
    month_dtot.10 = 273 + leap
    month_dtot.11 = 304 + leap
    month_dtot.12 = 334 + leap
return

week_table:
    week.0 = "Saturday "
    week.1 = "Sunday   "
    week.2 = "Monday    "
    week.3 = "Tuesday   "
    week.4 = "Wednesday"
    week.5 = "Thursday  "
    week.6 = "Friday    "
return

```

```

error:
syntax:
  arg err
  if calltype = "COMMAND" then,
    select
      when err=-2 then say "ERROR - Input Feb.29 in normal year"
      when err=-3 then say "ERROR - Invalid day"
      when err=-4 then say "ERROR - Output date before Jan. 1, 0000"
      otherwise say "ERROR - Invalid parms - " err
    end
  end
exit -1

/*****/

help:
say "DATEFUNC has 3 possible formats with the following parameters, "
say "      separated by commas: "
say " Format a: date_in , format_date_in , shift , format_date_out "
say "      returns date_in shifted by shift in format_date_out "
say " Format b: date1-date2 , format_of_dates "
say "      returns number of days between date1 and date2 "
say " Format c: year "
say "      returns 1 if year is leap, 0 otherwise "
say
say "For format a, all parms are optional and default as follows: "
say "  date_in defaults to today "
say "  format_date_in defaults to "default_fmt_in
say "  shift (number of days to add to date_in) defaults to zero "
say "  format_date_out defaults to "default_fmt_out
say
say "For format b, date1 is optional, defaulting to today. "
say "  Date1 and date2 must have same format. Parm 2 is optional, "
say "  and defaults as above. "
say
say "For format c, year must have four digits. "
say
say "All date formats are specified with the following symbols: "
say "      DD or DDD          for day "
say "      MM or MMM or MMMMMMMM for month "
say "      YY or YYYY          for year "
say "      WWWWWWWWW          for weekday "
say "  or using the symbols of REXX's DATE() function: "
say "      B, C, D, E, J, M, N, O, S, U, W. "
exit

```

---

*Luis Paulo Figueiredo Sousa Ribeiro*  
*Systems Programmer*  
*Edinfor (Portugal)*

---

© Xephon 1998

## REXX tracking system re-visited – part 4

*This month we conclude the code for the Problem Tracking Facility (PTF), which has been re-written to be Year 2000 compatible.*

```
IF (&Z4 = ' ')
  .ATTR (Z4) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z5 = ' ')
  .ATTR (Z5) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z6 = ' ')
  .ATTR (Z6) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z7 = ' ')
  .ATTR (Z7) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z8 = ' ')
  .ATTR (Z8) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z9 = ' ')
  .ATTR (Z9) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z10 = ' ')
  .ATTR (Z10) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z11 = ' ')
  .ATTR (Z11) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z12 = ' ')
  .ATTR (Z12) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z13 = ' ')
  .ATTR (Z13) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z14 = ' ')
  .ATTR (Z14) = 'SKIP(ON) TYPE(OUTPUT)'

)PROC

&CPFKEY = .PFKEY
&CRESP = .RESP
VPUT (CPFKEY CRESP) PROFILE

)END

PTF00 CONFIGURATION PANEL

*****
*** Source for the PTF00 configuration panel... ***
*****

)ATTR
$ TYPE(TEXT) INTENS(LOW) COLOR(TURQ) CAPS(OFF) SKIP(ON)
% TYPE(OUTPUT) INTENS(LOW) COLOR(TURQ) CAPS(OFF) SKIP(ON)
- TYPE(TEXT) INTENS(LOW) COLOR(RED) CAPS(ON) SKIP(ON)
? TYPE(TEXT) INTENS(HIGH) COLOR(WHITE) CAPS(OFF) SKIP(ON)
```

```

_ TYPE(INPUT) COLOR(WHITE) CAPS(ON) HILITE(USCORE)
| TYPE(INPUT) COLOR(WHITE) CAPS(OFF) HILITE(REVERSE)
\ TYPE(INPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(ON) COLOR(YELLOW)
@ TYPE(OUTPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(OFF) SKIP(ON)
COLOR(YELLOW)
# TYPE(OUTPUT) INTENS(LOW) COLOR(RED) CAPS(OFF)
~ TYPE(TEXT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF) SKIP(ON)
)BODY LMSG(LONGMSG)
$PTF00 %P61 $
%WDATE
%acctyp $ |P01 $
&ZTIME
$
$ Enter$&P33 $,?OPEN$or specific search criteria...
+
$ ~~~~~~<==$Select$&P33 $ or?OPEN$ $
+
$ ~~~~~~<==$Select OPEN, CLOSED or ALL$&P33 $ $
+
$ ~~~~~~<==$Select OPEN/CLOSED/ALL on/after this date (yy/mm/dd) $
+
$ ~~~~~~<==$Select author who originally opened the$&P33 $ $
+
$ ~~~~~~<==$Select$&P34 $ $
+
$ ~~~~~~<==$Select$&P36 $ $
+
$ ~~~~~~<==$Select$&P37 $ $
+
$ ~~~~~~<==$Find words in description or text $
#LONGMSG
-&MESSAGE +
?
? PF2$=Restore Defaults ?PF3$=Return ?PF4$=Perm Upd ?
)INIT
.HELP = ROUTEH

IF (&MESSAGE > ' ')
.ALARM = YES

)PROC

&CPFKEY = .PFKEY
&CRESP = .RESP
VPUT (CPFKEY CRESP) PROFILE
VER (&P01,NB)

)END

```

[illegible]

```

.HELP  = ROUTEH

IF (&MESSAGE > ' ')
    .ALARM = YES

)PROC

&CPFKEY = .PFKEY
&CRESP  = .RESP
VPUT (CPFKEY CRESP) PROFILE
VER (&P11,NB)

)END

```

## PTF02 CONFIGURATION PANEL

```

*****
*** Source for the PTF02 configuration panel... ***
*****

```

```

)ATTR
$ TYPE(TEXT) INTENS(LOW)  COLOR(TURQ)  CAPS(ON) SKIP(ON)
% TYPE(OUTPUT) INTENS(LOW) COLOR(TURQ)  CAPS(OFF) SKIP(ON)
- TYPE(TEXT) INTENS(LOW)  COLOR(RED)   CAPS(OFF) SKIP(ON)
? TYPE(TEXT) INTENS(HIGH) COLOR(WHITE) CAPS(OFF) SKIP(ON)
# TYPE(OUTPUT) INTENS(LOW) COLOR(RED) CAPS(OFF)
@ TYPE(OUTPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(OFF) SKIP(ON)
COLOR(YELLOW)
  ^ TYPE(TEXT) INTENS(LOW)  COLOR(YELLOW) CAPS(OFF)
  * TYPE(TEXT) INTENS(LOW)  COLOR(YELLOW) CAPS(OFF) HILITE(USCORE)
  _ TYPE(INPUT) COLOR(WHITE) CAPS(ON) HILITE(USCORE)
  | TYPE(INPUT) COLOR(WHITE) CAPS(OFF) HILITE(REVERSE)
)BODY LMSG(LONGMSG)
$PTF02                                %P61                                $
%WDATE
%acctyp          $          |P21          $
&ZTIME
$                                                         ?
$&P34            $                                                         ?
$                                                         ?
|CID1  $|CID7    $|CID13  $|CID19  $|CID25  $|CID31  $|CID37  $|CID43  ?
|CID2  $|CID8    $|CID14  $|CID20  $|CID26  $|CID32  $|CID38  $|CID44  ?
|CID3  $|CID9    $|CID15  $|CID21  $|CID27  $|CID33  $|CID39  $|CID45  ?
|CID4  $|CID10   $|CID16  $|CID22  $|CID28  $|CID34  $|CID40  $|CID46  ?
|CID5  $|CID11   $|CID17  $|CID23  $|CID29  $|CID35  $|CID41  $|CID47  ?
|CID6  $|CID12   $|CID18  $|CID24  $|CID30  $|CID36  $|CID42  $|CID48  ?
$                                                         ?
$&P36            $                                                         ?

```

```

$
|VID1 $|VID7 $|VID13 $|VID19 $|VID25 $|VID31 $|VID37 $|VID43 ?
|VID2 $|VID8 $|VID14 $|VID20 $|VID26 $|VID32 $|VID38 $|VID44 ?
|VID3 $|VID9 $|VID15 $|VID21 $|VID27 $|VID33 $|VID39 $|VID45 ?
|VID4 $|VID10 $|VID16 $|VID22 $|VID28 $|VID34 $|VID40 $|VID46 ?
|VID5 $|VID11 $|VID17 $|VID23 $|VID29 $|VID35 $|VID41 $|VID47 ?
|VID6 $|VID12 $|VID18 $|VID24 $|VID30 $|VID36 $|VID42 $|VID48 ?
$
#LONGMSG
-&MESSAGE
? PF2$=Restore Defaults ?PF3$=Return ?PF4$=Perm Upd ?
)INIT
.HELP = ROUTEH

IF (&MESSAGE > ' ')
.ALARM = YES

)PROC

&CPFKEY = .PFKEY
&CRESP = .RESP
VPUT (CPFKEY CRESP) PROFILE
VER (&P21,NB)

)END

```

## PTF03 CONFIGURATION PANEL

```

*****
*** Source for the PTF03 configuration panel... ***
*****

)ATTR
$ TYPE(TEXT) INTENS(LOW) COLOR(TURQ) CAPS(ON) SKIP(ON)
% TYPE(OUTPUT) INTENS(LOW) COLOR(TURQ) CAPS(OFF) SKIP(ON)
- TYPE(TEXT) INTENS(LOW) COLOR(RED) CAPS(ON) SKIP(ON)
? TYPE(TEXT) INTENS(HIGH) COLOR(WHITE) CAPS(OFF) SKIP(ON)
_ TYPE(INPUT) COLOR(WHITE) CAPS(ON) HILITE(USCORE)
@ TYPE(OUTPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(OFF) SKIP(ON)
COLOR(YELLOW)
# TYPE(OUTPUT) INTENS(LOW) COLOR(RED) CAPS(OFF)
¬ TYPE(TEXT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF) SKIP(ON)
* TYPE(TEXT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF) HILITE(USCORE)
| TYPE(INPUT) COLOR(WHITE) CAPS(OFF) HILITE(REVERSE)
)BODY LMSG(LONGMSG)
$PTF03 %P61 $
%WDATE
%acctyp $ |P31 $

```



```

&ZTIME
$
$Enter or update control information as necessary... $
$
$
|P33 $ $~~~~~?
$
|P34 ==>$~~~~~?
$
|P35 ==>$~~~~~?
$
==>$~~~~~?
$
|P36 ==>$~~~~~? $Created $~~~~~?$~~~~~? $by
$~~~~~?
$
|P37 ==>$~~~~~? $Last Update $~~~~~?$~~~~~?$by
$~~~~~?
$
|P38 ==>$~? $Status ==>$~?
$
$
$
#/LONGMSG
-&MESSAGE
? PF2$=Restore Defaults ?PF3$=Return ?PF4$=Perm Upd ?
)INIT
.HELP = ROUTEH

IF (&MESSAGE > ' ')
.ALARM = YES

)PROC

&CPFKEY = .PFKEY
&CRESP = .RESP
VPUT (CPFKEY CRESP) PROFILE
VER (&P31,NB)
VER (&P33,NB)
VER (&P34,NB)
VER (&P35,NB)
VER (&P36,NB)
VER (&P37,NB)
VER (&P38,NB)

)END

```

## PTF07 CONFIGURATION PANEL

```
*****
*** Source for the PTF07 configuration panel... ***
*****
```

)ATTR

```
$ TYPE(TEXT) INTENS(LOW) COLOR(TURQ) CAPS(ON) SKIP(ON)
% TYPE(OUTPUT) INTENS(LOW) COLOR(TURQ) CAPS(OFF) SKIP(ON)
- TYPE(TEXT) INTENS(LOW) COLOR(RED) CAPS(OFF) SKIP(ON)
? TYPE(TEXT) INTENS(HIGH) COLOR(WHITE) CAPS(OFF) SKIP(ON)
# TYPE(OUTPUT) INTENS(HIGH) COLOR(WHITE) CAPS(OFF)
@ TYPE(OUTPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(OFF) SKIP(ON)
```

COLOR(YELLOW)

```
¬ TYPE(TEXT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF)
* TYPE(TEXT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF) HILITE(USCORE)
```

SKIP(ON)

```
_ TYPE(INPUT) COLOR(WHITE) CAPS(ON) HILITE(USCORE)
| TYPE(INPUT) COLOR(WHITE) CAPS(OFF) HILITE(REVERSE)
```

)BODY LMSG(LONGMSG)

```
$PTF17          %P61          $          %WDATE
%ACCTYP          $          %P7ID          $          &ZTIME
$
*Nbr$*&P7ID      $ *Description$          ?
$
#Z |XID1          $          |XDS1          $          ?
#Z |XID2          $          |XDS2          $          ?
#Z |XID3          $          |XDS3          $          ?
#Z |XID4          $          |XDS4          $          ?
#Z |XID5          $          |XDS5          $          ?
#Z |XID6          $          |XDS6          $          ?
#Z |XID7          $          |XDS7          $          ?
#Z |XID8          $          |XDS8          $          ?
#Z |XID9          $          |XDS9          $          ?
#Z |XID10         $          |XDS10         $          ?
#Z |XID11         $          |XDS11         $          ?
#Z |XID12         $          |XDS12         $          ?
#Z |XID13         $          |XDS13         $          ?
#Z |XID14         $          |XDS14         $          ?
$
#LONGMSG
-&MESSAGE
$ PF2$=Restore Defaults ?PF3$=Return ?PF4$=Perm Upd ?
$ PF7$=Backward          ?PF8$=Forward ?
```

)INIT

```
.HELP = ROUTEH
.ZVARS = '(Z1 Z2 Z3 Z4 Z5 Z6 Z7 Z8 Z9 Z10 Z11 Z12 Z13 Z14)'
```

```

IF (&MESSAGE > ' ')
    .ALARM = YES

IF (&Z1 = ' ')
    .ATTR (Z1) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z2 = ' ')
    .ATTR (Z2) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z3 = ' ')
    .ATTR (Z3) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z4 = ' ')
    .ATTR (Z4) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z5 = ' ')
    .ATTR (Z5) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z6 = ' ')
    .ATTR (Z6) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z7 = ' ')
    .ATTR (Z7) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z8 = ' ')
    .ATTR (Z8) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z9 = ' ')
    .ATTR (Z9) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z10 = ' ')
    .ATTR (Z10) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z11 = ' ')
    .ATTR (Z11) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z12 = ' ')
    .ATTR (Z12) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z13 = ' ')
    .ATTR (Z13) = 'SKIP(ON) TYPE(OUTPUT)'
IF (&Z14 = ' ')
    .ATTR (Z14) = 'SKIP(ON) TYPE(OUTPUT)'

)PROC
    &CPFKEY = .PFKEY
    &CRESP = .RESP
    VPUT (CPFKEY CRESP) PROFILE

)END

```

## PTF99 PANEL

```

*****
*** Source for the PTF99 panel... ***
*****

```

```

)ATTR
$ TYPE(TEXT) INTENS(LOW) COLOR(TURQ) CAPS(ON) SKIP(ON)
% TYPE(OUTPUT) INTENS(LOW) COLOR(TURQ) CAPS(OFF) SKIP(ON)
{ TYPE(OUTPUT) INTENS(LOW) COLOR(YELLOW) CAPS(OFF) SKIP(ON)

```

```

! TYPE(OUTPUT) INTENS(NON)  COLOR(TURQ)  CAPS(OFF) SKIP(ON)
- TYPE(TEXT) INTENS(LOW)  COLOR(RED)  CAPS(ON) SKIP(ON)
? TYPE(TEXT) INTENS(HIGH)  COLOR(WHITE) CAPS(OFF) SKIP(ON)
} TYPE(OUTPUT) INTENS(HIGH) COLOR(WHITE)  CAPS(OFF) SKIP(ON)
\ TYPE(TEXT) INTENS(HIGH)  COLOR(YELLOW) CAPS(OFF) SKIP(ON)
_ TYPE(INPUT) COLOR(WHITE) CAPS(ON) HILITE(USCORE)
@ TYPE(OUTPUT) INTENS(HIGH) HILITE(REVERSE) CAPS(OFF) SKIP(ON)
COLOR(YELLOW)
# TYPE(OUTPUT) INTENS(LOW) COLOR(RED) CAPS(OFF)
¬ TYPE(TEXT) INTENS(LOW)  COLOR(YELLOW)  CAPS(OFF)
* TYPE(TEXT) INTENS(LOW)  COLOR(YELLOW)  CAPS(OFF) HILITE(USCORE)
| TYPE(INPUT) COLOR(WHITE) CAPS(OFF) INTENS(NON)
)BODY LMSG(LONGMSG)
$PTF99          %P61          $          %WDATE
@ACCTYP          $          PASSWORD PANEL          &ZTIME
$
$
$
$
$
$
%pwdc1          {pc1|pwd1      %PWDMSG1          ?
%pwdc2          {pc2|pwd2      %PWDMSG2          ?
$              %PWDMSG3          ?
$              %PWDMSG4          ?
$ %pwdc3        {pc3|pwd3      %PWDMSG5          ?
$              %PWDMSG6          ?
$ %pwdc4        {pc4|pwd4      %PWDMSG7          ?
$
!pwdnull$
$
$
$
$
$
$
#LONGMSG
-&MESSAGE
$          ?PF3$=Return  ?PF4$=Exit  ?

)INIT
.HELP      = ROUTEH

IF (&MESSAGE > ' ')
.ALARM = YES

IF (&PWDNULL = 'C','D')
.ATTR (PWD3) = 'SKIP(ON) TYPE(OUTPUT)'
.ATTR (PWD4) = 'SKIP(ON) TYPE(OUTPUT)'
.ATTR (PWDMSG1) = 'INTENS(NON)'
.ATTR (PWDMSG2) = 'INTENS(NON)'
.ATTR (PWDMSG3) = 'INTENS(NON)'

```

```

.ATTR (PWDMSG4) = 'INTENS(NON)'
.ATTR (PWDMSG5) = 'INTENS(NON)'
.ATTR (PWDMSG6) = 'INTENS(NON)'
.ATTR (PWDMSG7) = 'INTENS(NON)'

IF (&PWNUL = 'C')
.ATTR (PWDC2) = 'INTENS(NON)'
.ATTR (PC2)   = 'INTENS(NON)'
.ATTR (PWD2)  = 'INTENS(NON) SKIP(ON) TYPE(OUTPUT)'

IF (&PWNUL = 'D')
.ATTR (PWDC1) = 'INTENS(NON)'
.ATTR (PC1)   = 'INTENS(NON)'
.ATTR (PWD1)  = 'INTENS(NON) SKIP(ON) TYPE(OUTPUT)'

)PROC

&CPFKEY = .PFKEY
&CRESP  = .RESP
VPUT (CPFKEY CRESP) PROFILE

)END

```

---

*Steve Bernard*  
*Senior Systems Programmer (USA)*

Xephon 1998

---

## Call for papers

Why not share your expertise and earn money at the same time? *VM Update* is looking for REXX EXECs, macros, program code, etc, that experienced VMers have written to make their life, or the lives of their users, easier. We will publish it (after vetting by our expert panel) and send you a cheque when the article is published. Articles can be of any length and can be sent or e-mailed to Robert Burgess at any of the addresses shown on page 2. Why not call now for a free copy of our *Notes for contributors*?

# A professional copy tool

## GENERAL DESCRIPTION

FCOPYPRO EXEC is designed to fit precise copying requirements. It supports enhanced copy functions, which allow you to:

- Copy a selected set of files, residing on several mini-disks, to different target locations, by a single FCOPYPRO call.
- Split a file into different files of a predefined size.
- Join a set of files to form one file.

These functions may be performed depending on size, record position, or length of the source files. The filename specification for processed files may be deferred until FCOPYPRO is started. This allows you to change source and target copy locations dynamically at execution time.

FCOPYPRO is written in REXX and uses the FCOPY MODULE, which was described in *Fast copying of fixed length files, VM Update*, Issue 141, May 1998. FCOPY reduces file copying time and is therefore used to support repeatedly generated copy requests during FCOPYPRO execution.

## FCOPYPRO EXEC USAGE

The FCOPYPRO EXEC is invoked as shown below:

```
FCOPYPRO <C[opy]|S[plit]|J[oin]> <source> <target> [[<FCOPY parameters>]]
```

where:

- C[opy] specifies the COPY function.
- S[plit] specifies the SPLIT function.
- J[oin] specifies the JOIN function.
- Source specifies the name of the source file, or selection criteria describing the source set of files.

- Target specifies the name of the target file or the generic name of the set of target files.

Some FCOPY parameters, namely FR[om], FO[r], R[ep], and A[pp] are the same as the corresponding CMS COPYFILE parameters. Parameters which differ are:

- F n – write the target file with RECFM F and LRECL n.
- V n – write the target file with RECFM V and LRECL n.
- P[os] n – copy part of the source file starting from position n within each record.

To define a set of files, the following special characters may be used as placeholders in the ‘source’ parameter:

- ‘\*’ – which replaces any number of characters.
- ‘%’ – which represents a single character.
- ‘?’ – which defers the specification of FN, FT, or FM until execution.

The following special characters may be used as placeholders in the ‘target’ parameter:

- ‘\*’ – which replaces the corresponding FN, FT, or FM of the source file.
- ‘=’ – which is equivalent to the placeholder ‘\*’.
- ‘?’ – which defers the specification of FN, FT, or FM until execution.

If the placeholder ‘?’ is used in ‘source’, no other placeholder can be specified. In this case, the remaining file name parts must be explicitly declared or be replaced by ‘?’ as well.

In ‘target’, the placeholders ‘\*’, ‘=’, and ‘?’ may be used together. The placeholders ‘\*’ and ‘=’ may be used to replace the corresponding FN, FT, or FM of the source file in reply to the FCOPY prompt to specify, identified by the placeholder ‘?’ part of the target filename.

To auto-rename new files when the target file is split, FCOPYPRO

firstly checks the FT. If this contains a number, then FT uses this as a basis for renaming files incrementally. If FT is not a number, FCOPYPRO checks FN. If FN is a number, this becomes the basis for renaming. Otherwise the auto-renaming assumes a value of FN equal to 1 and uses this as a basis for renaming. For example:

<target>	Generated filenames
FN 100 A	FN 100 A; FN 101 A; FN 102 A
100 FT A	100 FT A; 101 FT A; 102 FT A
FN FT A	1 FT A; 2 FT A; 3 FT A

Note: FN and FT are not numbers.

Examples of FCOPYPRO usage are:

- Copying the first 10 lines of a set of files with filenames AN, ABN, ABCN, and filetypes FT, to mini-disk B:

```
FCOPYPRO C A* FT A = = B (FR 1 FO 10
```

- Copying a set of files with filenames FN and filetypes NA, NB, and NC to different mini-disks:

```
FCOPYPRO C FN N% A * * ?
```

- Copying a set of files with filenames ANAB, ABNCD, and ABCNDEFG, and filetypes FT, which reside on different mini-disks, to mini-disk B:

```
FCOPYPRO C *%N%* FT ? = = B
```

- Copying a set of files from different locations to different mini-disks:

```
FCOPYPRO C ? ? ? * * ?
```

- Split a file to give files of 1,000 records, with auto-renaming FT, residing on disk B:

```
FCOPYPRO S FN FT FM = 100 B (FO 1000
```

- Split a file to give files of 100,000 records, without auto-renaming. In this case FCOPYPRO will give a prompt for you to enter a filename for each part of the file at the time of execution. Different file parts may be written on different mini-disks:

```
FCOPYPRO S FN FT FM ? ? ? (FO 100000
```



- Join record number 7 of a set of files, with filetype FT, to file FN REC007 B, searching on all accessible mini-disks:

```
FCOPYPRO J * FT * FN REC007 B (FR 7 FO 1
```

- Combine different files from different locations into file FN FT B. Some files may repeatedly write to file FN FT B:

```
FCOPYPRO J ? ? ? FN FT B
```

## FCOPYPRO EXEC

```

/*****
/***                                     ***      ***/
/*** FCOPYPRO               copy tool          ***      ***/
/***                                     ***      ***/
/*****
/***   SIZE 00285  VER 1.0 MOD 000                                     ***/
/*****

```

```
PARSE UPPER ARG OP SN ST SM TN TT TM FCOPY_PARMS_IN
```

```
SIGNAL ON ERROR
```

```
IF ¬ (OP = 'C' | OP = 'COPY' | OP = 'J' | OP = 'JOIN' |
      OP = 'S' | OP = 'SPLIT') THEN
```

```
DO
```

```
  SAY '- Unknown function code'
```

```
  SAY '    Select C[opy|, J[oin| or S[plit|'
```

```
  SIGNAL ERROR
```

```
END
```

```
FCOPY_PARMS = TRANSLATE(FCOPY_PARMS_IN, '40'X, '(')
```

```
IF SUBSTR(OP, 1, 1) = 'S' THEN
```

```
DO
```

```
  DO I = 1 BY 1 WHILE (I <= WORDS(FCOPY_PARMS))
```

```
    KEY = WORD(FCOPY_PARMS, I)
```

```
    IF KEY = 'FO' | KEY = 'FOR' | KEY = 'FR' | KEY = 'FROM' THEN
```

```
      DO
```

```
        IF SUBSTR(KEY, 1, 2) = 'FO' THEN
```

```
          SIZE = WORD(FCOPY_PARMS, I + 1)
```

```
        ELSE
```

```
          START = WORD(FCOPY_PARMS, I + 1)
```

```
          FCOPY_PARMS = DELWORD(FCOPY_PARMS, I, 2)
```

```
          I = I - 1
```

```
        ITERATE
```

```
      END
```

```
    IF KEY = 'A' | KEY = 'APP' | KEY = 'FROM' | KEY = 'FR' THEN
```

```
      DO
```

```
        SAY '- A[pp| or FR[om| are not valid for SPLIT'
```

```

        SIGNAL ERROR
    END
END
IF ¬ DATATYPE(SIZE,'N') THEN
SIZE = 0
IF ¬ DATATYPE(START,'N') THEN
START = 1
IF SIZE < 1 THEN
DO
    SAY '- FO[r] value is not valid for SPLIT'
    SIGNAL ERROR
END
SIGNAL OFF ERROR
MAKEBUF
LISTFILE SN ST SM '(' STACK AL
IF QUEUED() > 1 THEN
DO
    SAY '- Too many files to split'
    DROPBUF
    SIGNAL ERROR
END
IF QUEUED() = 0 THEN
DO
    SAY '- Source file not found for SPLIT'
    SIGNAL ERROR
END
SIGNAL ON ERROR
PULL . . . . . RECS .
DROPBUF
IF INDEX(TN TT TM, '?') ≠ 0 THEN
MOD_TYPE = 'N'
ELSE
DO
    IF DATATYPE(TT,'N') THEN
MOD_TYPE = 'Y'
    ELSE
    IF ¬ DATATYPE(TN,'N') THEN
TN = 1
    END
I = RECS % SIZE + MIN(1,RECS // SIZE) + 1
OPER = 'SPLIT'
END
ELSE
DO
    SIGNAL OFF ERROR
    SET CMSTYPE HT
    STATE FCOPYPRO XEDIT A
    RC_CODE = RC
    SET CMSTYPE RT
    IF RC_CODE = 0 THEN

```

```

DO
  SAY '- 'FCOPYPRO XEDIT 'found - rename or erase it'
  OP = 'X'
  SIGNAL ERROR
END
SIGNAL ON ERROR
GEN_BUF.1 = '/* FCOPYPRO XEDIT at' TIME() 'on' DATE() '* DG'98 */'
GEN_BUF.2 = '  DMSXMS 8 24'
GEN_BUF.3 = '  FF'
EXECIO 3 DISKW FCOPYPRO XEDIT A '(' FINI STE GEN_BUF.
DROP GEN_BUF.
IF INDEX(SN ST SM, '?') = 0 THEN
DO
  LISTFILE SN ST SM '(' E FM
  XEDIT CMS EXEC A '(' PROF FCOPYPRO
  SIGNAL OFF ERROR
  MAKEBUF
  SIGNAL ON ERROR
  EXECIO '*' DISKR CMS EXEC A '(' FINI FIFO
  DO I = 1 TO QUEUED()
    PULL . . SN.I ST.I SM.I .
    SN.I = LEFT(SN.I, 8)
    ST.I = LEFT(ST.I, 8)
    SM.I = LEFT(SM.I, 2)
    IF I > 1 THEN
      DO
        J = I - 1
        IF SN.I = SN.J THEN
          IF ST.I = ST.J THEN
            DO
              DROPBUF
SAY '- Duplicate names ' SN.I ST.I SM.I '<->' SN.I ST.I SM.J
              SIGNAL ERROR
            END
          END
        END
      J_C_MODE = ''
    END
  ELSE
    DO
      I = 999
      J_C_MODE = 'D'
    END
  IF SUBSTR(OP, 1, 1) = 'J' THEN
    DO
      OPER = 'JOIN '
      IF FCOPY_PARMS_IN = '' THEN
        FCOPY_PARMS_IN = '('
        FCOPY_PARMS = FCOPY_PARMS_IN A

```

```

END
ELSE
DO
    OPER = 'COPY '
    FCOPY_PARMS = FCOPY_PARMS_IN
END
J_SN = SN
J_ST = ST
J_SM = SM
END
C_TN = TN
C_TT = TT
C_TM = TM
SAY COPIES('>', 21) 'Start' OPER 'at' TIME() COPIES('<', 21)
DO J = 1 TO I-1
    IF SUBSTR(OPER, 1, 1) = 'S' THEN
        FCOPY_PARMS = FCOPY_PARMS_IN FROM (J-1) * SIZE + 1
    ELSE
        DO
            IF J_C_MODE = '' THEN
                DO
                    SN = SN.J
                    ST = ST.J
                    SM = SM.J
                END
            ELSE
                DO
                    IF J_SN = '?' | J_ST = '?' | J_SM = '?' THEN
                        DO
                            IF J_SN = '?' THEN
                                DO
                                    GET_N = 'SN'
                                    PUT_N = 'FN'
                                END
                            ELSE
                                DO
                                    GET_N = ''
                                    PUT_N = ''
                                END
                            IF J_ST = '?' THEN
                                DO
                                    GET_T = 'ST'
                                    PUT_T = 'FT'
                                END
                            ELSE
                                DO
                                    GET_T = ''
                                    PUT_T = ''
                                END
                            IF J_SM = '?' THEN
                                DO

```

```

        GET_M = 'SM'
        PUT_M = 'FM'
    END
ELSE
DO
    GET_M = ''
    PUT_M = ''
END
    SAY '    Enter source' PUT_N PUT_T PUT_M '[origin'
                                           J_SN J_ST J_SM']' ,
    INTERPRET PULL GET_N GET_T GET_M
    IF SN = '' | ST = '' | SM = '' THEN
        LEAVE
    END
END
END
IF C_TN = '*' | C_TN = '=' THEN
    TN = SN
    IF C_TT = '*' | C_TT = '=' THEN
        TT = ST
        IF C_TM = '*' | C_TM = '=' THEN
            TM = SM
            IF SUBSTR(OPER, 1, 1) = 'C' | SUBSTR(OPER, 1, 1) = 'S' THEN
                DO
                    IF C_TN = '?' | C_TT = '?' | C_TM = '?' THEN
                        DO
                            IF C_TN = '?' THEN
                                DO
                                    GET_N = 'TN'
                                    PUT_N = 'FN'
                                    P_TN = C_TN
                                END
                            ELSE
                                DO
                                    GET_N = ''
                                    PUT_N = ''
                                    P_TN = TN
                                END
                            IF C_TT = '?' THEN
                                DO
                                    GET_T = 'TT'
                                    PUT_T = 'FT'
                                    P_TT = C_TT
                                END
                            ELSE
                                DO
                                    GET_T = ''
                                    PUT_T = ''
                                    P_TT = TT
                                END
                            IF C_TM = '?' THEN

```

```

DO
    GET_M = 'TM'
    PUT_M = 'FM'
    P_TM = C_TM
END
ELSE
DO
    GET_M = ''
    PUT_M = ''
    P_TM = TM
END
SAY '    Enter target' PUT_N PUT_T PUT_M '[origin'
      P_TN P_TT P_TM'|'
SAY '    for source ->' SN ST SM
INTERPRET PULL GET_N GET_T GET_M
IF TN = '' | TT = '' | TM = '' THEN
LEAVE
END
END
IF TN = '*' | TN = '=' THEN
TN = SN
IF TT = '*' | TT = '=' THEN
TT = ST
IF TM = '*' | TM = '=' THEN
TM = SM
IF J_C_MODE = '' THEN
ADD_INFO = '    Remain' FORMAT(ABS(J-I+1), 3, 0) 'files'
ELSE
ADD_INFO = '    Enter empty line to exit'
SAY LEFT(SN, 9)LEFT(ST, 9)LEFT(SM, 2) ' as '
      LEFT(TN, 9)LEFT(TT, 9)LEFT(TM, 2) ADD_INFO
FCOPY SN ST SM TN TT TM FCOPY_PARMS
IF OPER = 'SPLIT' THEN
    IF MOD_TYPE = 'Y' THEN
        TT = TT + 1
    ELSE
        IF MOD_TYPE  $\neq$  'N' THEN
            TN = TN + 1
        END
SAY COPIES('>', 21) 'End ' OPER 'at' TIME() COPIES('<', 21)
SIGNAL END
ERROR:
    SAY '— The above error caused FCOPYPROabend'
END:
SET CMSTYPE HT
ERASE FCOPYPRO XEDIT A
ERASE CMS EXEC A
SET CMSTYPE RT

```

---

*Dobrin Goranov*  
*Information Services Co (Bulgaria)*

© Dobrin Goranov 1998

---

## **VM-related newsgroups and mailing lists**

In the same way that someone without a television set can be oblivious to the entertainment, news, and information being beamed through the ether – and through him – it's possible to miss some of the most important information sources on the Internet by simply not tuning into them. Before the World Wide Web, before CU-SeeMe video conferencing, and before Internet telephony, one of the Internet's main functions was to carry NNTP – Network News Transfer Protocol, a character-based information architecture. Just as it's not necessary to be an electrical engineer to watch television, one needn't be a computer scientist to use and appreciate Network News – often called Usenet.

Cable television access has increased the number of channels available from single-digit numbers to dozens or hundreds, and, similarly, Usenet has increased the information sources available to tens of thousands. At the time of writing, one of my newsreaders can receive 27,295 distinct newsgroups – each the equivalent of a 'channel', being a self-contained bulletin board covering a single topic. Just as any one person is likely to watch only a relatively few channels out of those available, the typical Internet user will only subscribe to and follow a few newsgroups (out of the total number mentioned above, I'm subscribed to 'only' 113, and never manage to keep current on them all).

Where it once required specialized software to read newsgroups, it has become much easier as popular software (eg browsers from Netscape and Microsoft) has taken on aspects of the Swiss Army knife. Choosing Internet client applications – newsreaders, browsers, TELNET clients, whatever – is largely a matter of individual preference, skills, local community, and first impressions. So the most important suggestion regarding newsgroups is to read them in the most efficient and convenient way possible – with a Web browser, with a GUI newsreader, or with a text-based CMS tool on a venerable 3270 terminal.

Another traditional Internet information resource is the mailing list –

a subscription-based facility which lets self-selected communities of interest communicate via shared e-mail, by which notes sent to a special e-mail address are relayed ('exploded') to all subscribers.

There is often disagreement about whether it's more efficient, more reliable, more economical, etc, to read information via newsgroups or mailing lists. Some argue that newsgroups are better because there is a better focus and less off-topic posting, together with control of when the information is accessed, and avoiding cluttering one's e-mail in-box with (perhaps) hundreds of individual e-mails – although many mailing lists are available in 'digest' format, grouping together a number of postings in a single large e-mail compilation. Others believe that mailing lists are better because the desired information is delivered without user action, beyond subscribing, avoiding the need to remember to visit the newsgroups before the contents expire and are deleted from one's local news server.

Fortunately, much VM-related Internet content is simultaneously available from both mailing lists and newsgroups, so users can choose the technology and tools that best suit them for reading and posting information. Similarly, information posted to either distribution mechanism is automatically replicated to the other. This facility,

21	14	bit.listserv.asm370
22	8	bit.listserv.help-net
23	1056	bit.listserv.ibm-main
24	98	bit.listserv.nettrain
25	30	bit.listserv.new-list
26	21	bit.listserv.opers-l
27	2	bit.listserv.sfs-l
28	408	bit.listserv.techwr-l
29	35	bit.listserv.tsorexx
30	11	bit.listserv.vm-util
31	73	bit.listserv.vmesa-l
32		bit.listserv.vmx-l
33	22	bit.listserv.www-vm
34	5	bit.listserv.xedit-l

*Figure 1: Section from a list of newsgroups*



called ‘gatewaying’, allows bi-directional communication between the worlds of mailing lists and Usenet.

A snippet from the list of newsgroups to which I subscribe is shown in Figure 1. This shows a number of newsgroups in the news hierarchy ‘bit.listserv’, indicating that they are gatewayed feeds from mailing lists such as VMESA-L (discussing VM/ESA), XEDIT-L (XEDIT information), and IBM-MAIN (IBM mainframes). The first number is an index in my newsgroup subscription list, while the second number shows the number of postings in each newsgroup that I have not yet read.

Since Usenet contains all the newsgroups anyone has seen fit to create (although they’re not necessarily available on every local news server, which is where users retrieve postings), a newsreader only shows newsgroups selected for reading. (The total daily Usenet data feed is several gigabytes. This necessitates selectivity in subscribing, but also encourages Internet service providers to screen out largely useless newsgroups such as university internal class discussions, and requires that information eventually expires and is deleted.) Newsreaders allow either the entering of specific newsgroup names for subscription, or viewing the entire list of available newsgroups for searching and browsing and individual subscriptions. Once newsgroups are subscribed to, they must be ‘visited’ periodically to read postings since the last visit.

The retention period of a user’s news server will govern the required frequency of visit, to avoid postings being deleted without being read.

An easy way to find and read VM-related newsgroups is to notice links on Web sites and elsewhere, and click (while using appropriate multi-function software) links that look like ‘news:bit.listserv.ibm-main’. This will open the newsreader function and display (in this example) the IBM mainframes newsgroup. After perusal to ensure that it’s worth subscribing, you tell the newsreader to subscribe.

For those who favour receiving e-mail information (a ‘push’ technology that predates the trendy term), several Web sites catalogue and allow searching for mailing lists. One of the easiest sites to use is hosted by L-Soft International, vendor of the VM-born and popular mailing list

management software *LISTSERV(r)*. L-Soft's Web page (<http://www.lsoft.com/catalist.html>) is described as "*CataList, the official catalog of *LISTSERV(r)* lists*". Since many VM-related mailing lists are managed by *LISTSERV*, this is a good place to begin. Searching for the string 'VM/' (it insisted that 'VM' was too short a search argument) yielded:

- 9370-L@NIC.SURFNET.NL – IBM 9370 and VM/IS specific topics list (66 subscribers).
- HLPCMD-L@BROWNVN.BROWN.EDU – HELP commands for VM/CMS (135 subscribers).
- REXXLIST@NIC.SURFNET.NL – VM/SP REXX language discussion list (14 subscribers).
- TRACK-L@CUNYVM.CUNY.EDU – "*Forum for TRACK utility VM/SP, VM/XA, VM/ESA*" (98 subscribers).
- UNION@LIST.UVM.EDU – Talking Union forum created by UVM/FAHC employee (62 subscribers).
- VM-REXX@VM.MARIST.EDU – VM/SP REXX language discussion list (285 subscribers).
- VMESA-L@UAFSYSB.UARK.EDU – VM/ESA discussions (511 subscribers).

This list includes the list name and description, host site, and number of subscribers. The VMESA-L list is the mailing list version of the same-name newsgroup shown above; note that some information is only available as either mailing list or newsgroup – there is no requirement or architecture that maps all of one information type to the other.

Each list name is a live link to more information about the list, formatted as shown in Figure 2.

Each 'more information' line is in fact a link. This list's configuration reveals how it is subscribed, owned, edited, etc and notes that it "*is intended to provide a forum for discussion of topics relating to the IBM VM/ESA operating system*". The list owner link allows you to send e-mail to Dan Martin, who owns and supports the list, and the last

VMESA-L@UAFSYSB.UARK.EDU  
VM/ESA Discussions

List name: VMESA-L  
Host name: UAFSYSB.UARK.EDU  
Subscribers: 511  
Features: Spam filter  
Hands free bounce processing  
Archives  
Digests (with MIME support)  
Indexes  
Database functions

To subscribe, send mail to [LISTSERV@UAFSYSB.UARK.EDU](mailto:LISTSERV@UAFSYSB.UARK.EDU) with command  
(paste it!): SUBSCRIBE VMESA-L

For more information, you can:

Take a look at the list's configuration  
Contact the list owner at [VMESA-L-request@UAFSYSB.UARK.EDU](mailto:VMESA-L-request@UAFSYSB.UARK.EDU)  
Check the other lists at [UAFSYSB.UARK.EDU](http://UAFSYSB.UARK.EDU)

*Figure 2: Example of a live link format*

link offers other lists hosted at the same site. Since multiple related lists are often hosted together, this can be a quick way to locate unknown but useful lists. The 'To subscribe' link automatically begins composing e-mail to subscribe to the list, requiring only pasting in the Subscribe command (and deleting any signal file automatically included in e-mail). Since this list's configuration shows that subscription is 'open, confirm', it will automatically accept subscriptions (not requiring Dan's action to approve) – subject to new subscribers following the simple instructions provided in an automatic e-mail response to confirm subscribing. (This is a low-cost means of avoiding bogus or nuisance subscriptions, since it requires the confirmation of a new subscription from the address being subscribed.) This list's configuration specifies that all postings are archived in notebooks available to the public; the list of notebooks can be retrieved by sending the command 'index VMESA-L' to the designated server, [LISTSERV@UAFSYSB.UARK.EDU](mailto:LISTSERV@UAFSYSB.UARK.EDU). Desired files can be

```

* VMESA-L FILELIST for LISTSERV@uafsysb.uark.edu.
*
* Archives for list VMESA-L (VM/ESA Discussions)
*
* NOTEBOOK archives for the list (Monthly notebook)
*
*                               last - change
* filename filetype nrecs   date      time   Remarks

VMESA-L  LOG9401    3897 94/01/31 23:57:41 Started Mon, 3 Jan 1994
.
through
.
VMESA-L  LOG9803   15991 98/03/22 10:14:11 Started Sun, 1 Mar 1998

```

*Figure 3: Example of returned index command*

retrieved by sending the server commands such as 'get vmesa-l filename' where 'filename' is one of the files returned by the index command. Even if you prefer to read information via a newsreader,

```

bit.listserv.ibm-main (125T 353A 0K 0H)          h=help

49 + 2  Y2K and Budget plans                      Steven
50 +    3494 sharing between LPARS -Reply          James W. Fletcher
51 +    :)                                          aar@centrin.id.com
52 + 5  Defrag on RVA.                             Ed Gould
53 + 3  Migration to RVA                          David_G_Clark
54 + 2  IBM-MAIN History (was Re: Learning the Langua Barry Finkel
55 + 10 Relative Power of CPUs                     Phil Payne
56 + 2  IBM-MAIN History (was Re: Learning the Langua James Harrison
57 + 2  inbox filtering WAS: Learning the Language Gary T Kobs
58 + 7  Application Development Survey             Bob Shannon
59 +    PDS 8.5                                    John P. Kalinich
60 +    IBM-MAIN History                          Leonard D. Woren
61 + 2  inbox filtering                          Leonard D. Woren
62 + 2  Are Deleted Datasets Recoverable on Raid Boxe Sam Golob
63 +    Voice Response                            Rob Luebke
64 +    Fw: WLM dispatching priorities            Wayne Montefiore

```

*Figure 4: Sample index screen*

using index/get commands allows browsing backwards and catching up on content that has gone before and aged off the news server. In this case, the index command returned (slightly condensed here) is shown in Figure 3. The 'get vmesa-llog9803:' command returned a 700,000+ byte file with nearly 16,000 lines – the partial mid-March contents – containing more than 300 postings on dozens of topics, among them:

- 2nd Level Performance
- 64 Bit
- 9021 IOCP
- Accounting records
- AS/400 to VTAM connection
- Best Platform for ADSM: VM or MVS?

```
Wed, 18 Mar 1998 15:32:20 bit.listserv.ibm-main      Thread  60 of 125
Lines 20                      Re: IBM-MAIN History      No responses
ldw@LINDY.STANFORD.EDU      Leonard D. Woren
```

```
> I seem to remember during one of the geezer-fests here, someone
> mentioning 1986. Ring a bell with anyone?
```

```
I subscribed in mid-1986 and I think IBM-MAIN wasn't too new then.
Although in those days we tended to know the names of all the regulars
on the list...
```

```
IBM-MAIN is one of the original Bitnet-only discussion lists. In those
days it wasn't easy to get access to IBM-MAIN unless you were on Bitnet.
```

```
The traffic level on IBM-MAIN in 1986 was typically a few messages per
day. (I still have all that stuff on tape somewhere.) It would be nice
if people, before posting, would think "does this reply add value to the
discussion?", and not post if the answer is "no". The traffic level now
is very high, and too much of it is noise. The s/n ratio used to be much
much better before Internet access got to be nearly universal.
```

```
/Leonard
```

*Figure 5: Thread 60*

- CA-Dynam/T/VM
- CP TERMINAL SCRNSAVE ON alternative.

Returning to my newsreader and viewing the IBM mainframes newsgroup, whose link is shown above, a sample index screen is shown in Figure 4. The header shows that my news server offers me 125 separate topics (conversation ‘threads’) totalling 353 postings. Below the header, in each line, the first number is the position of the

1	+	VM/XA CMS filesys problem	Michael R. Walter
2	+	Best Platform for ADSM: VM or MVS?	Vadim Rapp
3	+ 2	In Praise of SNA	Gregg Levine
4	+	VTAM: SNA Subarea or APPN Network	Joseph P. Nelson
5	+ 3	Where do we post CMS Pipeline questions?	Dan Wenzinger
6	+	Comparing hex values in REXX (was: REXX timeb	Bob and Deb Levad
7	+ 5	VM/VTAM Question	Conrad R. Sanders
8	+ 5	Erasing files whose file names have mixed cas	Terry A. Moore
9	+ 2	Best way to implement SET MORE and SET HOLDIN	Romney White
10	+	erasing a mixed-case file	Byron D. Graham
11	+	CP send DOES!!!	Raul Fliman
12	+	X\$EXCM\$X EXEC Was (Erasing files whose file n	Colin Allinson
13	+ 4	Unofficial, Me Only Survey	Daniel A. McLaughl
14	+	IN PRAISE OF SNA	Tom Duerbusch
15	+ 3	CP TERMINAL SCRNSAVE ON alternative	Raul Fliman
16	+	Data Warehouse Software	Peter Carrier
17	+ 15	CMS End-Of-Command Cleanup	Simon Bishop
18	+ 4	VM/VTAM question	Nancy Reeves
19	+ 4	VM/VTAM QUESTION	Tom Duerbusch
20	+ 2	WAVV User Groups	Ron Campbell
21	+ 4	Concerned over the future of VM and VSE? Let	Rich Smrcina
22	+	VTAM menu	Bill Pettit
23	+	WAVV ?	Ron Campbell
24	+	Reminder to enroll for the next two VM Teleco	Pamela Christina
25	+	ftp through a firewall	Byron D. Graham
26	+	Getting FAL current – what are your favorite	Melinda Varian
27	+ 2	CMS 13	Helen P. Nulty
28	+	CA-Dynam/T/VM	Thigpen, Tony
29	+	e1 cheapo p390's....	Barton Robinson
30	+	Security mailing list	noname
31	+	Security software	noname
32	+	Differences between O.S.	noname

*Figure 6: Example of the bit.listserv.vmesa-1 newsgroup*

Does anybody know if there is a documented CMS programming interface to the "End-Of-Command Cleanup" functions which are performed by CMS after each console command has been executed?

I am looking for a way to perform some or all of this cleanup from an assembler program, without using unsupported interfaces (or worse, copying code from DMSINX).

*Figure 7: Example newsgroup question*

topic in the newsgroup, the plus sign (+) indicates that a topic has postings I have not read. The next number is the number of unread postings, followed by a brief topic and the name of the person who began the thread. A bit of IBM-MAIN history and perspective on its traffic is revealed by thread 60. This is shown in Figure 5.

Viewing the bit.listserv.vmesa-l newsgroup shows 32 threads with 55

\* March 26, 1998            VM/ESA (V2/R3): Early Customer Experiences  
10:30am ET (408167) or 2:30pm ET (408175)

VM/ESA Version 2 Release 3.0 has been in customers' hands for several months now, in an IBM Field test called: Early Support Program. Customers have installed and used new VM/ESA v2R3.0 capabilities. Call in to hear what customers think about the new release and pick up some tips on how you can get the most from the new VM/ESA release ASAP!

\* April 23, 1998            VM/ESA Guest Support  
10:30am ET (408181) or 2:30pm ET (408190)

VM's most popular feature, the ability to host guest S/390 operating systems, has undergone major advances in the last few years. This call will focus on the VM technologies that simplify the installation, testing, and operation of your other environments (including their Year 2000 testing). We will also explore ways in which you can use VM to interact with your guest systems, to get more from their synergy on a single processor or a multiprocessor.

*Figure 8: Example posting from IBM*

unread postings. This is shown in Figure 6.

Following this information source – via mailing list or newsgroup – immerses the viewer in the VM community, bringing experts close at hand, and allowing you to get to know colleagues near and far. On-line VM community resources are a good antidote to that feeling that you are alone with problems and questions, or at the mercy of vendors who say that no-one else is reporting a particular problem. For example, the longest thread in the bit.listserv.vmesa-1 newsgroup asks a question about CMS End-Of-Command Cleanup (Figure 7). This has resulted in 14 responses (so far) offering feedback and suggestions, and discussing details of the application being developed. Another type of posting comes from IBM, publicizing VM teleconferences. This is shown in Figure 8.

Visit IBM Enterprise Connection at its new home (<http://www.s390.ibm.com/events>) to read descriptions and registration information, and enrol on-line. This is shown in Figure 9.

Visiting newsgroup comp.lang.rexx (a more formally recognized newsgroup, not necessarily available as a mailing list) yields 30 threads with 80 postings, including those shown in Figure 10.

This shows the diversity of both topics and participants, with several people well-known as authors or software developers. On-line information sources impose the risk of drowning in data while seeking knowledge, but many resources simplify targeted searches, such as DejaNews (<http://www.dejanews.com>) for searching newsgroup contents. VM's traditional strengths as a robust and scalable

\* No charge to US and Canada customers and business partners.

Enroll via internet: <<http://www.s390.ibm.com/events>>

Enroll by phone: 1-800-289-0583 (priority code 6N8AD001)

Customers outside the US and Canada, can find more information at:

<<http://www.s390.ibm.com/events/note.html>>

Enroll 1-719-386-0025

*Figure 9: IBM Enterprise Connection*



2	+ 2	Determine the drive OS/2 resides on?	Dick Goran
3	+ 3	Problems with subscribers?	Brian {Hamilton Ke
4	+ 7	Inter Thread/Process communication	neworder@flash.net
5	+	Using an environment variable to specify a co	bparrill@usa.net
6	+ 7	porting Vispro/REXX OS/2-programms to Window	doug@hotrocks.msfc
7	+ 2	REGINA - sysgetkey	mcmack@usa.net
8	+	Receiving messages from PMCX controls via an	mowens@bigfoot.com
9	+	ObjRexx to query NT Registry	Ed Stevens
10	+ 2	Edit macros in batch MVS	J.D. Hill
11	+ 5	REXX date converter	hessling mark
12	+ 4	numeric fuzz	doug@hotrocks.msfc
13	+	Good NETREXX info	rr14@yahoo.com
14	+	least squares polynomial regression & linear	doug@hotrocks.msfc
15	+	free REXX implementation on Windows, DOS	hessling mark
16	+ 3	Minimizing a window from REXX (OS/2 Warp v3)	Laurie Chan

*Figure 10: Example from comp.lang.rexx*

connectivity-oriented platform are enhanced by tools such as Charlotte (available from <http://ukcc.uky.edu/%7Etools.1997/>), described as:

*“....providing text-mode full screen access to the World Wide Web from 3270-type screens, including support for Gopher and FTP (File Transfer) protocols and network news (NNTP) support.”*

The page listed above, (with /7E used by the server to represent the tilde (~) character) links to Charlotte’s read-me file, which illustrates Web and newsgroup capabilities:

- New more powerful HTML parser and formatter (written in C/370).
  - based on HTML 3.2 specification.
  - supports tables as defined for HTML 3.2, including nesting.
  - supports normal text formatting (indents, lists) inside tables.
  - adjusts table cell widths to make best use of screen width.
  - provides links to individual frames if document uses

## FRAMES.

- provides links for client-side image map AREA tags.
- Network news reading support:
  - tracks the last article read in newsgroups being followed.
  - lists current groups being followed if no newsgroup is specified.
  - supports leading/trailing wild card for newsgroup name list.
  - provides links to referenced articles.

Regarding CMS Internet clients, David Boyes of Dimension Enterprises in Herndon, VA, commented: *“The only ones I know of that fit the newsreader category are shown below.*

*PSUNews (Penn State) – full news system based on XEDIT macros and a lot of really wild DVMs written by Linda Littleton at PSU. Full news system, including incoming/outgoing newsfeeds, etc.*

*RNEWS (last known site was at Mitre, I don't know where to get it now that Mitre's VM systems are gone) – CMS Pipes and RXsocket based, very fast, NNTP client only implementation.*

*YNEWS (last known site was UCSF – Yossie Silverman's work) – Pipes and RXSOCKET based, very fast, NNTP client only, somewhat buggy the last time I saw it. Rick Troth and I came up with a bunch of changes, but never got them fully working before we both left Rice University.*

*As far as WWW browsers go, Albert and Charlotte are still top of the line, with Charlotte being vastly better. CMS Gopher is still available from Rick Troth, along with other TCPSHELL based servers and clients at <http://ua1vm.ua.edu/~troth/vm>. URI's POP server is still viable, although the version done at Temple University is a little more robust with the newest IBM C libraries.*

*Some commercial software is now available for WWW stuff, ie Sterling Software's Web server, Beyond Software's revision of Rick, and my old WEBSHARE code.”*

An outstanding commercial-product CMS-based e-mail client is MailBook(r), described as “*EMail for VM!*” at <http://www.mailbooksoftware.com>: “*MailBook, based on ten years of experience with electronic mail for IBM VM/CMS systems, is the premier electronic mail package for connecting VM systems to the Internet. MailBook includes two commands:*

- “*MAIL which reads and sends electronic mail.*
- “*MAILBOOK. which manages NOTEBOOK files created by the MAIL command.*

“*Mailbook supports MIME (Multipurpose Internet Mail Extensions), an Internet standard for e-mail, which specifies how to encode information beyond standard text composed of the 95-character US-ASCII character set. This standard has been implemented for many different computing environments.*”

The judicious combination of subscriptions and searches can magnify one’s productivity and greatly enlarge one’s circle of colleagues.

*Editor’s note: if you have comments on the Web sites reviewed in this series, or suggestions for relevant sites to review, please feel free to contact the author at [gabe@acm.org](mailto:gabe@acm.org) or Xephon at any of the addresses shown on page 2.*

---

Gabe Goldberg  
Computers and Publishing (USA)

© Xephon 1998

Subscribers who want copies of the code from this issue can call our Web site – [www.xephon.com](http://www.xephon.com) – and ask for the article they require. The article will then be e-mailed to them. This service is free to subscribers.

The first time you visit the site, you will need to register. For this, you will need the user-id, which appears on the label of the envelope in which you receive your issue of *VM Update*.

When you register, you will be asked to choose your own password and user name, which will be recognized when you log on again.

## VM news

---

VM users can now benefit from EnterWEB Version 3.7, announced by Macro 4, which enables direct access to information on the Internet and company intranets from 3270 terminals and PCs with 3270 emulation.

EnterWEB now supports OS/390, VM, and VSE. A new FTP download function enables users to obtain documents and files that are available on individual HTML pages. The MAILTO feature allows users to respond to particular items on Web pages, effectively providing e-mail capabilities from EnterWEB to individual destinations.

The HTTP Server function allows a user to serve his/her own pages within EnterWEB, enabling the product to be customized to meet individual user requirements. EnterWEB now gives 3270 users access to 'Web-enabled' information systems that have HTML servers including Lotus Domino and Microsoft's Exchange Server.

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 a range of new messaging upgrade packs, available for OV/VM, OV/MVS, and OV/400. Bundled software is aimed at getting users of other e-mail systems to migrate to Notes and Domino.

The packs for OV/VM and OV/MVS include a customized Lotus Messaging Switch

(LMS) for message switching and directory synchronization, the Domino Migration Engine, SMTP gateway, Lotus Calendar Connector for Office Vision, and a Domino server.

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

\* \* \*

IBM has announced enhancements to VM and VSE and introduced a new model to its IBM System/390 Multiprise 2000 family.

The new servers support VM/ESA, VSE/ESA, and OS/390 operating systems and include internal disk capacity up from 288GB to 576GB, cryptographic coprocessor enhancements, availability of OSA-2 Fast Ethernet, and ATM enhancements for improved availability and extended SNA/APPN network management.

VM and VSE users now have the option to run Unix applications on the System/390 platform. Users can continue to run VM and/or VSE and take advantage of OS/390 functions and applications without the requirement to install or operate these applications in the OS/390 environment.

For further information contact your local IBM representative.

\* \* \*



# xephon