

March 1997

In this issue

- 3 VM Web servers
- 10 Managing back-up tapes
- 20 Capturing lines written to the user's console
- 37 Dynamic menus system for CMS – part 2
- 39 Performance hints and tips
- 40 XEDIT extensions
- 52 VM news

© Xephon plc 1997

update

VM Update

Published by

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

North American office

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

© Xephon plc 1997. 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.

VM Web servers

The Internet has exploded across the computer industry like an unopened can of beans on a hot stove. It is beginning to be just as messy too! You cannot pick up any IS publication without seeing some article or study showing the way the Internet (or an intranet) can help your business make more money or provide better access to company information. Even one of the biggest executive decision-making publications has had articles on the Internet. That publication is, of course, the airline magazine you find in the seat pocket in front of you when you fly. Everywhere you look, even at billboards while driving on the highway, you find 'WWW' addresses for companies you wouldn't have guessed would ever be on the 'Web', such as the local pub or even a gas station. Companies are jumping on the Web by the hundreds, from Fortune 500 size all the way down to the single owner-operator.

So what does VM have to do with the Internet? Companies have invested in VM to manage their e-mail systems and to prototype, develop, and run production applications among many other things. But there isn't any good reason to build an Internet/intranet on VM. VM is a legacy system based on 'old' technology and is on the way out the door to make room for the 'new' technology that is more functional, reliable, and secure, right?

Not! The definition of a legacy system is one that works. In that respect VM is a legacy system, but it is not 'old' technology. VM's features and power have grown by leaps and bounds over the last few years, and it has been proven time and time again that VM is cheaper to run and manage, and is more secure than 'distributed' platforms. Another power you can give to VM is the ability to run a Web server and build an intranet or even connect it to the Internet. One of VM's strengths is its ability to run not only interactive sessions but high-power server virtual machines. Ones that can stand up to the stresses that a Web server can demand. And if one is not enough, define another — or ten.

Many organizations, or better stated many VM systems programmers,

are trying to find ways to revitalize VM and show their management that VM still has a place in their IS structure. VM is losing ground to the more graphical, yet sometimes less functional and less dependable, e-mail systems, and needs whatever it can get to help prove its continued worth. 3270 access is also taking a back seat to the newer GUI interfaces on LAN-based applications. That has been one of VM's weaknesses – its 3270 character-based interface.

There have been attempts to build GUI interfaces to OV/VM specifically, and recently to any VM application via CMS/GUI. CMS/GUI is OK and a number of very good people worked hard on it, but it's a little too cumbersome and limited, especially with the few clients it works on.

There is an easier and more cost-effective way to build a GUI interface to VM applications and provide access to data – use a VM Web server.

Getting started is actually quite easy, especially since there is a very good VM Web server package available for free called WEBSHARE, written by Rick Troth. You can get the code from the Internet from many locations, including <http://www.beyond-software.com/Software/Software.html>. You will need TCP/IP running on your VM system, RXSOCKETS – a REXX to TCP/IP socket package written by Arthur Ecock of City University of New York (included in VM/ESA 2.2.0, CMS 13), along with VMARC to unarchive the packages. All are available from the same location.

WEBSHARE offers support for serving almost any data/files you will need to make available, and is fairly easy to configure. It supports mini-disk or SFS directories, CGI scripts, and many other things that you will need for a Web server. About the only weakness to WEBSHARE is the lack of documentation that comes with it. You will have to dig a little to get the information you need to configure the Web server and all the required parts. Tip: look at CMSHTTPD README and also do a HELP HTTPD command. These will point to other tit-bits of information. Also look in the files HTTPD CONFIG, WEBSHARE FILELIST, HTBIN FILELIST. The discussion lists mentioned later will also help if you get stuck.

There are industrial strength Web servers available too that you can obtain for a trial/demo. One is from Sterling Software called VM:Webserver, and another is from Beyond Software called EnterpriseWeb. Both are excellent solutions depending on your planned use of or needs for the VM Web server. It is best to define your needs and then review the two products to determine which one best fits those needs – as you would with any other software product.

If you don't already have a VM Web server, focus on WEBSHARE because it's easy to get and, again, it's free. It will give you a good start to show off what VM can do as a Web server, and possibly even be a foundation for one of the commercial packages mentioned above. WEBSHARE also gives you time to work with it a while and show it to your management as time permits (make time for it though!) instead of being under the pressure of a trial version from one of the vendors. Did I mention WEBSHARE is free, too?

One other great tool you can get your hands on for free is Charlotte, written by Carl Forde. Charlotte is the best Web browser you can get for VM. Charlotte is also easy to obtain from the same URL mentioned earlier in this article. If you have access to the Internet from your VM system, you can install and configure Charlotte and be surfing from VM in no time. It works great at intranet-only sites too.

Charlotte is particularly handy when you are out of the office and don't have access to an Internet-capable computer, but you can get dialled into your VM system. From VM you can browse the pages you need, download files, and post messages whenever you need to. Or for those of us 3270 junkies that live on VM, Charlotte is a lot quicker than launching one of the many graphical Web browsers on your desktop OS.

Charlotte has been updated recently to include Table support along with other new features. It has had, and continues to get, a lot of attention. People offer suggestions, and even code at times, to improve Charlotte and make it feature-rich. Usually those suggestions/code updates end up in the next release very quickly.

One of the ways to keep up with the updates to Charlotte and WEBSHARE, and to get information, hints, and tips, is to join one of

the many discussion lists available. One discussion list that gives a lot of attention to WEBSHARE and Charlotte is WWW-VM. Other good lists to join are: CMS-PIPELINES, IBMTCP/IP, and VM-REXX.

Joining is easy too. You can join the discussion list relating to the World Wide Web on VM (WWW-VM) by sending a note containing the single line:

```
SUBSCRIBE WWW-VM Your_Real_Name
```

to listserv@sjuvm.stjohns.edu. You should receive a reply shortly saying you were added to the list and giving you instructions on how to post messages/questions to the list.

These lists offer access to dozens (or more) people including the authors of those tools. There is enough expertise available through the lists to answer almost any question. You can post just about anything on the list as long as it has some relationship to the discussion list topic – even if the subject is only distantly related. There have been many very long and controversial discussions on various topics, but they have all ended up being not only exciting but informative. Remember, no question is too trivial to ask. You may be only one of many who wanted to ask the same question or needed the same information.

Now that you have the tools and the means to talk to others about VM and its role in the Internet, what can you do with them to help show VM is still worth keeping around?

As mentioned previously, the weakness most VM shops are dealing with is that 3270 access to applications/data is quickly getting shoved out the door in preference to GUI interfaces. It is a common, yet false, thought that VM can't do a GUI interface. This is why OV/VM is getting replaced by cc:Mail, Lotus Notes, or POP mail servers even though those packages don't come close to the features, security, and cost-effective administration of OV/VM. Someday they all may catch up with OV/VM, but they have a long way to go. They do have a GUI interface with all the neat buttons, pictures, and pull-down menus, though. Wow!

VM can play in the GUI world, however. Both Sterling Software's and Beyond Software's Web servers also include what I like to call OV/

VM GUI access interface plug-ins. The vendors extended their Web server products to include CGIs that work with OV/VM. They give you the buttons, menus, and cool pictures that people want to have, while using OV/VM as the engine.

If your goal is to keep OV/VM around a while longer, this would be a good reason to work with both vendors to get a demo or even trial code. The Web interface is not only 'cool' to use, it works, and people love it! You'll amaze your peers, co-workers, and management with it. If a Web interface to OV/VM doesn't make some eyebrows raise up and at least get people (re)thinking about the direction of VM, then they're living in a vacuum.

Whether you use WEBSHARE or obtain one of the vendor Web server products, you can also easily show how VM can offer graphical interfaces to other applications or data you may have.

The first thing I would recommend doing is setting up a Web page structure with some data/information that is normally only available via 3270. You can either wrap a little HTML around the data, or write some simple CGI scripts to do that for you. You can then show that access is not restricted to 3270 and that VM can continue to process the information with little or no changes to the data. Using the Web beats converting to another platform the application that creates the data, the location of the data, and the means to access it.

The next step is to pick an application you have running on VM that a lot of people use. The one aspect of the application that you will want to check for is that it has some type of line-mode or API (Application Programming Interface) interface. In other words, can the application receive instructions and return results by using commands from the command line or using API function calls from REXX. OV/VM for example has a line mode interface; almost every function of OV/VM can be done without a 3270 screen. IBM's CallUp product also has a line mode interface and database API-like functions.

Once you have selected the application, write a CGI to interface to a portion of it. You do not have to write the CGI(s) to give full access to the application unless you really want to. The goal is to show that

'it _can_be' done! Start off small, and keep it fairly simple. Too many graphics can deter from the page, as can too many buttons, menus, and entry fields. Browse around the Internet and you'll find plenty of page layouts that are effective. You can even emulate those layouts to fit your application.

If you have the opportunity, you can expand on the application using the Web interface. For example, IBM's CallUp product is for name, address, phone (etc) directory information including who reports to whom. A little CGI I wrote is a CallUp interface to build GUI organizational charts from CallUp. The most difficult part of the process was getting the graphics to line up on all the different browsers! It was just a simple little CGI, but people loved it and it gets used frequently.

Once you have the application Web interface done, show it off. Even if your management doesn't have time or care to see it, show it to people you meet in the elevator or people who talk to you. Ask them if they want to see something that is the newest in technology and that's really neat. (It is new because, technically, you just built it!) Let them use it from their own desk if they want, ask them for feedback, and keep track of it to share with your management. In most cases, you can prove that building a Web interface to your existing 'legacy' VM applications is far more cost-effective than rewriting them to run on a different platform. You can keep the application where it is (where it belongs!) and continue to benefit from the existing function, security, and administration while providing the GUI interface to your users.

It's really amazing to show people the Web pages and application interfaces and see their reaction when they hear it's all running on VM. I had a good friend, who is now an independent consultant for Unix shops, ask me what I've been up to, and I told him that I had been building an intranet and Web servers. I showed him some of the things I have done so far and he was impressed and asked if he could see the PERL (Unix) scripts I used. I let him know at that point that everything he was using and seeing was coming from the VM system. The look on his face said it all, but he responded anyway: "VM can't do Web stuff". In fact it can, and does it very well too! He was amazed and even admitted that he has a new appreciation for what VM can do.

VM is not going to last forever, nor will having a Web server alone keep VM around. Using all the power and features of VM and exploiting them as much as possible will help. A lot of the problems VM is facing can be corrected with a little education, but it's up to us to do the educating. We know the tools, applications, and services that VM does well, but that information usually does not make it to the people that make the IS decisions. VM hasn't had the glossy (or GUI, if you like) flyers that have hyped the other platforms and software.

Creating a Web server on VM is one way to use VM to its fullest potential and offer a service to your users that they can enjoy and benefit from. A VM Web server will allow them to make their information available to people in a 'cool' format which has a great impact. That 'cool' format will make an impact on how the information is used and understood.

So don't let the waves crash down around you – grab a board and hang ten! Build and exploit a VM Web server!

James S Vincent
Software Consultant (USA)

© Xephon 1997

VM Web servers

We would be very interested to hear from VM sites that are using Web browsers either to access the Internet or internal intranets. Tell us how easy you have found using the browsers etc, and tell us about any problems you encountered along the way – and the solutions you tried. We'd also like to publish any code or CGI scripts you needed to write to get things working smoothly.

Managing back-up tapes

In a normal operating environment it would be nice to maintain a catalog of DDR tapes on the tape management system's own tape catalog. Unfortunately the IBM disk-to-tape dump/restore programs (DDR and IPL DDRXA) are not easily interfaced to a tape handling system. UPDATDDR and WRITLINE, however, make such an interface comparatively easy.

The original programs from IBM (DDR MODULE and IPL DDRXA) cannot be used in connection with magnetic tapes that have an internal label. The DDR MODULE (and IPL DDRXA) will overwrite those labels and, further, the programs can only be set up to bypass the label on the first tape of a multitape sequence.

In order to use magnetic tapes with an internal label and also to be able to use tape-handling software, eg VM:TAPE from Sterling Software, while performing DDR dump and restore jobs, some changes have been introduced to HCPDDR. By using UPDATDDR EXEC these changes will be made in a way that will not interfere with any updates that may be provided by IBM in the future. Such updates will, however, be included in the two modules created by UPDATDDR.

The new modules have been given the names XDDR MODULE and IPL DDRSM. All changes to HCPDDR apply to XDDR as well as to DDRSM. In addition, a patch is applied to the newly created XDDR MODULE (changing the name of the screen write module from LINEWRT to WRITLINE).

The specifications for DDR are still valid for XDDR, with the exception of specifying a minimum skip of 1, ie:

```
IN 181 3480 (SKIP 1 UNLOAD
```

while performing a DDR restore, and that this skip count will not be cleared to zero for the second and subsequent tapes.

In addition to XDDR MODULE and IPL DDRSM, this system consists of UPDATDDR EXEC and WRITLINE ASSEMBLE.

Both UPDATDDR and WRITLINE are well documented inside the respective programs.

UPDATDDR EXEC

```

/*
*****
*
* UPDATDDR EXEC      is used to build the new programs (XDDR and IPL
*                    DDRSM).
*                    UPDATDDR requires one parameter, which might be
*                    'UPDATE', 'GENMOD', or 'GENIPL'.
*
*                    UPDATDDR EXEC must run on MAINT.
*
* UPDATE            puts you in XEDIT UPDATE for HCPDDR. Creates HCPDDR
*                    UPD001 containing the local updates. HCPDDR UPD001 will
*                    be copied to MAINT's 2C4 disk (E disk).
*
*****
*
*                    The following updates have to be applied to HCPDDR.
*
* 1.  Log-on to MAINT.
*
* 2.  Type UPDATDDR UPDATE.
*      When, after a few seconds, in XEDIT:
*
* 3.  Locate the POSTAPE label.
*      Copy the complete POSTAPE routine including the comment lines
*      on top and down, including the EJECT line.
*      This copy goes to:
*      Locate the ERR709 label.
*      Go down to EJECT and place the POSTAPE sequence after this
*      line.
*
* 4.  In the copied POSTAPE change POSTAPE to POSTAPE1.
*      Change NEXTFSF1 to NEXTFSFX (two places).
*      Remove the line
*           STH    R2,IOSKIP      Set IOSKIP to zero
*
* 5.  Locate the POSTAPE label again.
*      Insert the line:
*      LH      R2,=H'1'          For Labelsip on next tape(s)
*      after:
*           BCT    R2,NEXTFSF1    DO IT FOR EACH FILE
*      change the comment on next line from:

```

```

*                               Set IOBSKP to zero                               *
*   to:                                                                    *
*                               Set IOBSKP to one                             *
*                                                                           *
* 6.   Locate the TPSWPA label.                                           *
*       After this line insert:                                           *
*       BAL    R14,POSTAPE1      Position after the label                 *
*                                                                           *
* 7.   Locate TVOLMSG label                                               *
*       Comment out these three lines (* in pos. 1)                       *
*       LH     R1,INCC      Point to the cylinder add                     *
*       LH     R2,INHH      Point to the head address                     *
*       BAL    R14,MSG005    and type the end of vol msg                 *
*                                                                           *
* 8.   Locate TPSWP label.                                                *
*       Copy the three lines just commented out under (7) and place      *
*       the lines after the line following the TPSWP label (after        *
*       PERFORM STARTIO,SIOEP) and remove the comments (*s).             *
*                                                                           *
* 9.   File the changes.                                                  *
*                                                                           *
*****
*
* GENMOD      assembles HCPDDR and builds the module XDDR.               *
*              As XDDR cannot run above 16M, the virtual size of MAINT    *
*              must be set lower then 16M before UPDATDDR is called.      *
*                                                                           *
*****
*
* GENIPL      creates a new textdeck with the aid of the PRELOAD MODULE.*
*              Generates an IPL-able card-deck on the virtual reader.      *
*              Receives this card-deck to the A disk and changes the      *
*              entry address on the END card to make the program runnable.*
*                                                                           *
*****
*
* UPDATDDR requires the following files, which must reside on MAINT's    *
* 2C4 disk (E disk).                                                       *
*                                                                           *
* UPDATE:     HCPDDR AUXFILE This file will be renamed to HCPDDR AUXLCL *
*              at call time. It will be renamed back to                  *
*              HCPDDR AUXFILE upon completion.                            *
*              Content:  UPD001 in column 1.                               *
*                                                                           *
*              HCPDDR UPD001 Will be created on first call, and has       *
*              UPDATE as filetype. It will be copied                      *
*              to 2C4 disk with filetype UPD001 (If HCPDDR*              *
*              AUXFILE have been made), and erased from                  *
*              the A disk.

```

```

*****
*****
*
*           All other files are on MAINT's system disks and are
*           delivered with the VM system.
*
* GENIPL:   HCPDDR AUXFILE   Same as for UPDATE.
*           HCPDDR UPD001    Same as for UPDATE.
*           DDRLOAD EXEC     Content is 3 lines as specified below.
*
*           -----
*           |  &CONTROLE OFF  |
*           |  &1 &2 &3 3CARD LOADER  |
*           |  &1 &2 &3 LOADDDR TEXT  |
*           |-----|
*
*           LOADDDR EXEC     Content is 5 lines as specified below.
*
*           -----
*           |  &1 &2 HCPDDR TEXT  |
*           |  &1 &2 HCPDNC TEXT  |
*           |  &1 &2 HCPDDC TEXT  |
*           |  &1 &2 HCPDNT TEXT  |
*           |  &1 &2 HCPDDT TEXT  |
*           |-----|
*
*           All other files are on MAINT's system disks and are
*           delivered with the VM system.
*
*           The system has been tested on VM/ESA 2.1 but should
*           function also under VM/ESA 1.0 and 1.1.
*
*****
*/
arg update
if update = '' then exit
select
  when update = 'UPDATE' then          /* Make the local updates */
  do
    'EXEC VMFSETUP ESA CP'
    'RENAME HCPDDR AUXFILE E = AUXLCL ='
    'COPYFILE HCPDDR ASSEMBLE P == A (UNP OLDD REPL'
    'XEDIT HCPDDR ASSEMBLE (UPD CTL HCPVM'
    'ERASE HCPDDR ASSEMBLE A'
    'RENAME HCPDDR AUXLCL E = AUXFILE ='
    'SET CMSTYPE HT'
    'STATE HCPDDR UPDATE A'
    if rc = 0 then

```

```

do
    'COPY HCPDDR UPDATE A = UPD001 E (OLDD REPL'
    'ERASE HCPDDR UPDATE A'
end
'STATE HCPDDR UPD001 A'
if rc = 0 then
do
    'COPY HCPDDR UPD001 A = = E (OLDD REPL'
    'ERASE HCPDDR UPD001 A'
end
exit
end
when update = 'GENMOD' then                /* Generate XDDR MODULE */
do
    'EXEC VMFSETUP ESA CP'
    'RENAME HCPDDR AUXFILE E = AUXLCL ='
    'VMFASM HCPDDR HCPVM (CTL'
    'RENAME HCPDDR AUXLCL E = AUXFILE ='
    'RENAME HCPDDR TXTLCL A = TEXT ='
    'LOAD HCPDDR'
    'GENMOD XDDR'
    'ERASE HCPDDR TEXT A'
    queue 'CHANGE /LINEWRT /WRITLINE/ * *'
    queue 'FILE XDDR MODULE A'
    'XEDIT XDDR MODULE A'                  /* Change the names */
    exit
end
when update = 'GENIPL' then                /* Generate IPL DDRSM */
do
    'CP SPOOL PUN *'
    'EXEC VMFSETUP ESA CP'
    'RENAME HCPDDR AUXFILE E = AUXLCL ='
    'VMFASM HCPDDR HCPVM (CTL'
    'RENAME HCPDDR AUXLCL E = AUXFILE ='
    'RENAME HCPDDR TXTLCL A = TEXT ='
    'ERASE IPL DDRSM A'
    'PRELOAD LOADDDR'
    'HCPLDR DDRLOAD (NOCTL PUNCH'
    'MAKEBUF'
    'EXECIO * CP (LIFO STRING Q RDR * ALL'
do while queued() > 0                      /* Receive completed IPL deck */
    parse pull . spoolid . . . . . fn ft .
    if spoolid = 'FILE' | spoolid = 'RDR' then iterate
    if fn = 'DDRLOAD' & ft = 'IPL' then
do
    'RECEIVE' spoolid 'IPL DDRSM A (REPL'
    leave
end
end
end

```

```

'DROPBUF'          /* Complete the deck with correct entry address */
'EXECIO * DISKR IPL DDRSM A (STEM DDRSM.'
'FINIS IPL DDRSM A'
'ERASE IPL DDRSM A'
i = 0
do while ddrsm.0 > 0
  ddrsm.0 = ddrsm.0 - 1
  i = i + 1
  if pos('ESD',ddrsm.i,2) > 0 then
    do
      if pos('HCPDDREP',ddrsm.i) > 0 then
        do
          addr = pos('HCPDDREP',ddrsm.i) + 9
          addr = substr(ddrsm.i,addr,3)          /* Got entry address */
        end
      end
      if substr(ddrsm.i,2,3) = 'END' then
        ddrsm.i = overlay(addr,ddrsm.i,6)      /* Place entry address */
        push ddrsm.i
        'EXECIO 1 DISKW IPL DDRSM A'
      end
    end
  'FINIS IPL DDRSM A'
  'ERASE LOADDDR TEXT A'
  'ERASE HCPDDR TEXT A'
end
otherwise exit
end

```

WRITLINE ASSEMBLE

```

WRITLINE CSECT
      TITLE 'WRITLINE - MAIN PROGRAM LOGIC'
*
* WRITLINE is designed to work together with XDDR MODULE, ie call from
* XDDR. XDDR is a modified version of the DDR MODULE.
* WRITLINE must be NUCXLOAded to work properly.
* WRITLINE stores internally the tape labels that are to be mounted
* during a restore of a previous DDR dump. To accomplish this,
* provisions have been made to make WRITLINE callable from CMS (EXEC).
* The call sequence must then be:
*
*      WRITLINE mount label1 label2 .....
*
* When WRITLINE is to mount tapes, a request to the tape handling
* software to mount the first tape must be issued prior to calling
* XDDR. This tape is not included in the WRITLINE call.
*
* If a mount request is not supplied to WRITLINE, it is supposed that

```

```

* the tapes are to be mounted manually.
*
* WRITLINE intercepts messages issued by XDDR and retypes these
* on the terminal. It looks at the message and, according to the
* content, then WRITLINE:
*
* 1. Returns to XDDR after typing.
* 2. If the message is 'RESTORING XXXXXX', sets an internal flag and
* returns to XDDR.
* 3. Prepares to mount next tape through DMSTVS if the message is:
* 'END OF VOLUME CYL xxxxxxxx HD xx, MOUNT NEXT TAPE'
* and we are making a DDR dump or performing a restore with
* automatic tape mounting.
*
*****
*
      STM    R14,R12,12(R13)
      BALR   R12,0
      USING  MAINLINE,R12
MAINLINE DS    0H
      LR     R14,R13
      LA     R13,SAVE1
      ST     R14,4(R13)
      ST     R13,8(R14)
      LR     R2,R1                      Save pointer to parms from caller
      CLC    8(5,R1),MOUNT              Is second parameter MOUNT
      BE     ASKMOUNT                   Yes, call is initial from CMS (EXEC)
      LA     R1,LINEWRT+8               Get address of my LINEWRT plist
      MVC    8(8,R1),8(R2)              Put XDDR's parms into my LINEWRT plis
      SR     R15,R15                    Clear R15
      SVC    204                        Tell CMS to call LINEWRT
*
* Message was displayed. Now, is there a chance that we need another
* tape?
* Check the message text to see.
*
ERROR4  ST     R15,SAVE2                Save in case of errors (for DDR)
      L      R8,8(R2)                  Get address of caller's message text
      CLC    34(15,R8),MNTREQ           Do we need to get next tape?
      BE     GETNEXT                    We need to get next tape
      CLC    0(9,R8),RESTORNG           Are we restoring from tape
      BE     SETRESTR                   Yes, SET we are restoring
      B      EXIT                       Return to DDR
SETRESTR LH     R5,=H'1'                Set RESTORSW to 1
      STH    R5,RESTORSW
      B      EXIT                       Return to DDR
ASKMOUNT NUEXT QUERY,NAME='WRITLINE' See if we are NUCXLOADED, if not,
*                                     then it is impossible to execute
      LTR    R15,R15                    Are we NUCXLOADED

```


	BP	LOADERR	No, set error and return
--	----	---------	--------------------------

*
* Prepare to load specified tape labels. Store labels in area
* CASSETTES. Label area is prefixed with 'WRITLINE' and 'MOUNT' from
* call.
*

	LR	R1,R2	R1 = Parm address
	L	R6,=X'FFFFFFFF'	R6 = Fence
	LA	R7,CASSETTS	R7 = Repository
	LA	R5,12	Maximum number of tapes + 4
NEXT	LM	R2,R3,Ø(R1)	Load value
	CLR	R2,R6	Is it end X'FFFFFFFF'
	BE	STOP	Save it and return
	BCT	R5,SKIP1	Processed max tapes
	B	PARMERR	Yes, return with error
SKIP1	STM	R2,R3,Ø(R7)	No, store label
	LA	R1,8(R1)	Set to next input position
	LA	R7,8(R7)	Set to next output position
	B	NEXT	Go for next label
STOP	BCT	R5,SKIP2	Processed more than max tapes
	B	PARMERR	Yes, return with error
SKIP2	STM	R2,R3,Ø(R7)	No, set end of request
	B	EXIT	Return to CMS (EXEC)

*
* We need to get the next tape via DMSTVS
*

GETNEXT	LH	R5,RESTORSW	Are we restoring
	LTR	R5,R5	
	BZ	TVS	Dumping, call DMSTVS to mount tape
	L	R6,CASSETTS	Restoring, Test further
	LTR	R6,R6	Automatic or manual mount
	BZ	EXIT	Manual tape mount requested, Return

*
* Requests have to go through VMTAPE (via DMSTVS call)
*

	L	R6,=X'FFFFFFFF'	Fence
	LA	R7,FIRST	Start of table
	AH	R7,VOLUME	
	L	R5,Ø(R7)	Get next volume
	CLR	R5,R6	End of volumes already reached
	BE	ERROR	
	LA	R6,NORING	Set to NORING (Write protected)
	LH	R5,VOLUME	Get volume pointer
	LA	R5,8(R5)	Increment by 8
	STH	R5,VOLUME	Save for next call
	LA	R1,DMSTVS	Prepare to call DMSTVS (for restoring from tape)

*
MVC 8(8,R1),Ø(R7) Set type of volume (SCRATCH or label)
MVC 24(8,R1),Ø(R6) set to RING/NORING

```

SVC  202                      Ask CMS to call DMSTVS for me
DC   AL4(BADMOUNT)           Go here on error
APPLMSG TEXT='Next tape mounted successfully'
B    EXIT                     We are now finished. Return to DDR
TITLE 'BADMOUNT - Error Detected During DMSTVS Processing'
BADMOUNT DS  0H
LR   R9,R15
APPLMSG TEXT='Unexpected return code &&1 from
DMSTVS',SUB=(DECXX
          ,0(9),4))
B    EXIT                     We are now finished
* Continues here while dumping disk to tape
TVS   LA   R6,RING             Set to RING (Not write protected)
      LA   R7,SCRATCH          Set to scratch volume
      LA   R1,DMSTVS           Prepare to call DMSTVS (for dumping
*                               to tape)
MVC   8(8,R1),0(R7)           Set type of volume (SCRATCH)
MVC   24(8,R1),0(R6)          Set to RING/NORING
SVC   202                      Ask CMS to call DMSTVS
DC   AL4(BADMOUNT)           Go here on error
APPLMSG TEXT='Next tape mounted successfully'
TAPECTL FSF,TAP1             Forward space past volume label
B    EXIT                     We are now finished. Return to DDR
*
*
TITLE 'PARMERR - Too many mounts specified'
PARMERR DS  0H
APPLMSG TEXT='WRITLINE can only accept 8 tapes for mounting.'
L     R15,=F'15'
B     EXIT
*
TITLE 'LOADERR - WRITLINE not NUCXLOADED'
LOADERR DS  0H
APPLMSG TEXT='WRITLINE can only be called as a nucleus extensiX
on. --- NUCXLOAD WRITLINE ---'
L     R15,=F'13'
B     EXIT
TITLE 'ERROR - Last defined tape already mounted'
ERROR DS  0H
APPLMSG TEXT='WRITLINE has already mounted the last predefinedX
tape.'
L     R15,=F'11'
B     EXIT
*
TITLE 'TYPELINE - Data Areas'
MNTREQ DC  CL15'MOUNT NEXT TAPE' Message from DDR on which we key
*
LINEWRT LINEWRT DATA=((R2),(R3)),ERROR=ERROR4 Write out message
*
```

RESTORNG	DC	CL9'RESTORING'	Mask for RESTORING xxxxxx
*			
MOUNT	DC	CL5'MOUNT'	
*			
SCRATCH	DC	CL8'SCRATCH'	
NORING	DC	CL8'NORING'	
RING	DC	CL8'RING'	
RESTORSW	DC	H'Ø'	Will be set = '1' when a restore
VOLUME	DC	H'Ø'	
	DS	ØD	
CASSETTS	DS	CL8' '	Will be set to 'WRITLINE' from call
	DS	CL8' '	Will be set to 'MOUNT' from call
FIRST	DS	CL8' '	Contains label of first tape
	DS	CL56' '	Space for labels for 7 additional
*			tapes. First one used will
*			contain 'FFFFFFFF FFFFFFFF' the rest
*			is empty.
	DS	ØD	
DMSTVS	DC	CL8'DMSTVS'	Command to mount next tape
	DC	CL8'SCRATCH'	Parms for DMSTVS
	DC	XL2'Ø181',CL6' '	
	DC	CL8'RING'	
	DC	CL8'SL'	
	DC	F'-1'	Fence
	DS	ØD	
NUCEXT	DC	CL8'NUCEXT'	
	DC	CL8'WRITLINE'	
	DC	XL4'AAAAAAA'	
	DC	XL4'FFFFFFFF'	
DETACH	DC	CL12'DETACH Ø181'	
DETLENGT	EQU	*-DETACH	
SAVE1	DS	18F	
SAVE2	DS	2F	
EXIT	L	R13,4(R13)	
	LM	R14,R12,12(R13)	
	L	R15,SAVE2	
	BR	R14	
	REGEQU		
	LTORG		
	END		

Odd Hatlevold
Senior Systems Programmer
Statoil (Norway)

© Xephon 1997

Capturing lines written to the user's console

The attached VM- and MVS-compatible EXEC is for capturing lines written to the user's console. A few years back I was responsible for supporting RACF on VM and MVS. RACF responses to user queries can be very long. RACF commands are quickest when entered from the command line, but long messages can't be studied because they roll off the screen and are gone. There are a number of ways to solve this problem, for example by using the ISPF RACF panels. I am one of those people who want to have their cake and eat it too. CONSAVER makes it easy for me to exploit commands like those in RACF, CMS, or TSO. If I couple it up with the 'ALL' command, I get features that surpass standard ISPF capabilities – especially when you consider the ability to direct the output into files and printers. What's more, having the same command available in VM and MVS saves me a lot of thought time when moving between the two systems. Fortunately, there is no RACF for the workstation, and the piping commands are very simple, or I would have made a CONSAVER for it too. An example follows:

```
TSO CS LISTDSD PREFIX(OPERDEPT) ALL *PRINTIT *VIEWIT
```

Note, CS may be set up as an abbreviation for CONSAVER.

CONSAVER

FUNCTION: Save console messages in a file or display them using the editor.

DESCRIBE: The command is designed to run under CMS, TSO, and MVS.

When a command is executed that displays information on the user's console or sysout queue/spool, CONSAVER will intercept those messages and direct them to:

For VM to a FILE or VM reader which can be PEEKed or edited.

For OS to a dataset which can be optionally ISPF edited.

HOWTORUN: Enter command as shown below:

```
CONSAVER|CS cms-tso-cmd < *keywords >
```

keywords...

*DEBUGIT - Turn on the trace facility for CONSAVER logic.

*ConcatIT - TSO only. Add the trapped console commands on to the or *CATIT prior set of trapped consoles messages.

*APPendIT - Tack the current messages after prior file of

messages.

- *TypeIT - Display all messages before redirection to wherever.
- *ViewIT - Edit the current messages file. This is the default.
- ¬ViewIT - Do not Edit the current messages file.
- *PEEKIT - In CMS use PEEK to view messages while in VM reader.
- *PrintIT - This keyword will print a copy of the command's output.
- *FileIT - In CMS store messages into a default file name of:
userid() CONSAVER A
- or- In TSO store messages into a default file name of:
'userid().CONSAVER'
- In TSO if the CONSAVER file is in use, code will write to
'userid().CONSAVE0' -> 'userid().CONSAVE1' etc...
and keep incrementing the last byte until a free
dataset name is found.
- *FileIT(x) - If an explicit file or dataset name is wanted use
the keyword parameter version of *FILEIT().
Using the file/dataset naming conventions of either
CMS or TSO enter name within parenthesis.
*FILEIT(CON SAVE) or *FILEIT(WORKFILE(CONTEMP))
In TSO it is recommended that the messages file
be explicitly allocated prior to running CONSAVER.

OPERANDS: When running in TSO, the keyword options *VIEWIT and *FILEIT
are the default settings.

EXAMPLES: To put CMS screen output of the command named HANDYIVP into
a file named HANDYIVP TESTRUN A enter...
CONSAVER HANDYIVP *FILEIT(HANDYIVP TESTRUN)

CONSAVER EXEC follows for CMS and TSO...

```

/* AN MVI EXEC */
/*  UNABLE TO GET THIS COMMAND TO WORK WHILE RUNNING ISPF EDIT MACROS.
    NO MATTER WHAT I DID THE TRAP COMMAND FAILED TO TRAP ANYTHING.
    "ISREDIT MACRO (V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12) PROCESS"
    IF RC = 0 THEN ADDRESS ISREDIT
    END
*/
/* THIS COMMAND MUST RUN IN ISPEXEC, ISREDIT, AND TSOBATCH MODE. */
/* GIVING UP, BELIEVE THAT OUTTRAP AND ISREDIT ARE INCOMPATIBLE. */
SYS=ADDRESS(); ISR = 0
IF SYS = 'TSO' | SYS = 'MVS' | SYS = 'ISREDIT' THEN TSO = 1; ELSE
  TSO = 0
IF SYS = 'DOS' | SYS = 'KEDIT' | SYS = 'CMD' THEN DOS = 1; ELSE D
  OS = 0
IF SYS = 'CMS' | SYS = 'XEDIT' | SYS = 'REXX' THEN CMS = 1; ELSE
  CMS = 0
IF CMS THEN SIGNAL CMSLOGIC
IF SYS = 'ISREDIT'
  THEN DO
    LOWSTRING = V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12

```

```

      'ISREDIT(TMEM) = MEMBER'  /* IF SEQ DSN ITS A NULL */
      'ISREDIT(TDSN) = DATASET' /* COMES W/O QUOTES      */
      X = TDSN TMEM           /* SO THE TRACE WILL SHOW THEIR VALUES */
      ISR = 1
      END
    ELSE DO
      PARSE ARG LOWSTRING
      ADDRESS TSO
      ISR = 0
      END
    SIGNAL MVSLOGIC
  CMSLOGIC:
    PARSE ARG LOWSTRING
  MVSLOGIC:
    IF ¬TSO & ¬DOS THEN CMS=1
    ARGSTRING = TRANSLATE(LOWSTRING)
    DEBUG='';X=FIND(TRANSLATE(ARGSTRING),'*DEBUGIT')
    IF X ¬= 0 THEN DO
      TRACE I
      LOWSTRING = DELWORD(LOWSTRING,X,1)
      ARGSTRING = TRANSLATE(LOWSTRING)
      DEBUG = '*DEBUG'
      END
    IF WORD(ARGSTRING,1) = '?' THEN SIGNAL DOC
    TYPEIT = 0; X = FIND(TRANSLATE(ARGSTRING),'*TYPEIT')
    IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*TYPIT')
    IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*TIT')
    IF X ¬= 0 THEN DO
      LOWSTRING = DELWORD(LOWSTRING,X,1)
      ARGSTRING = TRANSLATE(LOWSTRING)
      TYPEIT = 1
      END
    CONCATIT = 0; X = FIND(TRANSLATE(ARGSTRING),'*CONCATIT')
    IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*CATIT')
    IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*CIT')
    IF X ¬= 0 THEN DO
      LOWSTRING = DELWORD(LOWSTRING,X,1)
      ARGSTRING = TRANSLATE(LOWSTRING)
      CONCATIT = 1
      END
    APPENDIT = 0; X = FIND(TRANSLATE(ARGSTRING),'*APPENDIT')
    IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*APPIT')
    IF X ¬= 0 THEN DO
      LOWSTRING = DELWORD(LOWSTRING,X,1)
      ARGSTRING = TRANSLATE(LOWSTRING)
      APPENDIT = 1
      END
    VIEWIT = 1      /* MAKING VIEWIT THE DEFAULT */

```

```

FILEIT = 0; X = FIND(ARGSTRING,'*FILEIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*DSNIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*FIT')
IF X  $\neq$  0 THEN DO
    LOWSTRING = DELWORD(LOWSTRING,X,1)
    ARGSTRING = TRANSLATE(LOWSTRING)
    FILEIT = 1
    VIEWIT = 0    /* MUST BE MANUALLY ENTERED */
END
PRINTIT = 0; X = FIND(TRANSLATE(ARGSTRING),'*PRINTIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*PRTIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*PIT')
IF X  $\neq$  0 THEN DO
    LOWSTRING = DELWORD(LOWSTRING,X,1)
    ARGSTRING = TRANSLATE(LOWSTRING)
    PRINTIT = 1
    FILEIT = 1
    VIEWIT = 0    /* EXPLICITE ENTRY REQUIRED */
END
X = FIND(ARGSTRING,'*VIEWIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),'*VIT')
IF X  $\neq$  0 THEN DO
    LOWSTRING = DELWORD(LOWSTRING,X,1)
    ARGSTRING = TRANSLATE(LOWSTRING)
    VIEWIT = 1
END
X = FIND(ARGSTRING,' $\neg$ *VIEWIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),' $\neg$ *VIEWIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),' $\neg$ *VIT')
IF X = 0 THEN X = FIND(TRANSLATE(ARGSTRING),' $\neg$ *VIT')
IF X  $\neq$  0 THEN DO
    LOWSTRING = DELWORD(LOWSTRING,X,1)
    ARGSTRING = TRANSLATE(LOWSTRING)
    VIEWIT = 0
END
PEEKIT = 0; X = FIND(ARGSTRING,'*PEEKIT')
IF X  $\neq$  0 THEN DO
    LOWSTRING = DELWORD(LOWSTRING,X,1)
    ARGSTRING = TRANSLATE(LOWSTRING)
    PEEKIT = 1
END
X = POS('*FILEIT(',ARGSTRING); V = 8; U = X + V; W = U
IF X = 0 THEN DO
    X = POS('*FIT(',ARGSTRING); V = 5; U = X + V; W = U
END
IF X  $\neq$  0 THEN DO FOREVER    /* PARMREXX */
    Y = POS(')',ARGSTRING,U); IF Y = 0 THEN LEAVE
    Z = POS('(',ARGSTRING,W) /* CHK FOR *VAL1(*SUB1(X) *SUB2(VAL)) */

```

```

    IF Z  $\neq$  0 & Z < Y & LENGTH(ARGSTRING) > Y
        THEN DO; W = Z+1; U = Y+1; ITERATE; END
    ZS = X + V; ZL = Y - X - V
    FILEIT = STRIP(SUBSTR(ARGSTRING,ZS,ZL))
    ZL = Y - X + 1
    LOWSTRING = DELSTR(LOWSTRING,X,ZL)
    ARGSTRING = TRANSLATE(LOWSTRING)
    LEAVE
    END
BEGIN:
SFXLST = '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ' /* SUFFIXES FOR CONSAVE
X */
ARGSTRING = TRANSLATE(LOWSTRING)
IF LOWSTRING = '' THEN SIGNAL ERR190 /* NO CMD FND TO EXECUTE. */
IF CMS THEN SIGNAL CMSSAVER
IF TSO THEN SIGNAL TSOSAVER
SIGNAL ERR016 /* SYSTEM ERROR ENCOUNTERED. */
CMSSAVER:
START = ''; STOP = ''
'EXECIO 1 CP (STEM CONS. STR Q CONS'
IF CONS.0 = 1 THEN IF WORD(CONS.1,6) = 'STOP' THEN DO
    START = 'START'; STOP = 'STOP'
    END
'CPQ SPOOL CONSOLE CLOSE' START
'TERMQ' LOWSTRING
IF RC  $\neq$  0 THEN SAY 'CONSAVER - COMMAND ENDED WITH RC = ('RC').'
'CPQ SPOOL CONSOLE CLOSE' STOP
DO 27
    'SLEEP 3 SEC'
    'CMSQ GETSPLID * * *CLS(T) *TYP(CON)' DEBUG
    IF RC = 1 THEN ITERATE
    IF RC  $\neq$  0 THEN SIGNAL ERR120
    PULL . SPLID .
    IF PEEKIT
        THEN DO
            'PEEK' SPLID '( FOR *'
            'CPQ PURGE RDR' SPLID
            LEAVE
        END
    'CMSQ RECEIVE' SPLID '$$TEMP CONSAVER (REPL'
    IF RC  $\neq$  0 THEN SIGNAL ERR130
    IF WORDS(FILEIT) > 3 THEN SIGNAL ERR050
    IF FILEIT = 0 | FILEIT = 1 | FILEIT = ''
        THEN CONSFIL = USERID() 'CONSAVER A'
    ELSE DO
        IF WORDS(FILEIT) = 1 THEN FILEIT = FILEIT 'CONSAVER A'
        IF WORDS(FILEIT) = 2 THEN FILEIT = FILEIT 'A'
        CONSFIL = FILEIT

```



```

        END
        'CMSQ STATE' CONSOLE
        IF RC = 28 | RC = 0 THEN NOP; ELSE SIGNAL ERR060
        IF ¬APPENDIT
            THEN 'COPYFILE $$TEMP CONSAVER A' CONSOLE '( OLDD REPL'
            ELSE 'COPYFILE $$TEMP CONSAVER A' CONSOLE '( APPEND'
        IF RC ¬= 0 THEN SIGNAL ERR150
        IF TYPEIT THEN 'TYPE $$TEMP CONSAVER A'
        'CMSQ ERASE $$TEMP CONSAVER A'
        /* IF TYPEIT IS ON, SHOW CAPTURED RESPONSES BEFORE PROCESSING. */
        IF ¬VIEWIT /* IF *FILEIT OPTION THEN JUST ISSUE RETRIEVE */
            THEN LEAVE
        PUSH 'VER 1 80'
        PUSH 'PREFIX OFF'
        PUSH 'CMS ERASE' CONSOLE /* MAKE USER FILE TO KEEP FILE */

        'XEDIT' CONSOLE
        IF RC ¬= 0 THEN SIGNAL ERR140
        LEAVE
        END
SIGNAL EXIT
TSOSAVR:
IF ISR THEN ADDRESS TSO
NOCONCAT = 'NOCONCAT'
IF CONCATIT THEN NOCONCAT = 'CONCAT'
X = OUTTRAP(VAR., '*', NOCONCAT); X = X
IF ISR THEN ADDRESS ISREDIT LOWSTRING
ELSE ADDRESS TSO LOWSTRING
/* VAR.0 = COUNT OF LINES. */
X = OUTTRAP('OFF'); X = X
IF ISR THEN ADDRESS ISREDIT
IF VAR.0 = 0 THEN DO
    VAR.0 = 1; VAR.1 = 'CONSAVER - NO MSGES BY CMD ('LOWSTRING'), RC=
    'RC'.'
END
X = VAR.0
XRC = RC /* SAVE THE RETURN CODE FOR EXIT RC */
IF XRC = -3 THEN SIGNAL ERR180 /* COMMAND NOT VALID. */
IF ISR THEN ADDRESS TSO
/* IF TYPEIT IS ON SHOW CAPTURED RESPONSES BEFORE PROCESSING. */
IF TYPEIT THEN DO Z = 1 FOR X
    SAY VAR.Z
END
IF FILEIT = 0 | FILEIT = 1 | FILEIT = ''
    THEN CONSOLE = ""USERID().CONSAVER""
ELSE DO
    CONSOLE = FILEIT
END

```

```

/* ***** */
/* NEED LOGIC THAT WILL TEST FOR A VALID DATASET NAME. */
/* ***** */
CONSFIL = STRIP(CONSFIL)
SFX = 0
DO SFX = 1 FOR LENGTH(SFXLST)
  /* ONLY CHECK FOR DSN, NOT THE MEMBER NAME... */
  X = SYSDSN(SETDSN(CONSFIL))
  IF X = 'OK' THEN DO
    IF POS('(',CONSFIL) = 0
      THEN "DSNALLOC" CONSFIL "*OPSPACE(CYLS 3 3)"
      ELSE "DSNALLOC" CONSFIL "*OPSPACE(CYLS 3 3)",
        "*OPDIR(6)"
    IF RC = 0 THEN SIGNAL ERR160
    LEAVE SFX
  END
  /* THE FOLLOWING COMMAND IS SYSTEM DEPENDENT */
  X = TRANSLATE(XOUTTRAP('LOC' CONSFIL DEBUG))
  IF POS('NOT ALLOC',X) = 0 THEN LEAVE SFX
  IF LEFT(CONSFIL,1) = '('
    THEN Z = LENGTH(CONSFIL) - 1
    ELSE Z = LENGTH(CONSFIL) /* ADDRESS OF LAST CHAR */
  X = SUBSTR(SFXLST,SFX,1) /* SET NEW SFX CHAR. */
  CONSFIL = OVERLAY(X,CONSFIL,Z)
END
IF SFX > LENGTH(SFXLST) THEN SIGNAL ERRXXX /* NO MORE SUFFIXES LEFT */
"ALLOCATE DATASET("CONSFIL") FILE(CONDD) SHR REUSE"
IF RC = 0 THEN SIGNAL ERR010
IF APPENDIT THEN DO
  APP.0 = 0 /* APPEND BUFFER IS EMPTY */
  X = SYSDSN(CONSFIL) /* DSN PRESENCE ALREADY VERIFIED */
  IF X = 'OK' /* OK, THEN WE HAVE A MISSING MEMBER */
    THEN DO
      'EXECIO * DISKR CONDD ( STEM APP. FINIS )'
      IF RC = 20 & POS('(',CONSFIL) = 0
        THEN DO
          /* MEMBER DOES NOT EXIST */
          APP.0 = 0 /* SET ZERO RECS READ */
        END
      IF RC = 0 THEN SIGNAL ERR030
    END
  DO NDXV = 1 FOR VAR.0
    NDXA = APP.0 + NDXV
    APP.NDXA = VAR.NDXV
  END
  APP.0 = APP.0 + VAR.0
END
IF NOT APPENDIT

```

```

        THEN 'EXECIO * DISKW CONDD ( STEM VAR. FINIS )'
        ELSE 'EXECIO * DISKW CONDD ( STEM APP. FINIS )'
IF RC > 1 THEN SIGNAL ERR020
IF PRINTIT & FILEIT  $\neq$  0 THEN 'FPRT' CONSOLE
IF VIEWIT | PEEKIT
    THEN DO
        "FREE FILE(CONDD)"
        QUEUE "STATS OFF"
    /* QUEUE "XTSO DSNERASE ""USERID()".CONSAVER"" /* CANNOT WORK */ */
        "ISPEXEC EDIT DATASET("CONSOLE")",
            "MACRO(RDSTACK) MIXED(YES) "
        IF RC > 4 THEN SIGNAL ERR040
        SIGNAL EXIT
    END
FREE:
"FREE FILE(CONDD)"
IF ISR THEN ADDRESS ISREDIT
SIGNAL EXIT
/*****
EXIT:
IF RC = 0 & DATATYPE(XRC) = 'NUM' THEN RC = XRC
IF DATATYPE(XRC) = 'NUM'
    THEN IF XRC  $\neq$  0
        THEN SAY 'CONSAVER - CMD('LOWSTRING')'S RC WAS" XRC'.'
EXIT RC
*****/
/* SETDSN - CODE PROTOTYPE TO REMOVE MEMBER NAME FROM DATASET NAME. */
/*****
SETDSN: PROCEDURE
PARSE ARG FN
DSN = FN
IF POS('(',FN)  $\neq$  0          /* IS THERE A MEMBER NAME ATTACHED TO DSN */
    THEN DO                /* YES, STRIP IT OFF */
        IF LEFT(FN,1) = "'" /* IS THE DATASET NAME IN QUOTES */
            THEN DO        /* YES, DSN IS IN QUOTES */
                PARSE VAR FN "" DSN "(" .
                DSN = ""DSN""
            END
        ELSE DO            /* NO, THE DATASET NAME IS NOT IN QUOTES */
            PARSE VAR FN DSN "(" .
        END
    END
RETURN DSN                /* RETURN A MEMBER LESS DATASET NAME */
DOC:
/*BEGTYPE
REXXNAME: CONSAVER

```

FUNCTION: Save console messages in a file or display them using the

editor.

DESCRIBE: The command is designed to run under CMS, TSO, and MVS.

When a command is executed that displays information on the user's console or sysout queue/spool, CONSAVER will intercept those messages and direct them to:

- For VM to a FILE or VM reader which can be PEEKed or edited.
- For OS to a dataset which can be optionally ISPF edited.

HOWTORUN: Enter command as shown below:

```
CONSAVER|CS  cms-tso-cmd  < *keywords >
```

keywords...

- *DEBUGIT - Turn on the trace facility for CONSAVER logic.
- *ConcatIT - TSO only. Add the trapped console commands on to the or *CATIT prior set of trapped consoles messages.
- *APPendIT - Tack the current messages after prior file of messages.
- *TypeIT - Display all messages before redirection to wherever.
- *ViewIT - Edit the current messages file. This is the default.
- ¬ViewIT - Do not Edit the current messages file.
- *PEEKIT - In CMS use PEEK to view messages while in VM reader.
- *PrintIT - This keyword will print a copy of the command's output.
- *FileIT - In CMS store messages into a default file name of:
 - userid() CONSAVER A
 - or- In TSO store messages into a default file name of:
 - 'userid().CONSAVER'
 - In TSO if the CONSAVER file is in use, code will write to 'userid().CONSAVE0' -> 'userid().CONSAVE1' etc... and keep incrementing the last byte until a free dataset name is found.
- *FileIT(x) - If an explicit file or dataset name is wanted use the keyword parameter version of *FILEIT(). Using the file/dataset naming conventions of either CMS or TSO enter name within parenthesis.
 - *FILEIT(CON SAVE) or *FILEIT(WORKFILE(CONTEMP))
 - In TSO it is recommended that the messages file be explicitly allocated prior to running CONSAVER.

OPERANDS: When running in TSO, the keyword options *VIEWIT and *FILEIT are the default settings.

EXAMPLES: To put CMS screen output of the command named HANDYIVP into a file named HANDYIVP TESTRUN A enter...

```
CONSAVER HANDYIVP *FILEIT(HANDYIVP TESTRUN)
```

```
ENDTYPE*/
TEXT='' /* BUT DOCUMENTATION DOES NOT HAVE TO BE AT THE BEGINNING. */
SOURCECNT = SOURCELINE() /* REDUCES NUMBER OF CALLS TO SOURCELINE*/
DO SEQ = 1,
  UNTIL POS('-*/',TEXT)>0 | POS('ENDTYPE*/',TEXT)>0 | SEQ>SOURCECNT
  SOURCEREC = SOURCELINE(SEQ) /* REDUCES NUMBER OF CALLS TO SOURCELINE*/
```

```

NE*/
  IF TEXT == ''
    THEN IF POS('/*- ',WORD(SOURCEREC,1)) = 0 &,
      POS('/*BEGTYPE',WORD(SOURCEREC,1)) = 0
      THEN ITERATE
  TEXT = SOURCEREC
  IF POS('/*BEGTYPE',WORD(TEXT,1))>0 | POS('ENDTYPE*/',WORD(TEXT,1))>0
    THEN ITERATE
  SAY TEXT
END
EXIT 000
ERR010:
SAY "CONSAVER - DATASET '"USERID()".CONSAVER' IS NOT ALLOCATED."
EXIT 010
ERR016:
SAY "CONSAVER - SYSTEM ERROR OCCURRED. NOT A CMS OR TSO REQUEST. "
EXIT 016
ERR020:
SAY "CONSAVER - EXECIO DISKW TO: '"USERID()".CONSAVER' FAILED, RC="RC".
"
RC = 020
SIGNAL FREE
ERR030:
SAY "CONSAVER - EXECIO DISKR OF: '"USERID()".CONSAVER' FAILED, RC="RC".
"
RC = 030
SIGNAL FREE
ERR040:
SAY "CONSAVER - EXECIO DISKR OF: '"USERID()".CONSAVER' FAILED, RC="RC".
"
EXIT 040
ERR050:
SAY "CONSAVER - *FILEIT("FILEIT") HAS TOO MANY OPERANDS IN FILE NAME."
EXIT 050
ERR060:
SAY "CONSAVER - *FILEIT("FILEIT") HAS AN INVALID CONSAVER FILE NAME."
EXIT 060
ERR110:
SAY 'CONSAVER - THE COMMAND FAILED WITH A RC =' RC'.'
EXIT 110
ERR120:
SAY 'CONSAVER - THE GETSPLID FAILED WITH A RC =' RC'.'
EXIT 120
ERR130:
SAY 'CONSAVER - THE RECEIVE FAILED WITH A RC =' RC'.'
EXIT 130
ERR140:
SAY 'CONSAVER - VIEW WAS NOT COMPLETED SUCCESSFULLY.'

```

```

EXIT 140
ERR150:
SAY "CONSAVER - COPYFILE TO: '"USERID()" CONSAVER' FAILED, RC="RC"."
EXIT 150
ERR160:
SAY "CONSAVER - ALLOCATE OF ("CONSFIL") FAILED WITH RC="RC"."
EXIT 160
ERR170:
SAY "CONSAVER - NO MORE SUFFIXES LEFT FOR CONSAVER FILE NAME."
EXIT 170
ERR180:
SAY "CONSAVER - INPUT COMMAND WAS NOT FOUND, ERROR TEXT FOLLOWS."
SAY "  XX----->" LOWSTRING
EXIT 180
ERR190:
SAY "CONSAVER - NO INPUT COMMAND WAS ENTERED FOR CONSAVER TO EXECUTE."
SAY "          TO SEE DOCUMENTATION ENTER...  CONSAVER ?          "
EXIT 190

```

GETSPLID EXEC

EXEC for accessing console files in the virtual reader.

```

/* */
ARG ARGSTRING; DEBUG = ''; $X = (FIND(ARGSTRING,'*DEBUG'))
IF $X ≠ 0 THEN DO; ARGSTRING = (DELWORD(ARGSTRING,$X,1)); TRACE I
  DEBUG = '*DEBUG'; END
IF (FIND(ARGSTRING,'?')) = 1 THEN SIGNAL DOC
TMR = 90; X = POS('*TMR(',ARGSTRING)
IF X ≠ 0 THEN DO          /* PARMREXX */
  Y = POS(')',ARGSTRING,X)
  ZS = X + 05; ZL = Y - X - 05
  Z = SUBSTR(ARGSTRING,ZS,ZL)
  IF DATATYPE(Z) = 'NUM' THEN TMR = Z
  ZL = Y - X + 1
  ARGSTRING = DELSTR(ARGSTRING,X,ZL)
END
CLS = ''; X = POS('*CLS(',ARGSTRING)
IF X ≠ 0 THEN DO          /* PARMREXX */
  Y = POS(')',ARGSTRING,X)
  ZS = X + 05; ZL = Y - X - 05
  CLS = SUBSTR(ARGSTRING,ZS,ZL)
  ZL = Y - X + 1
  ARGSTRING = DELSTR(ARGSTRING,X,ZL)
END
TYP = ''; X = POS('*TYP(',ARGSTRING)
IF X ≠ 0 THEN DO          /* PARMREXX */

```

```

    Y = POS(')',ARGSTRING,X)
    ZS = X + 05; ZL = Y - X - 05
    TYP = SUBSTR(ARGSTRING,ZS,ZL)
    ZL = Y - X + 1
    ARGSTRING = DELSTR(ARGSTRING,X,ZL)
END
BEGIN:
RTNCD = 1
PARSE VAR ARGSTRING FN FT .
IF FN = '' THEN IF TYP = '' & CLS = '' THEN SIGNAL DOC; ELSE FN =
    '*'
IF FT = '' THEN FT = 'OUTPUT'
IF FN = '*' & FT = '*'
    THEN IF TYP = '' & CLS = '' THEN SIGNAL ERR020
'QUERY RDR * ALL (STACK LIFO'
IF RC = 88 THEN SIGNAL ERR010
IF RC ≠ 0 THEN SIGNAL EXIT
IF QUEUED() = 0 THEN SIGNAL EXIT
'TIMECALC' TIME(L) 'QUIET'
PULL . NOWSECS .
LMT = 0; DO X = QUEUED() BY -1 UNTIL X ≤ 1 | LMT > 3
    LMT = LMT + 1
    PULL SPLREC
    SAY SPLREC
    PARSE VAR SPLREC . SPLID SCLS STYP . . . SDT STM SFN SFT SDIST .
    PARSE VAR STM HH ':' MM ':' SS
    /* MAKE SURE THAT THIS IS NOT THE HEADER LINE */
    IF DATATYPE(HH|MM|SS) ≠ 'NUM' THEN ITERATE
    'TIMECALC' STM 'QUIET'
    PULL . SPLSECS .
    IF (FN = SFN | FN = '*') &,
        (FT = SFT | FT = '*') &,
        ABS(NOWSECS - SPLSECS) ≤ TMR
    THEN DO 1
        IF CLS ≠ '' THEN IF CLS ≠ SCLS THEN LEAVE
        IF TYP ≠ '' THEN IF TYP ≠ STYP THEN LEAVE
        RTNCD = 0
        SIGNAL EXIT
    END
END
EXIT:
DESBUF
IF RTNCD = 0,
    THEN DO
        PUSH '*' SPLID SFN SFT SDIST SUBWORD(SPLREC,9)
        SAY 'GETSPLID - FILE ('SFN SFT') ENTERED RDR WITH SPOOL ID' SPLID'.'
        END
    ELSE SAY 'GETSPLID - FILE ('FN FT') NOT FOUND IN RDR.'

```

```

EXIT RTNCD
DOC:
/*BEGTYPE
REXXNAME: GETSPLID
FUNCTION: Simply get RDR spool ID for matching file name/type from RSCS.
COMMAND: Enter command as shown below. If the desired file has
recently been put in your reader it will respond with a return
code of zero and stack the spool ID (fmt = * 3939) for
your use.
If no match is found then a return code of 1 is set.
GETSPLID FILENAME < FILETYPE > < *CLS() > < *TYP() >
< > - means field within is optional.
filename - enter the job name from the jobcard sent to MVS.
A wildcard star(*) can be used for any filename.
filetype - enter job type for job returned, default is OUTPUT.
A wildcard star(*) can be used for any filetype.
*cls - enter to screen by files class. ex (T) for console.
*typ - enter to screen by files type. ex. (prt) for print.
Note... The format of the stacked spool ID record will be an
asterisk followed by the spool ID (ie * 2343).
Also, there must be nothing in the stack when this
command is run.
EXAMPLES: To see if a batch Job called MVSALLOC has returned from MVS
enter:
GETSPLID MVSALLOC
ENDTYPE*/
'REXSAYIT GETSPLID EXEC * /*BEGTYPE ENDTYPE*/'
EXIT 000
ERR010:
SAY 'GETSPLID - TOO MANY FILES ARE IN YOUR READER. PURGE SOME AND
RERUN.'

```

DSNALLOC EXEC

The following EXEC contains TSO code that allocates datasets.

```

/* AN MVI EXEC */
PARSE UPPER ARG ARGSTRING;DEBUG='';$X=FIND(TRANSLATE(ARGSTRING),'*DEB
UG')
IF $X = 0 THEN DO; ARGSTRING = DELWORD(ARGSTRING,$X,1); TRACE I
DEBUG = '*DEBUG'; END

```



```

IF WORD(ARGSTRING,1) = '?' THEN SIGNAL DOC
OPRECFM = ''; X = POS('*OPRECFM(',TRANSLATE(ARGSTRING)); V = 9
IF X = 0 THEN DO
    X = POS('*RECFM(',TRANSLATE(ARGSTRING)); V = 7
    END
IF X = 0 THEN DO 1      /* PARAMETER SETTING IN THIS ROUTINE. */
    Y = POS(')',ARGSTRING,X); IF Y = 0 THEN LEAVE
    ZS = X + V; ZL = Y - X - V
    Z = STRIP(SPACE(SUBSTR(ARGSTRING,ZS,ZL),0))
    ZL = Y - X + 1
    ARGSTRING = DELSTR(ARGSTRING,X,ZL)
    DO X = 1 FOR LENGTH(Z)
        Y = SUBSTR(Z,X,1)
        IF POS(Y,OPRECFM',;.: ' ) = 0 THEN ITERATE /* SKIP DUPES COMM
AS ETC*/
        IF FIND('A B F M S U V',Y) = 0 THEN SIGNAL ERR080
        IF X = 1 /* FORMAT IS RECFM(F,B,A) */
            THEN OPRECFM = Y
            ELSE OPRECFM = OPRECFM','Y
        END
    END
OPLRECL = ''; X = POS('*OPLRECL(',TRANSLATE(ARGSTRING)); V = 9
IF X = 0 THEN DO
    X = POS('*LRECL(',TRANSLATE(ARGSTRING)); V = 7
    END
IF X = 0 THEN DO
    X = POS('*RSIZ(',TRANSLATE(ARGSTRING)); V = 6
    END
IF X = 0 THEN DO 1      /* PARAMETER SETTING IN THIS ROUTINE. */
    Y = POS(')',ARGSTRING,X); IF Y = 0 THEN LEAVE
    ZS = X + V; ZL = Y - X - V
    OPLRECL = STRIP(SUBSTR(ARGSTRING,ZS,ZL))
    ZL = Y - X + 1
    ARGSTRING = DELSTR(ARGSTRING,X,ZL)
    IF DATATYPE(OPLRECL) = 'NUM' THEN SIGNAL ERR070
    END
OPBLKSZ = ''; X = POS('*OPBLKSIZE(',TRANSLATE(ARGSTRING)); V = 11
IF X = 0 THEN DO
    X = POS('*BLKSIZE(',TRANSLATE(ARGSTRING)); V = 9
    END
IF X = 0 THEN DO
    X = POS('*BLKSZ(',TRANSLATE(ARGSTRING)); V = 7
    END
IF X = 0 THEN DO
    X = POS('*BSIZ(',TRANSLATE(ARGSTRING)); V = 6
    END
IF X = 0 THEN DO 1      /* PARAMETER SETTING IN THIS ROUTINE. */
    Y = POS(')',ARGSTRING,X); IF Y = 0 THEN LEAVE
    ZS = X + V; ZL = Y - X - V

```

```

OPBLKSZ = STRIP(SUBSTR(ARGSTRING,ZS,ZL))
ZL = Y - X + 1
ARGSTRING = DELSTR(ARGSTRING,X,ZL)
IF DATATYPE(OPBLKSZ)  $\neq$  'NUM' THEN SIGNAL ERR060
END
OPDCB = ''
IF OPRECFM  $\neq$  '' THEN OPDCB = OPDCB'RECFM('OPRECFM') '
IF OPLRECL  $\neq$  '' THEN OPDCB = OPDCB'LRECL('OPLRECL') '
IF OPBLKSZ  $\neq$  '' THEN OPDCB = OPDCB'BLKSIZE('OPBLKSZ') '
OPMODEL = ''; X = POS('*OPMODEL(',ARGSTRING); T = 9; U = X + T; W = U
IF X = 0 THEN DO; X = POS('*MODEL(',ARGSTRING);
T = 7; U = X + T;
W = U; END
IF X  $\neq$  0 THEN DO FOREVER /* PARMREXX */
Y = POS(')',ARGSTRING,U); IF Y = 0 THEN LEAVE
Z = POS('(',ARGSTRING,W) /* CHK FOR *VAL1(*SUB1(X) *SUB2(VAL)) */
IF Z  $\neq$  0 & Z < Y & LENGTH(ARGSTRING) > Y
THEN DO; W = Z+1; U = Y+1; ITERATE; END
ZS = X + T; ZL = Y - X - T
OPMODEL = STRIP(SUBSTR(ARGSTRING,ZS,ZL))
ZL = Y - X + 1
ARGSTRING = DELSTR(ARGSTRING,X,ZL)
LEAVE
END
OPDIR = ''; X = POS('*OPDIR(',ARGSTRING); T = 7
IF X = 0 THEN DO; X = POS('*DIR(',ARGSTRING); T = 5; END
IF X  $\neq$  0 THEN DO 1 /* IS THE *OPDIR() OPTION USED? */
Y = POS(')',ARGSTRING,X) /* FIND THE END OF THE INPUT PARM */
ZS = X + T; ZL = Y - X - T /* CALC START & LENGTH OF IP PARM */
OPDIR = SUBSTR(ARGSTRING,ZS,ZL) /* SET THE OPDIR PARM VALS */
IF OPDIR = '' THEN LEAVE
IF DATATYPE(OPDIR,'NUM') = 0 THEN SIGNAL ERR050
IF OPDIR > 0
THEN OPDIR = 'DSORG(PO) DIR('OPDIR')'
ELSE OPDIR = 'DSORG(PS) DIR(0)'
ZL = Y - X + 1 /* CALC LENGTH OF IP PARM TO DROP */
ARGSTRING = DELSTR(ARGSTRING,X,ZL) /* DROP THE INPUT PARM FLD */
END
OPDASD = ''; X = POS('*OPDASD(',ARGSTRING); T = 8 /* FIND DASD DEF
OS*/
IF X = 0 THEN DO; X = POS('*DASD(',ARGSTRING); T = 6; END
IF X  $\neq$  0 THEN DO /* IS THE *OPDASD() OPTION USED?*/
Y = POS(')',ARGSTRING,X) /* FIND THE END OF THE INPUT PARM */
ZS = X + T; ZL = Y - X - T /* CALC START & LENGTH OF IP PARM */
OPDASD = SUBSTR(ARGSTRING,ZS,ZL) /* SET THE CC PARM VALS */
OPDASD = TRANSLATE(OPDASD,' ','_./.;')
PARSE VAR OPDASD DSDUNIT DSDVOLS
IF DSDUNIT  $\neq$  '' THEN OPDASD = 'UNIT('DSDUNIT')'
IF DSDVOLS  $\neq$  '' THEN OPDASD = OPDASD 'VOLUME('DSDVOLS')'

```

```

        ZL = Y - X + 1                /* CALC LENGTH OF IP PARM TO DROP */
        ARGSTRING = DELSTR(ARGSTRING,X,ZL) /* DROP THE INPUT PARM FLD */
        END
OPSPACE = ''; X = POS('*OPSPACE(',ARGSTRING); T = 9 /* LK SPAC DEF
OS*/
IF X = 0 THEN DO; X = POS('*SPACE(',ARGSTRING); T = 7; END
IF X ≠ 0 THEN DO /* IS THE *OPSPAC() OPTION USED?*/
    Y = POS(')',ARGSTRING,X) /* FIND THE END OF THE INPUT PARM */
    ZS = X + T; ZL = Y - X - T /* CALC START & LENGTH OF IP PARM */
    OPSPACE = SUBSTR(ARGSTRING,ZS,ZL) /* SET THE OPSPACE PARM VALS */
    OPSPACE = TRANSLATE(OPSPACE,' ','_','./;')
    IF SUBWORD(OPSPACE,2) ≠ '' &,
        DATATYPE(SPACE(SUBWORD(OPSPACE,2),0)) ≠ 'NUM' THEN SIGNAL ERR03
0
    PARSE VAR OPSPACE SPCTYPE SPC1ST SPC2ND SPCBLKS Z
    IF Z ≠ '' THEN SIGNAL ERR040
    IF LEFT(SPCTYPE,2) = 'TR'
        THEN SPCTYPE = 'TRACKS'
        ELSE IF LEFT(SPCTYPE,2) = 'CY'
            THEN SPCTYPE = 'CYLINDERS'
            ELSE IF LEFT(SPCTYPE,2) = 'BL'
                THEN DO
                    IF SPCBLKS = '' THEN SIGNAL ERR040
                    IF DATATYPE(SPCBLKS) ≠ 'NUM' THEN SIGNAL ERR040
                    SPCTYPE = 'BLOCK('SPCBLKS')'
                END
            ELSE IF DATATYPE(SPCTYPE) = 'NUM'
                THEN SPCTYPE = 'BLOCK('SPCTYPE')'
            ELSE SIGNAL ERR040
    IF (SPC1ST SPC2ND) = ''
        THEN OPSPACE = SPCTYPE 'SPACE(018,009)'
        ELSE OPSPACE = SPCTYPE 'SPACE('SPC1ST','SPC2ND')'
    ZL = Y - X + 1                /* CALC LENGTH OF IP PARM TO DROP */
    ARGSTRING = DELSTR(ARGSTRING,X,ZL) /* DROP THE INPUT PARM FLD */
    END
IF OPSPACE = '' & OPMODEL = ''
    THEN OPSPACE = 'CYLINDERS SPACE(1,3)'
IF OPMODEL = ''
    THEN OPLIKE = "LIKE('TDMVS.MIRVI.FB0132')'"
    ELSE OPLIKE = 'LIKE('OPMODEL')'
OPDEF = 'NEW CATALOG' OPLIKE OPDIR OPDASD OPSPACE
BEGIN:
    PARSE VAR ARGSTRING XDSN ERROPT
    IF ERROPT ≠ '' THEN SIGNAL ERR008 /* INPUT SYNTAX ERROR */
    IF XDSN = '' THEN SIGNAL ERR010
    X = "ALLOCATE DATASET("XDSN")",
        "FILE(DD1)" OPDEF OPDCB
X
    IF RC ≠ 0 THEN SIGNAL ERR020

```

```

"FREE DDNAME(DD1)"
EXIT:
EXIT 000
DOC:
SAY 'EXECNAME: DSNALLOC'
SAY
SAY 'FUNCTION: ALLOCATE DATASETS BASED ON MODEL() SPACE() OPTIONS.'
SAY
SAY 'ENTRYFMT: ENTER COMMAND IN THE FORMAT SHOWN BELOW.          '
SAY
SAY '          DSNALLOC &DSN *MODEL(DSN)'
SAY
SAY '  OTHER KEYWORDS FOLLOW...'
SAY '          *DIR(N) *SPACE(T P S) *RECFM(X) *RSIZ(N) *BSIZ(N) *DASD(U V)'
SAY
SAY
EXIT 000
ERR008:
SAY 'DSNALLOC - UNRECOGNIZED INPUT PARAM WAS:' ERROPT
EXIT 008
ERR010:
SAY 'DSNALLOC - NO INPUT DATASET NAME FOUND. '
EXIT 010
ERR020:
SAY 'DSNALLOC - RC='RC 'ALLOCATING ('XDSN'). '
SAY 'DSNALLOC - ('||X')'
EXIT 020
ERR030:
SAY 'DSNALLOC - *OPSPACE('OPSPACE') HAS NON-NUMERIC AMOUNT.'
EXIT 030
ERR040:
SAY 'DSNALLOC - *OPSPACE('OPSPACE') HAS INVALID FORMAT.'
EXIT 040
ERR050:
SAY 'DSNALLOC - *OPDIR('OPDIR') HAS NON-NUMERIC AMOUNT.'
EXIT 050
ERR060:
SAY 'DSNALLOC - *OPBLKSIZE('OPBLKSZ') HAS NON-NUMERIC AMOUNT.'
EXIT 060
ERR070:
SAY 'DSNALLOC - *OPLRECL('OPLRECL') HAS NON-NUMERIC AMOUNT.'
EXIT 070
ERR080:
SAY 'DSNALLOC - *OPRECFM('OPRECFM') HAS AN INVALID FORMAT INDICATOR.'
EXIT 080

```

Marc Vincent Irvin

Move Immediate Software (USA)

© M V Irvin 1997

Dynamic menus system for CMS – part 2

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

```
/* START BUILDING MY EXEC */
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING /* 'OPTION' EXEC */
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING "SET CMSTYPE HT"
/* WHAT TEMPORARY DISK DO I WANT ? */
IF STRIP(NCYL)≠'N' THEN DO
  IF STRIP(NCYL)='Y' THEN NCYL='10'
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING CALL MAKETDSK 'NCYL
END /* END TEMP DISK STATEMENT */

/* LINK STATEMENTS */
DO I=1 TO LNKID.0
LNKPSWD.I=REVERSE(X2C(STRIP(LNKPSWD.I)))
DET_VDSK.I=""DET 'TRGADDR.I' ""
LNK_VDSK.I=""CP LINK 'LNKID.I ORGADDR.I TRGADDR.I LNKMODE.I LNKPSWD.I'
""
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING 'DET_VDSK.I ' ; ' LNK_VDSK.I
END
/* ACCESS STATEMENTS */
ACC_LINE='ADDRESS CMS '
DO I=1 TO ACCADDR.0
ACC_LINE=ACC_LINE||' ;"ACCESS 'ACCADDR.I ACCMODE.I""'
END
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING 'ACC_LINE
/*'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING "ACCESS 'ACCADDR.I
ACCMODE.I""*/
/* EXEC STATEMENT */
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING QUEUE 'EX_NAME
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING QUEUE
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING QUEUE FIN'
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING EX FOCUS'
/* THIS IS FOR SOME HOUSEKEEPING */
DET_LINE=''
DO I=1 TO TRGADDR.0
DET_LINE=""DET 'TRGADDR.I"" ; '||DET_LINE
END
DET_LINE=DET_LINE||""ACCESS 100 X/A""
'EXECIO 1 DISKW 'OPTION' EXEC A3 (STRING 'DET_LINE
'EXECIO 1 DISKW 'OPTION' EXEC A3 (FINIS STRING DESBUF;"SET CMSTYPE
RT";RETURN'
EXIT 000
```

TAFUME REP

```
*
*      DYNAMIC MENUS SYSTEM MESSAGE REPOSITORY
*
* AFTER EVERY CHANGE IN THIS FILE, YOU MUST :
*
* 'GENMSG TAFUME REPOS A TAF'
* 'SET LANGUAGE (ADD TAF USER'
& 3
* DYNAMIC MENUS MESSAGES
00000101I FUNCTION SUCCESSFULLY COMPLETED
00010101E &1 KEY IS NOT OPERATIONNAL
00020101E CANNOT SCROLL TO THE RIGHT
00030101E CANNOT SCROLL TO THE LEFT
00040101E ENTER AN OPTION IN ANY IF THE MENU BOXES
00050101E OPTION &2 IS INVALID IN &1
00060101E THIS OPTION IS NOT OPERATIONNAL
00070101E &1 FUNCTION &3 FAILED WITH RETURN CODE &2
00080101E UNKNOWN TYPE IN OPTION &1
00090101E MAKEEXEC FAILED TO GENERATE &1 .RC= &2
* LOGO / IDENTIFICATION MESSAGES
00120101E MENU NAME WAS NOT SUPPLIED
00130101E MENU PASSWORD WAS NOT SUPPLIED
00140101E NEW PASSWORD IS SAME AS CURRENT
00150101E WHAT ? SERVER RESPONSE IS INVALID
00160101E REQUESTED MENU (&1) UNKNOWN
00170101E SIGNON PASSWORD IS INVALID
00180101E ADD USER PASSWORD IS INVALID
00190101I PASSWORD SUCCESSFULLY CHANGED
00200101E MENU &1 SUCCESSFULLY ADDED
00210101E MENU &1 IS NOT REGISTERED
```

VMU SRL

A:READ.ME	TEXT~READ	ME	C
A:LOGO.EXC	TEXT~LOGO	EXEC	A
A:MENU.EXC	TEXT~MENU	EXEC	A
A:SERVPASW.EXC	TEXT~SERVPASW	EXEC	A
A:GOOUT.EXC	TEXT~GOOUT	EXEC	A
A:INTM.EXC	TEXT~INTM	EXEC	A
A:MAKEEXEC.EXC	TEXT~MAKEEXEC	EXEC	A
A:MAKETDSK.EXC	TEXT~MAKETDSK	EXEC	A
A:TAFUME.REP	TEXT~TAFUME	REPOS	A
A:XLINES.MEN	TEXT~MENU	XLINES	A
A:XAUTH.MEN	TEXT~MENU	XAUTH	A
A:XSTATMS.MEN	TEXT~MENU	XSTATMS	A
A:XPASW.LOG	TEXT~LOG1	XPASW	A

XPASW LOG

```
=USERID==PASSWORD=DATE=TIME===HISTORY=DATA=====
MAINT  !MAINT  !9621760038!MAINT  !MAINTABI!9621759531!
759518!
MENUADMN!MENUADMN!9624249167!
FOCUSER !FOCUSER !9624249192!
INFOCENT!INFOCENT!9624249430!
TESTER  !TESTER  !9624249446!
TEMPUS  !TEMPUS  !9624249470!
```

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

© Xephon 1997

Performance hints and tips

It seems that many of the leading-edge sites have had VM/ESA for a long time now, and there must have been many opportunities to work out ways to make it perform better for them.

We're looking to publish the hints and tips from a number of sites, so that they can be shared by all. Send us your tips on performance management, improving hardware performance, enhancing software performance, reducing paging and swapping, useful SET commands, using DCSS, improving I/O, monitoring and controlling network performance, guest system and subsystem performance, and tuning and capacity planning.

Articles can be sent to Trevor Eddolls at any of the addresses shown on page 2, or e-mailed to xephon@compuserve.com. Remember that we pay \$250 (£170) per 1000 words and \$140 (£90) per 100 lines of code published (if you give us copyright).

XEDIT extensions

The XEDIT extensions are designed mainly for application 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.

The XEDIT enhancements concern the following tools:

- Function key definitions.
- Cursor movement control.
- Line pointer movement control.
- Copying line above the cursor.
- Copying, moving, and deleting text in marked boxes.
- Useful commands and synonyms.

	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 40	Schange 9
PF7	Up 21	Cursor to previous word
PF8	Down 21	Cursor to next word
PF9	Right 40	Tabf
PF10	Up 5	Cursor 10 columns to left
PF11	Down 5	Cursor 10 columns to right
PF12	Box processing	Centre cursor

Figure 1: PFK settings


```

*** Extended cursor movement control
PF2  moves cursor to column 1 of line;
PF3  moves cursor to column after last word in line;
PF7  moves cursor to column preceding word left of the cursor;
PF8  moves cursor to column preceding word right of the cursor
PF10 moves cursor 10 columns to left of the cursor;
PF11 moves cursor 10 columns to right of the cursor.
*** Extended line pointer movement control
PF4  moves line pointer between top of the file and
      current line pointer position, if pressed sequentially;
PF5  moves line pointer between bottom of the file and
      current line pointer position, if pressed sequentially;
PF7  moves line pointer up 21 lines;
PF8  moves line pointer down 21 lines;
PF10 moves line pointer up 5 lines;
PF11 moves line pointer down 5 lines.
*** Copying line above the cursor
PF4  copies text left of the cursor from above line;
PF5  copies text right of the cursor from above line;
PA2  undo line, after PF4 or PF5 copying.
*** Box copy operations

```

Figure 2: Additional PF key actions

BASIC SOFTWARE FOR XEDIT/E

XEDIT/E is developed in CMS with VM/SP Release 5.

THE XEDIT/E EDITING TOOLS

The PFK settings are shown in Figure 1. Additional PFK settings actions are shown in Figure 2. The PFK settings after box operations are selected by pressing PF12 are shown in Figure 3. The PFK settings to process a box are shown in Figure 4.

The PF1 and PF2 operations require only one mark to specify the column to insert/delete. In this case, a box may not be marked and the current position of the cursor is ignored.

All the following operations, except PF1 and PF2, require a box to be marked to start processing. The PF4, PF5, PF6, and PF7 operations are processed using the relative current cursor position. The PF9 and

```

PF1  - Mark;
PF2  - Unmark;
PF3  - Escape;
PF4  - moves line pointer between top of the file and
      top of the marked box, if pressed sequentially;
PF5  - moves line pointer between bottom of the file and
      bottom of the marked box, if pressed sequentially;
PF6  - moves cursor 10 columns to left of the cursor;
PF7  - Up 21;
PF8  - Down 21;
PF9  - moves cursor 10 columns to right of the cursor.
PF10 - Left 40;
PF11 - Right 40;
PF12 - Process box.

```

Figure 3: PFK settings after box operations

PF10 operations insert/delete a number of columns beginning from the left side of the box.

The only boxes with width greater than 2 are marked. The box processing is shown in Figure 5.

The following commands are defined:

- Q CMS exit, PF3 return to XEDIT environment.
 - (Set PF10 as LE 40, PF11 as RI 40, and PREFIX OFF to browse files with filetype LISTING.
 -) Restore old definitions to PF10 and PF11 after command '('.
- The following synonyms may be used:
- G Get
 - P Put.

XEINSTL EXEC

```

/*****
/****
/**** XEINSTL      XEDIT  Extensions
/****
/****
/****
/****
/****

```

```

PF1  - Process all file to insert a number of columns;
PF2  - Process all file to delete a number of columns;
PF3  - Escape;
PF4  - Copy box to replace data;
PF5  - Move box to replace data;
PF6  - Copy box to insert data;
PF7  - Move box to insert data;
PF8  - Delete box;
PF9  - Process box to insert a number of columns;
PF10 - Process box to delete a number of columns;
PF11 - Fill in box with a character.

```

Figure 4: PFK settings to process a box

```

/***      SIZE 00031  VER 1.0 MOD 00                      ***/
/*****/
CLRSCRN
MESSAGE = 'user request'
SAY ' --- Start XEDIT/E 1.0 installation - reply Y or N'
PULL REPLY
IF REPLY = 'Y' THEN
SIGNAL ERROR
SET CMSTYPE HT
SIGNAL ON ERROR
MESSAGE = ' assemble 'XEHMA
ASSEMBLE XEHMA
ERASE XEHMA LISTING A
MESSAGE = ' load      'XEHMA
LOAD XEHMA '(' NOMAP NOLIBE AUTO RLDSAVE
MESSAGE = ' genmod    'XEHMA
GENMOD
ERASE XEHMA TEXT A
SIGNAL OFF ERROR
SET CMSTYPE RT
SAY ' --- XEDIT/E 1.0 installed successfully'
EXIT
ERROR:
SET CMSTYPE RT
SAY ' --- XEINSTL not properly executed -> 'MESSAGE

```

PROFILE XEDIT

```

/*****/
/***                      ***          ***/

```

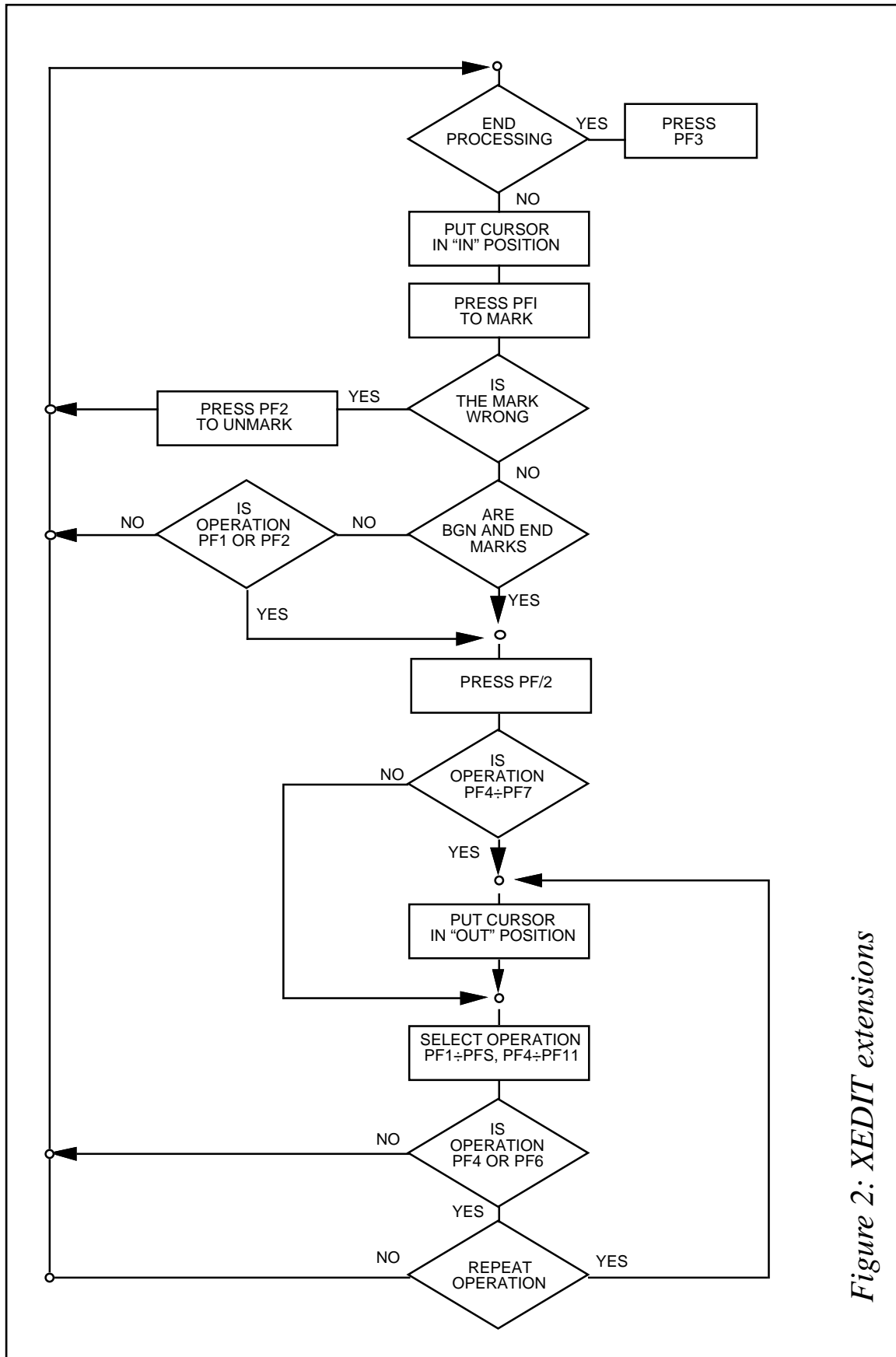


Figure 2: XEDIT extensions

```

/*** PROFILE          XEDIT  Extensions          ***      ***/
/***                                                         ***      ***/
/*****
/***   SIZE 00036  VER 1.0 MOD 00  TIME 16:35:47          ***/
/*****

PA2  ONLY      MACRO XEUNDO
DO I = 1 TO 12
  PF || I ONLY  MACRO XEPF || RIGHT(I, 2, '0')
END
ENT  IGNORE  CURS HOME
CASE      MIXED
CMDLINE   BOT
CURLINE   ON M
FULL      ON
MSGLINE   ON 24 0
NONDISP   .
NUMB      ON
NULLS     ON
PREF      NULL
TOFEOF    OFF
SCALE     OFF
SERIAL    OFF
STAY      ON
ZONE      1 '*'
SYN  G     GET
SYN  P     PUT
SYN  '('   MACRO XE1011ST
SYN  ')'   MACRO XE1011RS
SYN  Q     MACRO XECMS
TABS 1 3 5 7 9 11 13 25 45 60
EXT'/LR'
VER      OFF 1 MIN(73, LRECL.1)
11

```

XEPF01 XEDIT

```

/*****
/***                                                         ***      ***/
/*** XEPF01          XEDIT  Extensions          ***      ***/
/***                                                         ***      ***/
/*****
/***   SIZE 00017  VER 1.0 MOD 00  TIME 15:56:01          ***/
/*****

EXT'/CURS'
IF CURSOR.1 = 24 THEN
X XEHLF XEHLF '(' PROF XEHLF
ELSE
DO
  SOS PU

```

```

SOS LINEADD
SOS POP
END

```

XEPF02 XEDIT

```

/*****
/****
/**** XEPF02          XEDIT  Extensions          ****
/****
/****
/*****
/****  SIZE 00016  VER 1.0 MOD 00  TIME 16:58:16          ****
/*****
EXT'/CURS'
IF CURSOR.1 = 24 THEN
SAVE
ELSE
DO
    LE 0
    CUR S CURSOR.1 7
END

```

XEPF03 XEDIT

```

/*****
/****
/**** XEPF03          XEDIT  Extensions          ****
/****
/****
/*****
/****  SIZE 00028  VER 1.0 MOD 00  TIME 16:54:51          ****
/*****
EXT'/CURS/LI'
IF CURSOR.1 = 24 THEN
QUIT
ELSE
DO
    IF CURSOR.3 > 0 THEN
DO
    ':'CURSOR.3
EXT'/CURL'
POS = LENGTH(CURLINE.3) - VERIFY(REVERSE(CURLINE.3), ' ') - 50
LE 0
RI MAX(0, POS)
IF POS < 0 THEN
POS = POS + 58
ELSE
POS = 58

```

```

        ':'LINE.1
CUR S CURSOR.1 POS
END
END

```

XEPF04 XEDIT

```

/*****
/****
/**** XEPF04          XEDIT  Extensions          ****
/****
/****
/*****
/****  SIZE 00041  VER 1.0 MOD 00  TIME 16:04:35          ****
/*****
EXT'/CURS/LI/SIZ'
IF CURSOR.3 < 0 THEN
DO
  IF LINE.1 = 11 THEN
  DO
    ADDRESS CMS
    SAVE_POS = LINE.1
    GLOBALV SELECT XEDIT PUT 'SAVE_POS'
    ADDRESS XEDIT ':'11
  END
  ELSE
  DO
    ADDRESS CMS GLOBALV SELECT XEDIT GET SAVE_POS
    IF SAVE_POS < 1 | SAVE_POS > SIZE.1 THEN
      SAVE_POS = 11
      ':'SAVE_POS
    END
  EXIT
END
IF LINE.1 < 2 | CURSOR.4 < 0 THEN
EXIT
':'CURSOR.3
EXT'/CURL'
OLD_NUM = CURSOR.3
OLD_LINE = CURLINE.3
OLD_LINE_SET = 'Y'
ADDRESS CMS GLOBALV SELECT XEDIT PUT 'OLD_NUM OLD_LINE OLD_LINE_SET'
-1
EXT '/CURL'
1
CL ':'1'
CR SUBSTR(CURLINE.3, 1, CURSOR.4)
':'LINE.1

```

XEPF05 XEDIT

```

/*****
/***
/*** XEPF05          XEDIT  Extensions          ***
/***
/*****
/***  SIZE 00043  VER 1.0 MOD 00  TIME 16:00:29          ***
/*****

EXT'/CURS/LI/SIZ'
IF CURSOR.3 < 0 THEN
DO
  IF LINE.1 = SIZE.1 THEN
  DO
    ADDRESS CMS
    SAVE_POS = LINE.1
    GLOBALV SELECT XEDIT PUT 'SAVE_POS'
    ADDRESS XEDIT BOT
  END
ELSE
DO
  ADDRESS CMS GLOBALV SELECT XEDIT GET SAVE_POS
  IF SAVE_POS < 1 | SAVE_POS > SIZE.1 THEN
    SAVE_POS = SIZE.1
    ':' SAVE_POS
  END
EXIT
END
IF LINE.1 < 2 | CURSOR.4 < 0 THEN
EXIT
':' CURSOR.3
EXT '/CURL'
OLD_NUM = CURSOR.3
OLD_LINE = CURLINE.3
OLD_LINE_SET = 'Y'
ADDRESS CMS GLOBALV SELECT XEDIT PUT 'OLD_NUM OLD_LINE OLD_LINE_SET'
-1
EXT'/CURL/LR'
1
CL ':'CURSOR.4
CD LRECL.1
IF LENGTH(CURLINE.3) >= CURSOR.4 THEN
CR SUBSTR(CURLINE.3, CURSOR.4)
':'LINE.1

```

XEPF06 XEDIT

```

/*****
/***
/*** XEPF06          XEDIT  Extensions          ***
/***

```



```

/****
/*****
/**** SIZE 00013 VER 1.0 MOD 00 TIME 16:13:51 ****
/*****
EXT'/CURS'
IF CURSOR.1 = 24 THEN
LE 50
ELSE
PF06 SCHANG 9

```

XEPF07 XEDIT

```

/*****
/****
/**** XEPF07 XEDIT Extensions ****
/****
/*****
/**** SIZE 00028 VER 1.0 MOD 00 TIME 16:15:29 ****
/*****
EXT'/CURS/LI'
IF CURSOR.1 = 24 THEN
-21
ELSE
DO
IF CURSOR.3 < 0 | CURSOR.4 < 0 THEN
EXIT
': 'CURSOR.3
EXT'/CURL/'
SUBLINE = REVERSE(STRIP(SUBSTR(CURLINE.3, 1, CURSOR.4-1), 'T'))
POS = LENGTH(SUBLINE) - INDEX(SUBLINE, ' ') - 50
LE 0
RI MAX(0, POS)
IF POS < 0 THEN
POS = POS + 57
ELSE
POS = 57
': 'LINE.1
CUR S CURSOR.1 POS
END

```

XEPF08 XEDIT

```

/*****
/****
/**** XEPF08 XEDIT Extensions ****
/****
/*****
/**** SIZE 00029 VER 1.0 MOD 00 TIME 16:19:34 ****

```

```

/*****/
EXT'/CURS/LI'
IF CURSOR.1 = 24 THEN
21
ELSE
DO
  IF CURSOR.3 < 0 | CURSOR.4 < 0 THEN
  EXIT
  ':'CURSOR.3
  EXT'/CURL/'
  NEXT_POS = VERIFY(SUBSTR(CURLINE.3, CURSOR.4 + 1), ' ')
  POS = CURSOR.4 + NEXT_POS - 50 +
      INDEX(SUBSTR(CURLINE.3, CURSOR.4 + NEXT_POS + 1), ' ')
  LE 0
  RI MAX(0, POS)
  IF POS < 0 THEN
  POS = POS + 56
  ELSE
  POS = 56
  ':'LINE.1
  CUR S CURSOR.1 POS
END

```

XEPF09 XEDIT

```

/*****/
/***                                     ***      ***/
/*** XEPF09           XEDIT  Extensions      ***      ***/
/***                                     ***      ***/
/*****/
/***  SIZE 00013  VER 1.0 MOD 00  TIME 16:22:02      ***/
/*****/
EXT'/CURS'
IF CURSOR.1 = 24 THEN
RI 50
ELSE
SOS TABF

```

XEPF10 XEDIT

```

/*****/
/***                                     ***      ***/
/*** XEPF10           XEDIT  Extensions      ***      ***/
/***                                     ***      ***/
/*****/
/***  SIZE 00021  VER 1.0 MOD 00  TIME 16:09:24      ***/
/*****/
EXT '/CURS'

```

```

IF CURSOR.1 = 24 THEN
DO
  -5
  EXIT
END
ELSE
DO
  POS = CURSOR.2 - 10
  IF POS < 1 THEN
  POS = 80
  CUR S CURSOR.1 POS
END

```

XEPF11 XEDIT

```

/*****/
/****                                     ****      ****/
/**** XEPF11           XEDIT  Extensions          ****      ****/
/****                                     ****      ****/
/*****/
/****  SIZE 00021  VER 1.0 MOD 00  TIME 16:03:16          ****/
/*****/
EXT '/CURS'
IF CURSOR.1 = 24 THEN
DO
  5
  EXIT
END
ELSE
DO
  POS = CURSOR.2 + 10
  IF POS > 80 THEN
  POS = 1
  CUR S CURSOR.1 POS
END

```

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

Dobrin Goranov
Systems Programmer
Information Services (Bulgaria)

© Dobrin Goranov 1997

VM news

MiraSoft has announced LPRLaser. LPRLaser, general-purpose RSCS laser printer support, augments RSCS support for TCP/IP-connected printers. It includes an exit for the RSCS Line Printer Protocol (LPR) driver that supports external specification of printer capabilities. Installations can use LPRLaser to define how various printer facilities (eg fonts) are exploited so that users can take advantage of printer features and functions.

For further information contact:
MiraSoft Inc, 60 Alban Street, Boston, MA
02124-3709, USA.
Tel: (617) 825 9121.

* * *

IBM has made available a PCI version of the P/390 card on the PC Server 330 RAID model. The card contains a CMOS processor, PCI interface chip, and 32MB or 128MB ECC memory. The CPU executes the S/390 ESA instruction set and runs most of the VM/ESA and VSE/ESA applications with 32MB and 128MB memory. The P/390-PCI with 128MB is required for running OS/390 and MVS/ESA applications, and is recommended for VM/ESA and VSE/ESA applications where performance is a concern. Functioning as a co-processor in the PC Server System/390 and occupying a PCI card slot, the device uses the PC Server System/390 resources to emulate S/390 I/O devices. Prices weren't announced.

For further information contact your local IBM representative.

A number of vendors are supporting IBM's Magstar 3590 tape subsystem for ESCON attachment to System/390 enterprise servers. Computer Associates has CA-1 Release 5.2, CA-DYNAM/TLMS Release 5.4, CA-ASM2 Release 4.2, and CA-DYNAM for VM and VSE.

For further information contact:
Computer Associates, 1 Computer Associates Plaza, Islandia, NY 11788-7000, USA.
Tel: (516) 342 5224.
Computer Associates, Computer Associates House, 183-187 Bath Road, Slough, Berks, SL1 4AA, UK.
Tel: (01753) 577733.

* * *

Xephon is holding a briefing in London on the 12-13 May aimed at VM sites. The event is called *VM Update '97*, and is being held at the Chelsea Hotel, Sloane Street, London. The briefing takes a detailed look at various aspects of working with VM including future VM directions, integrating OfficeVision and the Web, connecting VM applications to Web browsers, coding a Pipeline stage in Assembler, creating a custom CSL, using the CSL direct interface to CMS files, and the CMS parsing facility. The attendance fee is £690 plus VAT. *VM Update* subscribers pay the discount price of £525 plus VAT.

For further information contact:
Xephon, 27-35 London Road, Newbury, Berks, RG14 1JL, UK.
Tel: (01635) 33823.



xephon