158

VM update

October 1999

In this issue

VM Update

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

North American office
Xephon/QNA
1301 West Highway 407, Suite 201-405
Lewisville, TX 75077-2150
USA
Telephone: 940 455 7050

Subscriptions and back-issues
A year's subscription to *VM Update*, comprising twelve monthly issues, costs £180.00 in the UK; $275.00 in the USA and Canada; £186.00 in Europe; £192.00 in Australasia and Japan; and £190.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the January 1990 issue, are available separately to subscribers for £16.00 ($23.00) each including postage.

Editor
Robert Burgess

***VM Update* on-line**
Code from *VM Update* can be downloaded from our Web site at http://www.xephon.com/vmupdate.html; 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*.

Optimizing processing with high-level languages

High-level languages, such as PL/I, COBOL, and Fortran, can speed up code writing, but at the same time they can create inefficiency. As a result, when high-level language code is executed, performance degradation may be observed. So, it is useful to take advantage of fast Assembler code to optimize data processing, at the same time extending the capabilities of high-level languages.

CMSFRW is a subroutine intended to accelerate file input/output. It provides a common interface to CMS macros that process disk files and allows programmers to utilize their full power.

The functions supported by CMSFRW are:

- Dynamic opening for reading or writing of CMS files on arbitrary accessible mini-disks, with no FILEDEF command needed.
- Executing implicit CLOSE on endfile events for input files, and performing explicit CLOSE for output files.
- Supporting the specification of record number, which is the defined starting position in a file for a given read or write operation.
- Writing self-defining records, containing their length in the first two bytes of each record, into variable-format CMS files.
- Erasing files, specified by filename, filetype, and filemode.
- Buffering fixed-length files and blocking read/write operations.

CMSFRW is written in Assembler and was created in CMS with VM/SP Release 5. The size of CMSFRW is 1,219 bytes. At execution time, when fixed-length files are processed, a buffer area of 256KB is allocated. For successful execution, the virtual machine size must be at least 4MB.

The CMSFRW CALL parameter list (PPLIST) includes the following parameters:

```
CALL CMSFRW (req,fn,ft,fm,buf,buf_len,read_len,start_rec_no,format)
```

where:

- ‘req’ – 1 byte, character.
- ‘fn’ – 8 bytes, character.
- ‘ft’ – 8 bytes, character.
- ‘fm’ – 2 bytes, character.
- ‘buf’ – 4 bytes, pointer.
- ‘buf_len’ – 4 bytes, binary.
- ‘read_len’ – 4 bytes, binary.
- ‘start_rec_no’ – 4 bytes, binary.
- ‘format’ – 1 byte, character.

The parameter ‘req’ specifies the requested operation and can have the following values:

- C – close output file.
- D – delete file.
- R – read operation.
- W – write operation.

After CALL, ‘req’ contains a copy of the first byte of register 15. For proper program execution, the user must always verify that the value of req is X'00'. A different value of req means that the requested action has failed, with the corresponding CMS macro return code. Note that when the read operation is executed, the req value of X'0C' signals the end of the file.

The parameters ‘fn’, ‘ft’, and ‘fm’ specify the CMS identifier of the processed file.

The parameters ‘buf’ and ‘buf_len’ set the address and length of the file record buffer.

If a variable-length file is read, then, after CALL, the parameter ‘read_len’ contains the actual length of the read record in the buffer.

The parameter ‘start_rec_no’ specifies the starting record number for read and write operations. Note that when file ‘fn ft fm’ exists, and the write operation is executed, if ‘start_rec_no’ has a value of 0, then records append to the end of file.

The parameter ‘format’ defines the file record format and can have the following values:

- ‘F’ – fixed-length record.
- ‘V’ – variable-length record.
- ‘W’ – self-defining records to be written with variable-length.

An example of parameter descriptions in PL/I follows:

```
DCL CMSFRW ENTRY (CHAR (1),
                  CHAR (8),
                  CHAR (8),
                  CHAR (2),
                  *,
                  FIXED BIN (31),
                  FIXED BIN (31),
                  FIXED BIN (31),
                  CHAR (1))
     OPTIONS (ASM INTER);
```

EXAMPLES OF CMSFRW USE

The following examples are created with PL/I, but they are applicable to any other high-level language, with minor modifications. All examples use the following common declarations:

```
DCL CH_1 CHAR (1) AUTO;
DCL CH_1Ø_F CHAR (1Ø) AUTO;
DCL CH_1Ø_V CHAR (1Ø) VAR AUTO;
DCL READ_LEN FIXED BIN (31) AUTO;
```

- Reading file ‘1 F A’ with a fixed-length record of 10 bytes:

```
DO UNTIL (UNSPEC(CH_1) = 'ØØØØ11ØØ'B); /* RG15 = X'ØC' -> EOF */
    CH_1 = 'R';
     CALL CMSFRW (CH_1, '1', 'F', 'A', CH_1Ø, 1Ø, Ø, Ø, 'F');
END;
```

- Reading file ‘1 F A’, starting from record number 3:

```
DO UNTIL (UNSPEC(CH_1) = 'ØØØØ11ØØ'B); /* RG15 = X'ØC' -> EOF */
    CH_1 = 'R';
      CALL CMSFRW (CH_1, '1', 'F', 'A', CH_1Ø, 1Ø, Ø, 3, 'F');
END;
```

- Reading file ‘1 V A’ with a variable-length record of 10 bytes:

```
DO UNTIL (UNSPEC(CH_1) = 'ØØØØ11ØØ'B); /* RG15 = X'ØC' -> EOF */
    CH_1 = 'R';
     CALL CMSFRW (CH_1, '1', 'V', 'A', CH_1Ø, 1Ø, READ_LEN, Ø, 'V');
    CH_1Ø_V = SUBSTR (CH_1Ø, 1, READ_LEN);
END;
```

- Writing 7 records to file ‘1 F A’ with a fixed-length record of 10 bytes:

```
DO I = 1 TO 7;
    CH_1Ø = I;
    CH_1 = 'W';
     CALL CMSFRW (CH_1, '1', 'F', 'A', CH_1Ø, 1Ø, Ø, Ø, 'F');
END;

    CH_1 = 'C';
     CALL CMSFRW (CH_1, '1', 'F', 'A', Ø, Ø, Ø, Ø, 'F');
```

- Writing 7 records to file ‘1 V A’ with a variable-length record of 10 bytes:

```
DO I = 1 TO 7;
  CH_1Ø_V = I;
  CH_1Ø = CH_1Ø_V;
   CH_1 = 'W';
   CALL CMSFRW (CH_1, '1', 'V', 'A', CH_1Ø, LENGTH(CH_1Ø_V), Ø, Ø, 'V');
END;

CH_1 = 'C';
 CALL CMSFRW (CH_1, '1', 'V', 'A', Ø, Ø, Ø, Ø, 'V');
```

- Writing 7 self-defined records to file ‘1 V A’ with a variable-length record of 10 bytes:

```
DO I = 1 TO 7;
  CH_1Ø_V = I;
  CH_1 = 'W';
  CALL CMSFRW (CH_1, '1', 'V', 'A', CH_1Ø_V, Ø, Ø, Ø, 'W');
END;

CH_1 = 'C';
CALL CMSFRW (CH_1, '1', 'V', 'A', Ø, Ø, Ø, Ø, 'W');
```

- Erasing file ‘1 E A’:

```
CH_1 = 'D';
 CALL CMSFRW (CH_1, '1', 'E', 'A', Ø, Ø, Ø, Ø, '');
```

BENCHMARKS

The productivity of CMSFRW was examined during real file processing, with files ranging in size from 55MB to 500MB, on 3380 devices with a 3MB per second transfer rate. The number of read records was between 2,543,000 and 3,363,000. A comparison was made between natural PL/I program code and PL/I code containing a CALL to CMSFRW instead of the native PL/I READ statement. This showed that CMSFRW increased system performance by approximately three times.

CMSFRW ASSEMBLE

```
***********************************************************************
****                                                  ***         ****
**** CMSFRW           CMS read/write driver           ***         ****
****                                                  ***         ****
***********************************************************************
*                                                                     *
CMSFRW   CSECT
         SAVE  (14,12)
         BALR  12,Ø
         USING *,12
         ST    13,SA+4
         LA    13,SA
         L     2,Ø(1)
         LM    3,9,4(1)
         MVC   FN(8),Ø(3)
         MVC   FT(8),Ø(4)
         MVC   FM(2),Ø(5)
         LA    11,1
         L     1Ø,Ø(9)
         CLC   RECNO(4),=F'-1'
         BNE   DONTSET
         ST    1Ø,RECNO
DONTSET  EQU   *
         L     7,Ø(7)
         LA    3,FN
         L     4,32(1)
         CLI   Ø(4),C'W'
         BNE   SETRECFM
         IC    4,=C'V'
         LH    7,Ø(6)
         LA    6,2(6)
         B     PROC
SETRECFM EQU   *
         IC    4,Ø(4)
PROC     EQU   *
         ST    6,RGØ6
         CLI   Ø(2),C'R'
         BE    CHECKF
         CLI   Ø(2),C'W'
         BNE   PROCREQ
CHECKF   EQU   *
         CLI   ALLOCFOR,X'ØØ'
         BNE   PROCREQ
         STH   7,LRECL
         SR    Ø,Ø
         CLM   4,1,=C'F'
         BE    DOBLOCK
         LR    1,7
         B     COUNTREC
DOBLOCK  EQU   *
         L     1,=A(1Ø24*256)
COUNTREC EQU   *
         DR    Ø,7
         ST    1,NOREC
         MH    1,LRECL
         ST    1,BUFLEN
         LA    Ø,7(1)
         SRL   Ø,3
         DMSFREE DWORDS=(Ø),TYPE=USER,ERR=RET
         ST    1,BUFADDR
         MVC   ALLOCFOR(1),Ø(2)
         XC    BUFPOS(4),BUFPOS
         CLI   Ø(2),C'R'
         BNE   PROCREQ
         L     11,NOREC
         L     7,BUFLEN
         L     6,BUFADDR
         FSREAD (3),RECFM=(4),BUFFER=(6),BSIZE=(7),                    X
               RECNO=(1Ø),NOREC=(11),FORM=E
         ST    Ø,BUFLEN
         XC    RECNO(4),RECNO
         CLM   4,1,=C'F'
         BE    PROCREQ
         ST    Ø,Ø(8)
PROCREQ  EQU   *
         CLI   Ø(2),C'R'
         BE    READ
         CLI   Ø(2),C'W'
         BE    WRITE
         CLI   Ø(2),C'D'
         BE    DELETE
         CLI   Ø(2),C'C'
         BE    CLOSE
         MVI   Ø(2),X'FF'
         B     MISS
WRITE    EQU   *
         L     Ø,RGØ6
         L     14,BUFADDR
         A     14,BUFPOS
         BAL   5,CHECKBUF
         FSWRITE (3),RECFM=(4),BUFFER=(6),BSIZE=(7),                   X
               RECNO=(1Ø),NOREC=(11),FORM=E
         XC    RECNO(4),RECNO
         B     RET
READ     EQU   *
         L     Ø,BUFADDR
         A     Ø,BUFPOS
         L     14,RGØ6
         BAL   5,CHECKBUF
         FSREAD (3),RECFM=(4),BUFFER=(6),BSIZE=(7),                    X
               RECNO=(1Ø),NOREC=(11),FORM=E
         ST    Ø,BUFLEN
         CLM   4,1,=C'F'
         BE    CHECKEOF
         ST    Ø,Ø(8)
CHECKEOF EQU   *
         CLM   15,1,=X'ØC'
         BE    CLOSE
         B     RET
DELETE   EQU   *
         FSERASE (3)
         B     RET
CLOSE    EQU   *
         CLI   ALLOCFOR,X'ØØ'
         BE    DOCLOSE
         CLI   ALLOCFOR,C'R'
         BE    FREEMEM
         L     7,BUFPOS
         LTR   7,7
         BZ    FREEMEM
         L     6,BUFADDR
         SR    1Ø,10
         LR    11,7
         LH    1,LRECL
         DR    1Ø,1
         L     1Ø,RECNO
         FSWRITE (3),RECFM=(4),BUFFER=(6),BSIZE=(7),                   X
               RECNO=(1Ø),NOREC=(11),FORM=E
FREEMEM  EQU   *
         L     1,NOREC
         MH    1,LRECL
         LA    Ø,7(1)
         SRL   Ø,3
         L     1,BUFADDR
         DMSFRET DWORDS=(Ø),LOC=(1)
DOCLOSE  EQU   *
         FSCLOSE (3)
         LA    15,12
         MVI   ALLOCFOR,X'ØØ'
         MVC   RECNO(4),=F'-1'
```

```
RET      EQU   *
         XC    Ø(1,2),Ø(2)
         LTR   15,15
         BZ    MISS
         CLM   15,1,=X'ØC'
         BNE   CLOSE
         STCM  15,1,Ø(2)
MISS     EQU   *
         L     13,4(13)
         RETURN (14,12)
CHECKBUF EQU   *
         LH    1,LRECL
         LR    15,1
         MVCL  14,Ø
         L     Ø,BUFPOS
         AH    Ø,LRECL
         ST    Ø,BUFPOS
         C     Ø,BUFLEN
         BL    RET
         XC    BUFPOS(4),BUFPOS
         L     11,NOREC
         L     1Ø,RECNO
         L     7,BUFLEN
         L     6,BUFADDR
         BR    5
SA       DC    2ØF'Ø'
BLOCK    DS    F
NOREC    DS    F
RECNO    DC    F'-1'
BUFLEN   DS    F
BUFADDR  DS    F
BUFPOS   DS    F
RGØ6     DS    F
FN       DS    CL8
FT       DS    CL8
FM       DS    CL2
LRECL    DS    H
ALLOCFOR DC    X'ØØ'
         END   CMSFRW
```

PROCESSING NUMEROUS FILES

The code of CMSFRW is not re-enterable and so it can process only one file at a time. To process several files, you should make the corresponding number of copies of CMSFRW, ie CMSFRW1, CMSFRW2, etc. This will not have any effect on system performance.

Dobrin Goranov
Information Services Co (Bulgaria)

Deleting selected lines from a file

GENERAL DESCRIPTION

ALLDEL and NALLDEL are two XEDIT macros that delete selected lines from a file:

- ALLDEL deletes lines containing the specified targets.
- NALLDEL deletes lines not containing those targets.

Although this may sound similar to the standard ALL macro followed by a delete operation, the main difference is that these macros accept multiple targets (up to 20), separated by a standard locate operator:

```
ALLDEL/this week/next week/next month/
NALLDEL $/*$if$then$else$do$end
```

Like the ALL macro, these macros do not affect zone setting. To avoid deleting unwanted lines, you must ensure that zone setting is as you require, especially on NALLDEL.

Because you are deleting multiple lines, the only way to recover them correctly is to quit XEDIT and risk losing any previous changes. As a precaution, I always save my file before using these macros, especially when specifying multiple targets.

Finally, an important observation – these macros act only upon displayed lines, which means hidden lines are not affected. This is a matter of personal preference, and also of safety. It also means that I do not change set select and set display. I ensure that scope display is in effect.

ALLDEL

```
/*==================================================================*/
/*  ALLDEL macro for XEDIT                                          */
/*  This macro deletes all lines containing the selected targets.   */
/*  Up to 20 targets can be specified simultaneously.               */
/*  Example: ALLDEL/Darth Vader/Jabba/                              */
/*==================================================================*/
```

```
cmd = "command"
parse arg arg1
arg1 = strip(arg1)
sep = left(arg1,1)
if right(arg1,1) <> sep then arg1=arg1||sep
x = 2
do k = 1 to 2Ø
   y = pos(sep,arg1,x)
   if y=Ø | y=x then leave
   z = y-x
   target.k = substr(arg1,x,z)
   x = y+1
end
maxtarget = k-1
cmd "extract/line/wrap/msgmode/stay"
first = line.1
cmd "set wrap on"
cmd "set msgmode off"
cmd "set stay on"
cmd "set scope display"
do k = 1 to maxtarget
   cmd "top"
   do forever
      cmd "locate" sep||target.k
      if rc <> Ø then leave
      cmd "extract/line/"
      if line.1 < first then first = first-1
      cmd "delete 1"
   end
end
":" first
cmd "set msgmode" msgmode.1
cmd "set stay" stay.1
cmd "set wrap" wrap.1
exit
```

NALLDELL

```
/*===================================================================*/
/*  NALLDEL macro for XEDIT                                          */
/*  This macro deletes all lines not containing the selected targets. */
/*  Up to 2Ø targets can be specified simultaneously.                */
/*  Example: NALLDEL/Princess Leia/R2D2/C3PO                         */
/*===================================================================*/

cmd = "command"
parse arg arg1
arg1 = strip(arg1)
```

```
sep = left(arg1,1)
if right(arg1,1) <> sep then arg1=arg1||sep
x = 2
do k = 1 to 2Ø
   y = pos(sep,arg1,x)
   if y=Ø | y=x then leave
   z = y-x
   target.k = substr(arg1,x,z)
   x = y+1
end
maxtarget = k-1
cmd "extract/size/line"
first = line.1
newfirst = first
cmd "preserve"
cmd "set wrap off"
cmd "set msgmode off"
cmd "set scope display"
do k = 1 to maxtarget
   cmd "top"
   do forever
      cmd "locate" sep||target.k
      if rc <> Ø then leave
      cmd "extract/line/"
      x = line.1
      table.x = "Y"
   end
end
cmd "top"
x = 1
do z = 1 to size.1
   ":" x
   if rc <> Ø then leave
   if table.z = "Y" then do
      x = x+1
      iterate
   end
   cmd "delete"
   if z < first then newfirst = newfirst-1
end
cmd "restore"
":" newfirst
exit
```

Luis Paulo Figueiredo Sousa Ribeiro
Systems Engineer
Edinfor (Portugal)

Exploring the deeper levels of VM

On 1 January 1999 the new euro currency was introduced in many European countries. For the financial sector this represented a big challenge – the euro had to be incorporated into existing applications, often in co-existence with the old currency.

Our company had to change over 4,000 COBOL programs, hundreds of VSAM files, several databases, and lots of JCL. These changes all went into production together on 1 January 1999, making this the biggest cut-over we had ever done.

Our site has two mainframes – one for running a productionVM/ESA Version 2.2 (called 'VM1'), and one for running a development and test VM/ESA Version 2.2 (called 'VM2'). Both VM systems host several VSE/ESA Version 2.2 guests. Our DASD reside on EMC Symmetrix boxes.

Normally, we test our programs in the development and test system (VM2), using programmer-supplied test data. But for the euro project that was not enough. End users also wanted to test the new programs with real data, and we wanted to simulate the scenario for the conversion weekend itself as realistically as possible.

We also saw the Year 2000 problem coming towards us, seemingly faster and faster. It was clear that we needed a new test environment that allowed us to cope with these new requirements.

ANOTHER LEVEL

After considering some possible solutions, we decided to use one of the wonderful facilities of VM – to start a VM 'in second level' under an existing VM. This second-level VM had to be a copy of our production VM (VM1), and had to be IPLed in a virtual machine on our development-and-test VM (VM2). In this second-level VM (we call it 2nd-VM1) we could then run the programs to convert our databases and VSAM files with 'real' production data, after which the end users could test the new euro programs and applications. And, most importantly, the process of copying, converting, and testing could be done over and over again.

The second-level VM allowed us to test our euro conversion programs several times. We could even simulate the correct date, by altering the system date during the IPL of the second-level VM. End users filtered out many errors while testing the new or updated applications, which were corrected and tested again. As a result, the euro conversion at our site was a great success.

Right now we are using the same second-level VM infrastructure to tackle the Y2K problem. One of the things we have already encountered is that some software, although being confirmed as 'Y2K-compliant' by the vendor, in fact is not Y2K-ready. There are still many bugs in the Y2K arena. One piece of software wouldn't even accept the licence key for the year 2000 when running with a system date in 2000!

HOW TO SET UP A SECOND-LEVEL VM

How do you set up a second-level VM testing environment? There are several steps:

- Acquire enough disk space (3390) to hold an extra copy of the production data.
- Find a flexible tool to repeatedly make a fast copy of the production data.
- Create a first-level VM user to host the second-level VM.
- Customize the 'original' first-level VM in anticipation of being IPLed as a second-level VM.

COPYING DASD

The first two steps were easy. Because we already had an EMC Symmetrix DASD infrastructure, we asked our boss to buy some extra disk space from EMC. We asked for the EMC TIMEFINDER feature as well. TIMEFINDER makes it possible to copy DASD full-packs at the hardware level. All you have to do is pass to TIMEFINDER a list of DASD to be copied in the EMC Symmetrix box. Of course, DDR is an alternative to TIMEFINDER, but it takes a lot more (CPU) time and system overhead, and increases system downtime.

CREATING A LEVEL TWO USER ON VM2

Now that a copy of the production VM is available, the next step is to IPL it in a virtual machine on our VM2 test system. The following code shows the directory entry for that user, called 'LEVEL2':

```
USER LEVEL2   PASSWORD   256M 256M BDEFG
MACHINE ESA
OPTION CPUID Ø6592Ø MIH TODENABLE QUICKDSP
IPL CMS
CONSOLE 54D 327Ø  T
SPOOL ØØC 254Ø READER *
SPOOL ØØD 254Ø PUNCH  A
SPOOL ØØE 3211 A
LINK MAINT 19Ø 19Ø RR
LINK MAINT 193 193 RR
LINK MAINT 19D 19D RR
LINK MAINT 19E 19E RR
MDISK 191 339Ø 52Ø ØØØ2 EMC61A ALL READ WRITE MULTI

* 3174 Control Unit
DEDICATE 58Ø 58Ø

* Non SNA 327Ø Screens for DIAL
SPECIAL 5ØØ 327Ø
SPECIAL 5Ø1 327Ø
SPECIAL 5Ø2 327Ø
SPECIAL 5Ø3 327Ø
SPECIAL 5Ø4 327Ø
SPECIAL 5Ø5 327Ø
SPECIAL 5Ø6 327Ø
```

You should pay attention to the operator-console address in the CONSOLE directory statement. This also has to be present in the VM/ESA Version 2 'SYSTEM CONFIG' file in the 'OPERATOR_CONSOLES' entry. We used the same address as the real production VM console. The 'TODENABLE' option allows you to alter the system date in the LEVEL2 virtual machine. The 'SPECIAL' statements provide some non-SNA terminals to allow the 'DIAL LEVEL2' command to get a VM logo screen.

Note that the LEVEL2 user can't perform class A commands. That prevents you from accidentally performing a SHUTDOWN command in the first-level VM user, instead of in the second-level VM (yes, we learned this the hard way).

When the LEVEL2 user has been logged on, the disks containing the copy of the production data have to be varied on, and attached to, the LEVEL2 virtual machine. The ATTACH is done using a virtual address that corresponds to the real address of the disk on the real production VM, so that the second-level VM will find its disks with exactly the same addresses as on the real production VM. This was necessary because, in the SYSTEM CONFIG file of our real production VM, the addresses of the disks containing the production copy were set to 'offline_at_ipl', to avoid duplicate labels when IPLing our real production VM. Because the second-level VM will use that same SYSTEM CONFIG file (a copy) to start itself up, we have to present the copy-disks to the second-level VM using the disk-addresses from the production system.

It sounds more complicated than it really is – just remember that everything that's virtual for user LEVEL2 becomes real for the second-level VM running in user LEVEL2.

You may have noticed that, initially, CMS will be IPLd in the LEVEL2 user. This allows us to put all the necessary commands for attaching disks and the IPL command in STARTUP EXEC, a simplified version of which is provided here. The virtual device to be IPLed is, of course, the disk containing the copy of the production IPL disk.

```
/* STARTUP EXEC for 2nd-level VM */

/* rdev = real first level device addr. of the disk to be attached */
/* vdev = virtual device addr. of the disk, appears real to the 2nd-
   level VM */

'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
...
...
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'
'CP ATTACH rdev TO LEVEL2 AS vdev'

nl='15'x
/* do the IPL */
/* vdev is the virtual device address of the VM-ipl-disk (220RES),
   appears real to the 2nd-level VM */
```

```
/* 54D  is the virtual device address of the operator-console,
   appears real to the 2nd-level VM */
'CP TERMINAL CONMODE 3270 'nl'CP IPL vdev LOADPARM 54D'
```

The IPL command activates the SAPL (Stand Alone Program Loader) which in turn will IPL VM, using the user-defined defaults. But by supplying the address of the console as a LOADPARM on the IPL statement, we force SAPL to show the SAPL options screen, on which you can alter defaults before actually IPLing VM itself.

The SAPL parameter we have to modify is the CP LOAD address. On the real production VM we have defined a 380MB V=R zone, and, as a consequence, the CP LOAD address has to be defined higher, ie 380MB+4KB. The LEVEL2 user, however, only has 256MB of virtual storage (which will become real for the second-level VM, remember). So unless we change the CP LOAD address, CP would be loaded outside the memory, which, of course is deadly for any program. Also, in a second-level VM you cannot define a V=R zone. So we change the CP LOAD address to a modest '1000'X.

After IPL, the second-level VM comes up and asks you exactly the same start-up questions as on a first-level system. You can now alter the system date and time if you wish.

The terminal on which you performed the IPL will become the console for the second-level VM, and the OPERATOR user will be logged on. You can toggle from the second-level OPERATOR user to the first-level LEVEL2 user by pressing the PA1 key (Break key).

CUSTOMIZING THE ‘ORIGINAL’ FIRST-LEVEL VM

When we had our second-level VM running, we realized that it would be a good idea to customize a few things on our real production VM, anticipating that they would only become active on the second-level VM system.

The first thing you notice is that it’s not easy to distinguish the original system from the copy. They both use the same VM logos, the same USSTAB VTAM screens, the same system identifier, etc. But there’s a solution to everything. For example, the SYSTEM CONFIG file can be prepared to supply different system identifiers depending on the CPU serial number on which VM is running. The following code results in a system identifier 'VM1' when running on the real production CPU, and a system identifier '2nd-VM1' when running as a second-level VM on the test CPU:

```
System_identifier_default          VM1
System_identifier  *   %65920  2nd-VM1
```

It's also a good idea to create a different VM logo for the '2nd-VM1' system. Again, the SYSTEM CONFIG offers a solution. Instead of using the standard LOCAL CONFIG file to reference the set of VM logos to be used, you can provide a 'Logo_Config' statement in SYSTEM CONFIG as follows:

```
      Logo_Config  -system- CONFIG
```

At IPL time, the '-system-' parameter will be replaced by the appropriate system identifier. When running on the production CPU, a file called VM1 CONFIG will be used. When running as a second-level VM on the test CPU, a file called '2ND-VM1 CONFIG' will be used. You can then reference a specific set of logo files in these files. VM logos can easily be created by using the IBM-supplied DRAWLOGO EXEC (remember to put all the files on the CF1/CF2 CP parameter disks before you IPL). Using the 'DIAL LEVEL2' command on the first-level system (the system where user LEVEL2 is running) you get the VM logo from the second-level VM.

AVOIDING CONFUSED END USERS

To be certain that end users are testing in the second-level VM, and not in the real production system (which can be catastrophic), we attached one 3174 SNA terminal control unit directly to the second-level VM, and placed the terminals in a separate 'test' room. The selected end users have to test their applications on these terminals.

To make end users even more aware of the situation, we defined an eye-catching USSTAB for these terminals in VTAM. Note that all the necessary definitions for this control unit (in VM and VTAM) were done on the real production VM. Of course, they don't work there because the 3174 isn't available there. That way we avoid having to redo these definitions in the second-level VM every time we make a new copy of the real production VM.

AND WHAT ABOUT VSE ?

Our VSE guests did not require any change. They work exactly the same on the 2nd-VM1 system as they do on the VM1 system. Again, this shows how good VM is at creating a virtual world. We also succeeded in establishing a (virtual) channel-to-channel POWER PNET connection between a VSE guest on our development and test system and a VSE guest on the 2nd-VM1 system, in order to provide a channel for downloading new versions of euro programs.

HOW TO USE REAL DEVICES IN SECOND-LEVEL VM

Real devices such as printers, tapes, and cartridges can be made available to the second-level VM simply by attaching them to the LEVEL2 user. They then appear as real devices to the second-level VM.

For example, to make tape unit 495 available to user MAINT in 2nd-VM1, we do a 'CP ATTACH 495 LEVEL2' on VM2 (the first level VM) followed by a 'CP ATTACH 495 MAINT' on 2nd-VM1 (the second-level VM). The device immediately becomes available in the second-level VM through device sensing (if supported by VM for that device type). Remember that some older device types still have to be predefined in VM.

MISSION ACCOMPLISHED

And there it is, an extra test environment that meets all our requirements: flexible testing, fast refresh of production data, Y2K system date, etc. The only drawback I have to mention is poor performance. A second-level VM-system implies double, sometimes triple, paging (VSE). CPU consumption is high as well. The 'CP SET SHARE' command can be helpful here.

But still, we are very pleased with this solution. For once, the guys from the PC department were jealous of us instead of the other way round.

Geert Dieltiens
Systems Programmer
Informatica J Van Breda & Co (Belgium)

REXX/CMS talks to VB over TCP/IP – part 2

This month we conclude the code for the InfoServer, allowing direct access to various kinds of VM system information directly from Word or Excel.

IS0003

```
/*====================================================================*/
/* Name        :  IS0003   REXX                                       */
/*====================================================================*/
/* Application :  InfoServer                                          */
/*                                                                    */
/* Usage       :  Pipeline Stage Command                              */
/*                                                                    */
/* Arguments   :  -                                                   */
/*                                                                    */
/* Result      :  -                                                   */
/*                                                                    */
/* Function    :  Request Handler for REQ3 (DB2 Table Description)    */
/*                                                                    */
/* InStream  0 :  Request_Record                                      */
/* OutStream 0 :  Requested_Data_Stream_for_Client                    */
/*                                                                    */
/*====================================================================*/

/*====================================================================*/
/* main processing logic                                              */
/*====================================================================*/
,readto request_record'
eor_marker='#eor#'
eob_marker='#eob#'
parse value request_record with req_id req_data
dbname=translate(word(req_data,1))
creator=translate(word(req_data,2))
tabname=translate(word(req_data,3))
call do_connect
call do_selects
ADDRESS COMMAND  ,RXSQL COMMIT RELEASE'
return

/*====================================================================*/
/* perform SQL CONNECT                                                */
/*====================================================================*/
do_connect:
USER_STRING='SQLOWNER IDENTIFIED BY HANDSOFF'
ADDRESS COMMAND  ,RXSQL CONNECT' user_string ,TO' dbname
sqlrc=rc
```

```
if sqlcode ¬= Ø | sqlrc ¬= Ø
then
do
call error_message
return
end
return

/*=================================================================*/
/* get and prepare the table data                                  */
/*=================================================================*/
do_selects:

/*=================================================================*/
/* Get Table infos from SYSCATALOG                                 */
/*=================================================================*/
stmt='select dbspacename,remarks' ,
,from system.syscatalog',
"where tname = ,"strip(tabname)"' and creator = ,"strip(creator)"'"
address command ,RXSQL PREP STAT1' stmt
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

address command ,RXSQL OPEN STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,
,RXSQL FETCH STAT1 DBSPACE, TABREMARKS'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

address command ,RXSQL CLOSE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
```

```
then
do
call error_message
return
end

if strip(tabremarks) = ,' then tabremarks='No remarks on table'
orec=left(tabremarks,254) || eob_marker

/*==================================================================*/
/* Get Column Info from SYSCOLUMNS                                  */
/*==================================================================*/

stmt='SELECT' ,
,CNAME,COLTYPE,LENGTH,CLABEL,REMARKS,FLDPROC,NULLS,COLNO' ,
,FROM SYSTEM.SYSCOLUMNS' ,
"WHERE TNAME = ,"STRIP(TABNAME)"' AND CREATOR = ,"STRIP(CREATOR)"'",
"ORDER BY COLNO"
address command ,RXSQL PREP STAT1' stmt
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

address command ,RXSQL OPEN STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

do forever
address command ,
,RXSQL FETCH STAT1 W_CNAME, W_COLTYPE, W_LENGTH, W_LABEL,' ,
,W_REMARKS, W_FLDPROC, W_NULLS, W_COLNO'
if rc =4 &  sqlcode = 1ØØ then leave
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

if w_label = ,W_LABEL' then w_label=w_remarks
orec=orec || left(w_colno,5)  ,
|| left(w_cname,3Ø) ,
|| left(w_coltype,12) ,
|| left(w_length,12) ,
|| left(w_nulls,1) ,
|| left(w_fldproc,8) ,
|| left(w_label,40)
end

orec=orec || eob_marker
address command ,RXSQL CLOSE STAT1'
if rc ¬= 0 | sqlcode ¬= 0
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT1'
if rc ¬= 0 | sqlcode ¬= 0
then
do
call error_message
return
end

/*=================================================================*/
/* Get Key Info from SYSKEYS, SYSKEYCOLS                            */
/*=================================================================*/

stmt='SELECT' ,
,KEYNAME, KEYTYPE,INAME,REFTNAME,DELETERULE,STATUS' ,
,FROM SYSTEM.SYSKEYS',
"WHERE TNAME = ,"STRIP(TABNAME)"' AND TCREATOR = ,"STRIP(CREATOR)"'",
address command ,RXSQL PREP STAT1' stmt
if rc ¬= 0 | sqlcode ¬= 0
then
do
call error_message
return
end

address command ,RXSQL OPEN STAT1'
if rc ¬= 0 | sqlcode ¬= 0
then
do
call error_message
return
end

do forever
address command ,
,RXSQL FETCH STAT1 KEYNAME, KEYTYPE, INAME, REFTNAME,' ,
,DELETERULE, STATUS'
if rc =4 &  sqlcode = 100 then leave
if rc ¬= 0 | sqlcode ¬= 0
then
```

```
do
call error_message
return
end

orec=orec || left(,Key Name',2Ø),
|| left(keyname,6Ø)
orec=orec || left(,Key Type',2Ø),
|| left(decode(,1'keytype),6Ø)
if keytype ¬= ,F'
then
do
orec=orec || left(,Index',2Ø),
|| left(iname,6Ø)
end
else
do
orec=orec || left(,Parent Table',2Ø),
|| left(reftname,6Ø)
orec=orec || left(,Delete Rule',2Ø),
|| left(decode(,2'deleterule),6Ø)
end

orec=orec || left(,Status',2Ø),
|| left(decode(,3'status),6Ø)
call get_key_columns
orec=orec || left(,Columns',2Ø),
|| left(keycols,6Ø)
orec=orec || eor_marker
end

orec=orec || eob_marker
address command ,RXSQL CLOSE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

/*==================================================================*/
/* Get Index Info from SYSINDEXES                                   */
/*==================================================================*/

stmt='SELECT' ,
,INAME,COLNAMES,INDEXTYPE,CLUSTER,CLUSTERRATIO,KEYTYPE' ,
,FROM SYSTEM.SYSINDEXES' ,
"WHERE TNAME = ,"STRIP(TABNAME)"' AND CREATOR = ,"STRIP(CREATOR)"'"
address command ,RXSQL PREP STAT1' stmt
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL OPEN STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

do forever
address command ,
,RXSQL FETCH STAT1' ,
,INAME,COLNAMES,INDEXTYPE,CLUSTER,CLUSTERRATIO,KEYTYPE'
if rc =4 &  sqlcode = 1ØØ then leave
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

if w_label = ,W_LABEL' then w_label=''
orec=orec || left(iname,2Ø) ,
|| left(colnames,7Ø) ,
|| left(decode(,4'indextype),4Ø) ,
|| left(decode(,5'cluster),4Ø) ,
|| left(clusterratio' (Maximum: 1ØØØØ)',4Ø) ,
|| left(decode(,6'keytype),4Ø) ,
|| eor_marker
end
orec=orec || eob_marker
address command ,RXSQL CLOSE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
```

```
call error_message
return
end

/*=====================================================================*/
/* Get DBSpace Info from SYSDBSPACES                                   */
/*=====================================================================*/
STMT='SELECT DBSPACENO,OWNER,NTABS,NPAGES,LOCKMODE,POOL',
,FROM SYSTEM.SYSDBSPACES',
"WHERE DBSPACENAME = ,"STRIP(DBSPACE)"'"

address command ,RXSQL PREP STAT1' stmt
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

address command ,RXSQL OPEN STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end

address command ,
,RXSQL FETCH STAT1 DBSPACENO,DBSPACEOWNER,NTABS,NPAGES,' ,
|| ,lockmode,pool'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL CLOSE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT1'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
orec=orec || left(dbspace,2Ø) ,
|| left(dbspaceno,10) ,
|| left(dbspaceowner,10) ,
|| left(ntabs,10) ,
|| left(npages,20) ,
|| left(decode(,7'lockmode),10) ,
|| left(pool,10) ,
|| eob_marker

/*===================================================================*/
/* send output data                                                  */
/*===================================================================*/
,output' orec
return

/*===================================================================*/
/* SQL error message routine                                         */
/*===================================================================*/
error_message:
SAY ,ERROR'
output ,ERROR'
return

/*===================================================================*/
/* decode: translate codes to English text                           */
/*===================================================================*/
decode:
select
when arg(1) = ,1P' then ktext='Primary key'
when arg(1) = ,1F' then ktext='Foreign key'
when arg(1) = ,1U' then ktext='Unique constraint'
when arg(1) = ,2R' then ktext='Restrict'
when arg(1) = ,2C' then ktext='Cascade'
when arg(1) = ,2N' then ktext='Set Null'
when arg(1) = ,3A' then ktext='Active'
when arg(1) = ,3I' then ktext='Inactive'
when arg(1) = ,3D' then ktext='Implicitly inactive'
when arg(1) = ,4U' then ktext='Unique'
when arg(1) = ,4D' then ktext='Duplicates allowed'
when arg(1) = ,5C' then ktext='clustered'
when arg(1) = ,5N' then ktext='not clustered'
when arg(1) = ,5F' then ktext='clustered, default insert index'
when arg(1) = ,5W' then ktext='not clustered, default insert index'
when arg(1) = ,5 , then ktext='inactive primary key index'
when arg(1) = ,6P' then ktext='index for active primary key'
when arg(1) = ,6I' then ktext='index for inactive primary key'
when arg(1) = ,6U' then ktext='index for unique constraint'
when arg(1) = ,6 , then ktext=' ,
when arg(1) = ,7S' then ktext='Dbspace'
when arg(1) = ,7P' then ktext='Page'
when arg(1) = ,7T' then ktext='Row'
otherwise ktext='code' arg(1) ,unknown in decode routine'
```

```
end
return ktext

/*==================================================================*/
/* get Key Columns                                                  */
/*==================================================================*/
get_key_columns:
keycols=''
stmt='SELECT' ,
,CNAME, KEYORD' ,
,FROM SYSTEM.SYSKEYCOLS' ,
"WHERE TNAME = ,"STRIP(TABNAME)"' AND TCREATOR = ,"STRIP(CREATOR)"'",
"AND KEYNAME = ,"STRIP(KEYNAME)"'"
address command ,RXSQL PREP STAT2' stmt
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL OPEN STAT2'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
do forever
address command ,
,RXSQL FETCH STAT2 W_CNAME, W_KEYORD'
if rc =4 &  sqlcode = 1ØØ then leave
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
keycols=keycols w_cname
end
address command ,RXSQL CLOSE STAT2'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
return
end
address command ,RXSQL PURGE STAT2'
if rc ¬= Ø | sqlcode ¬= Ø
then
do
call error_message
```

```
return
end
return
```

VISUAL BASIC COMPONENTS

InfoClient1

```
'————————————————————————————————————————————————————————————————————
'
' InfoClient1.InfoClient1Macros  VBA code for starting DB2
'                                 Table List Generator
'
'
' Function description:
'
'   The macro DB2TableList simply displays UserForm1 which
'   contains all the code of the DB2 Table List Generator
'————————————————————————————————————————————————————————————————————
Sub DB2TableList()
    UserForm1.Show
End Sub

'————————————————————————————————————————————————————————————————————
'
' InfoClient1.UserForm1     VBA code for the DB2 Table List Generator
'
'
' Function description:
'
'   The user selects a DB2 database and clicks
'   CommandButton1. A connection to the InfoServer is established and
'   request "REQ1" sent. After receipt of the response (ie the
'   list of tables), each entry of the list is inserted into
'   the Excel worksheet as a separate row.
'
' This form contains a Winsock control which is used to
' communicate with the InfoServer in VM.
'
'————————————————————————————————————————————————————————————————————

'————————————————————————————————————————————————————————————————————
' Variable declarations
'————————————————————————————————————————————————————————————————————
Dim indata As String
Dim answer As String

'————————————————————————————————————————————————————————————————————
' Connect to server when button is clicked
'————————————————————————————————————————————————————————————————————
```

```
Private Sub CommandButton1_Click()
    Winsock1.RemoteHost = "es9"
    Winsock1.RemotePort = 4444
    Winsock1.Connect
End Sub

'_____________________________________________________________________
' Load DB selection listbox
'_____________________________________________________________________
Private Sub UserForm_Initialize()
    ListBox1.AddItem "SQLPROD"
    ListBox1.AddItem "SQLTEST"
    ListBox1.AddItem "SQLMILL"
End Sub

'_____________________________________________________________________
' Send request to server as soon as connection is OK.
' Hide form1 and display form2 while data is transferred.
'_____________________________________________________________________
Private Sub Winsock1_Connect()
    Winsock1.SendData "REQ1 " & ListBox1.Value
    Debug.Print ListBox1.Value
    indata = ""
    UserForm1.Hide
    UserForm2.Show
End Sub

'_____________________________________________________________________
' Receive incoming data from server
' When the string "#eod#" is found, the connection to the
' server is closed and the data inserted into the Excel worksheet.
'_____________________________________________________________________
Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
    Debug.Print "data arrival len"; bytesTotal
    Winsock1.GetData answer
    indata = indata & answer
    If InStr(answer, "##eod##") > 0 Then
        Winsock1.Close
        Call insert_data
        UserForm2.Hide  ' this ends execution of the macro
    End If
End Sub

'_____________________________________________________________________
' Insert data into Excel worksheet
'_____________________________________________________________________
Private Sub insert_data()

    If Left(indata, 5) = "ERROR" Then
        MsgBox ("InfoServer error occurred")
        Exit Sub
    End If
    ixrow = 3
    With ActiveWorkbook.ActiveSheet
    .Cells.Clear

    Columns("A:A").ColumnWidth = 30
    Columns("B:B").ColumnWidth = 9
    Columns("C:C").ColumnWidth = 65
    Cells(1, 1) = "DB2 Tables in Database " & ListBox1.Value
    Range("A1").Font.ColorIndex = 5
    Range("A1").Font.Name = "Arial"
    Range("A1").Font.Size = 16

    t = InStr(indata, "@")
    While t > 0
        tx = Left(indata, t)
        indata = Mid(indata, t + 1)
        t2 = InStr(tx, "&")
        tname = Left(tx, t2 - 1)
        tx = Mid(tx, t2 + 1)
        t2 = InStr(tx, "&")
        ttype = Left(tx, t2 - 1)
        tx = Mid(tx, t2 + 1)
        tdescr = Left(tx, Len(tx) - 1)
        Cells(ixrow, 1) = tname
        Cells(ixrow, 2) = ttype
        Cells(ixrow, 3) = tdescr
        ixrow = ixrow + 1
        t = InStr(indata, "@")
    Wend
    End With
End Sub

'————————————————————————————————————————————————————————————————————————
' Handle Socket Errors
'————————————————————————————————————————————————————————————————————————
Private Sub Winsock1_Error(ByVal Number As Integer, Description As
String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As
String, ByVal HelpContext As Long, CancelDisplay As Boolean)
    MsgBox (Description & Str(Number))
End Sub
```

InfoClient 2

```
'————————————————————————————————————————————————————————————————————————
'
' InfoClient2.InfoClient2Macros     VBA code for starting
'                                    CMS file grabber
'
'
```

```
' Function description:
'
'   The macro CMSFileGrabber simply displays UserForm1 which
'   contains all the code of the CMS file grabber
'______________________________________________________________________

Sub CMSFileGrabber()
    UserForm1.Show
End Sub

'______________________________________________________________________
'
' InfoClient2.UserForm1     VBA code for the CMS file grabber
'
'
' Function description:
'
'   The user enters VM mini-disk and file identification and clicks
'   CommandButton1. A connection to the InfoServer is established and
'   request "REQ2" sent. After receipt of the response (ie the
'   contents of the CMS file), each record of the file is inserted
'   into the Word document as a separate line.
'
' This form contains a Winsock control which is used to
' communicate with the InfoServer in VM
'
'______________________________________________________________________

'______________________________________________________________________
' Variable declarations
'______________________________________________________________________
Dim indata As String
Dim answer As String
'______________________________________________________________________
' Connect to server when button is clicked
'______________________________________________________________________
Private Sub CommandButton1_Click()
    Winsock1.RemoteHost = "es9"
    Winsock1.RemotePort = 4444
    Winsock1.Connect
End Sub

'______________________________________________________________________
' Send request to server as soon as connection is OK
' hide form1 and display form2 while data is transferred
'______________________________________________________________________
Private Sub Winsock1_Connect()
    Winsock1.SendData "REQ2 " & UserForm1.inFileSpec
    indata = ""
    UserForm1.Hide
    UserForm2.Show
```

```
End Sub

'-----------------------------------------------------------------------
' Receive incoming data from server
' When the string "#eod#" is found, the connection to the
' server is closed and the data inserted into the Word document
'-----------------------------------------------------------------------
Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
    Debug.Print "data arrival len"; bytesTotal

    Winsock1.GetData answer
    Debug.Print answer
    indata = indata & answer
    If InStr(answer, "##eod##") > 0 Then
        Winsock1.Close
        Call insert_data
        UserForm2.Hide  ' this ends execution of the macro
    End If
End Sub

'-----------------------------------------------------------------------
' Insert data into Word document
'-----------------------------------------------------------------------
Private Sub insert_data()
    While Len(indata) > 0
        lx = Val(Left(indata, 8))
        rec = Mid(indata, 9, lx)
        indata = Mid(indata, 9 + lx)
        Selection.TypeText Text:=rec
        Selection.TypeParagraph
    Wend
    Selection.WholeStory
    Selection.Font.Name = "Courier New"
    Selection.Font.Size = 10
End Sub

'-----------------------------------------------------------------------
' Handle Socket Errors
'-----------------------------------------------------------------------
Private Sub Winsock1_Error(ByVal Number As Integer, Description As
String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As
String, ByVal HelpContext As Long, CancelDisplay As Boolean)
    MsgBox (Description & Str(Number))
End Sub
```

InfoClient3

```
'-----------------------------------------------------------------------
'
' InfoClient3.InfoClient3Macros   VBA code for starting
'                                  DB2 table description generator
```

```
'
'
' Function description:
'
'   The macro DB2TableDescription simply displays UserForm1 which
'   contains all the code of the DB2 table description generator
'_____________________________________________________________________

Sub DB2TableDescription()
    UserForm1.Show
End Sub

'_____________________________________________________________________
'
' InfoClient3.UserForm1   VBA code for DB2 Table Description Generator
'
'
' Function description:
'
'   The user enters dbname, creator, and tablename and clicks
'   CommandButton1. A connection to the InfoServer is established and
'   request "REQ3" sent. After receipt of the response (ie the
'   table description data), the DB2 table description is inserted into
'   the Word document and formatted.
'
' This form contains a Winsock control which is used to
' communicate with the InfoServer in VM
'
'_____________________________________________________________________

'_____________________________________________________________________
' Variable declarations
'_____________________________________________________________________
Dim indata As String
Dim answer As String
Dim w_db, w_creator, w_table

'_____________________________________________________________________
' Connect to server when button is clicked
'_____________________________________________________________________
Private Sub CommandButton1_Click()
    Winsock1.RemoteHost = "es9"
    Winsock1.RemotePort = 4444
    Winsock1.Connect
End Sub

'_____________________________________________________________________
' Send request to server as soon as connection is OK.
' Hide form1 and display form2 while data is transferred.
'_____________________________________________________________________
Private Sub Winsock1_Connect()
```

```
    w_db = UserForm1.inDB
    w_creator = UserForm1.inCreator
    w_table = UserForm1.inTable
    Winsock1.SendData "REQ3 " & w_db & " " & w_creator & " " & w_table
    indata = ""
    UserForm1.Hide
    UserForm2.Show
End Sub

'---------------------------------------------------------------------
' Receive incoming data from server
' When the string "#eod#" is found, the connection to the
' server is closed and the data inserted into the Word document
'---------------------------------------------------------------------
Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)
    Winsock1.GetData answer
    indata = indata & answer
    If InStr(answer, "##eod##") > Ø Then
        Winsock1.Close
        Call insert_data
        UserForm2.Hide  ' this ends execution of the macro
    End If
End Sub

'---------------------------------------------------------------------
' Insert data into Word document and format it
'---------------------------------------------------------------------
Private Sub insert_data()
'-- handle errors ---
If Left(indata, 5) = "ERROR" Then
    MsgBox ("InfoServer error occurred")
    Exit Sub
End If
Dim eor_marker As String
eor_marker = "#eor#"
Dim eob_marker As String
eob_marker = "#eob#"
Dim p As Integer
'----  extract data blocks from received data stream --------
p = InStr(indata, eob_marker)
table_block = Left(indata, p - 1)
indata = Mid(indata, p + 5)
p = InStr(indata, eob_marker)
column_block = Left(indata, p - 1)
indata = Mid(indata, p + 5)

p = InStr(indata, eob_marker)
key_block = Left(indata, p - 1)
indata = Mid(indata, p + 5)

p = InStr(indata, eob_marker)
index_block = Left(indata, p - 1)
```

```
indata = Mid(indata, p + 5)

p = InStr(indata, eob_marker)
dbspace_block = Left(indata, p - 1)
indata = Mid(indata, p + 5)

Dim rr As Range
Set para = Selection.Paragraphs(1)
Set rr = Selection.Paragraphs(1).Range

' -------- general information --------------------------------------
Selection.TypeText Text:="DB2 Table " & w_db & "." & w_creator & "." _
                          & w_table & "     "
Selection.InsertDateTime DateTimeFormat:="t. MMMM jjjj",
InsertAsField:=False

Selection.TypeParagraph
Selection.TypeText Text:=vbVerticalTab & Trim(table_block) &
vbVerticalTab
Selection.TypeParagraph

' ------ column information -----------------------------------------
Selection.TypeText Text:="Column Information"
Selection.TypeParagraph
Selection.TypeParagraph

Selection.TypeText Text:=vbTab & "Colno" & vbTab & "Column Name"  _
    & vbTab & "Type" & vbTab & "Length" & vbTab & "Nulls" & vbTab _
    & "Fieldproc" & vbTab & "Label/Remarks" & vbVerticalTab _
 & vbVerticalTab
While Len(column_block) > 0
    s = 1
    l = 5
    colno = Trim(Mid(column_block, s, l))
    s = s + l
    l = 30
    cname = Trim(Mid(column_block, s, l))
    s = s + l
    l = 12
    coltype = Trim(Mid(column_block, s, l))
    s = s + l
    l = 12
    Length = Trim(Mid(column_block, s, l))
    s = s + l
    l = 1
    nulls = Trim(Mid(column_block, s, l))
    s = s + l
    l = 8
    fldproc = Trim(Mid(column_block, s, l))
    s = s + l
    l = 40
    Label = Trim(Mid(column_block, s, 1))

    column_block = Mid(column_block, 109)
    Selection.TypeText Text:=vbTab & colno & vbTab & cname _
        & vbTab & coltype & vbTab & Length & vbTab _
        & nulls & vbTab & fldproc & vbTab & Label & vbVerticalTab
Wend

Selection.TypeParagraph
' ────── key information ──────────────────────────────────────
Selection.TypeText Text:="Key Information"
Selection.TypeParagraph
Selection.TypeParagraph

While Len(key_block) > 0
    p1 = InStr(key_block, eor_marker)
    key_entry = Left(key_block, p1 - 1)
    key_block = Mid(key_block, p1 + 5)
    While Len(key_entry) > 0
        w_Name = Left(key_entry, 20)
        w_Value = Mid(key_entry, 21, 60)
        key_entry = Mid(key_entry, 81)
        Selection.TypeText Text:=Trim(w_Name) & vbTab & ":" & vbTab _
                & Trim(w_Value) & vbVerticalTab
    Wend
    Selection.TypeText vbVerticalTab
Wend
Selection.TypeParagraph

' ────── index information ──────────────────────────────────────
Selection.TypeText Text:="Index Information"
Selection.TypeParagraph
Selection.TypeParagraph

While Len(index_block) > 0
    p1 = InStr(index_block, eor_marker)
    index_entry = Left(index_block, p1 - 1)
    index_block = Mid(index_block, p1 + 5)
    Selection.TypeText Text:="Index Name" & vbTab & _
          ":" & vbTab & Trim(Mid(index_entry, 1, 20)) & vbVerticalTab
    Selection.TypeText Text:="Columns" & vbTab & _
         ":" & vbTab & Trim(Mid(index_entry, 21, 70)) & vbVerticalTab
    Selection.TypeText Text:="Index Type" & vbTab & _
         ":" & vbTab & Trim(Mid(index_entry, 91, 40)) & vbVerticalTab
    Selection.TypeText Text:="Cluster Status" & vbTab & _
        ":" & vbTab & Trim(Mid(index_entry, 131, 40)) & vbVerticalTab
    Selection.TypeText Text:="Cluster Ratio" & vbTab & _
        ":" & vbTab & Trim(Mid(index_entry, 171, 40)) & vbVerticalTab
    Selection.TypeText Text:="Key Type" & vbTab & _
        ":" & vbTab & Trim(Mid(index_entry, 211, 40)) & vbVerticalTab
    Selection.TypeText vbVerticalTab
Wend
```

```
Selection.TypeParagraph
' ———— Dbspace information ————————————————————————
Selection.TypeText Text:="Dbspace Information"
Selection.TypeParagraph
Selection.TypeParagraph

    Selection.TypeText Text:="Dbspace Name" & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 1, 2Ø)) & vbVerticalTab
    Selection.TypeText Text:="Dbspace No." & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 21, 1Ø)) & vbVerticalTab
    Selection.TypeText Text:="Owner" & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 31, 1Ø)) & vbVerticalTab
    Selection.TypeText Text:="Tables" & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 41, 1Ø)) & vbVerticalTab
    Selection.TypeText Text:="Pages" & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 51, 2Ø)) & vbVerticalTab
    Selection.TypeText Text:="Lockmode" & vbTab & _
       ":" & vbTab & Trim(Mid(dbspace_block, 71, 1Ø)) & vbVerticalTab
    Selection.TypeText Text:="Storage Pool" & vbTab & _
        ":" & vbTab & Trim(Mid(dbspace_block, 81, 1Ø)) & vbVerticalTab
    Selection.TypeText vbVerticalTab

Selection.TypeParagraph

' ————————— do some formatting ————————————————————————

With rr.Paragraphs(1).Range.ParagraphFormat
    .Alignment = wdAlignParagraphCenter
        With .Shading
            .Texture = wdTextureNone
            .ForegroundPatternColorIndex = wdWhite
            .BackgroundPatternColorIndex = wdBlue
        End With
        With .Borders(wdBorderLeft)
            .LineStyle = wdLineStyleSingle
            .LineWidth = wdLineWidthØ5Øpt
            .ColorIndex = wdAuto
        End With
        With .Borders(wdBorderRight)
            .LineStyle = wdLineStyleSingle
            .LineWidth = wdLineWidthØ5Øpt
            .ColorIndex = wdAuto
        End With
        With .Borders(wdBorderTop)
            .LineStyle = wdLineStyleSingle
            .LineWidth = wdLineWidthØ5Øpt
            .ColorIndex = wdAuto
        End With
        With .Borders(wdBorderBottom)
            .LineStyle = wdLineStyleSingle
```

```
                .LineWidth = wdLineWidth050pt
                .ColorIndex = wdAuto
            End With
            With .Borders
                .DistanceFromTop = 1
                .DistanceFromLeft = 4
                .DistanceFromBottom = 1
                .DistanceFromRight = 4
                .Shadow = True
            End With
        With Options
            .DefaultBorderLineStyle = wdLineStyleSingle
            .DefaultBorderLineWidth = wdLineWidth050pt
            .DefaultBorderColorIndex = wdAuto
        End With
    End With
    With rr.Paragraphs(1).Range.Font
            .Name = "Courier New"
            .Size = 12
            .Bold = True
            .ColorIndex = wdWhite
    End With

    With rr.Paragraphs(3).Range.ParagraphFormat
        .Alignment = wdAlignParagraphCenter
        .Shading.Texture = wdTexture12Pt5Percent
        .Shading.ForegroundPatternColorIndex = wdAuto
        .Shading.BackgroundPatternColorIndex = wdWhite
    End With

    With rr.Paragraphs(5).Range.ParagraphFormat.TabStops
        .Add Position:=CentimetersToPoints(1), _
            Alignment:=wdAlignTabRight, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(1.5) _
            , Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(5), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(9), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(11), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(12), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(14), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
    End With

    With rr.Paragraphs(6).Range.ParagraphFormat
        .Alignment = wdAlignParagraphCenter
        .Shading.Texture = wdTexture12Pt5Percent
        .Shading.ForegroundPatternColorIndex = wdAuto
        .Shading.BackgroundPatternColorIndex = wdWhite
    End With

    With rr.Paragraphs(8).Range.ParagraphFormat.TabStops
        .ClearAll
        .Add Position:=CentimetersToPoints(4) _
            , Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(5), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
    End With

    With rr.Paragraphs(9).Range.ParagraphFormat
        .Alignment = wdAlignParagraphCenter
        .Shading.Texture = wdTexture12Pt5Percent
        .Shading.ForegroundPatternColorIndex = wdAuto
        .Shading.BackgroundPatternColorIndex = wdWhite
    End With

    With rr.Paragraphs(11).Range.ParagraphFormat.TabStops
        .ClearAll
        .Add Position:=CentimetersToPoints(4) _
            , Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(5), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
    End With

    With rr.Paragraphs(12).Range.ParagraphFormat
        .Alignment = wdAlignParagraphCenter
        .Shading.Texture = wdTexture12Pt5Percent
        .Shading.ForegroundPatternColorIndex = wdAuto
        .Shading.BackgroundPatternColorIndex = wdWhite
    End With
    With rr.Paragraphs(14).Range.ParagraphFormat.TabStops
        .ClearAll
        .Add Position:=CentimetersToPoints(4) _
            , Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
        .Add Position:=CentimetersToPoints(5), _
            Alignment:=wdAlignTabLeft, Leader:=wdTabLeaderSpaces
    End With
    End Sub
    '______________________________________________________________________
    ' Handle Socket Errors
    '______________________________________________________________________
    Private Sub Winsock1_Error(ByVal Number As Integer, Description As
    String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As
    String, ByVal HelpContext As Long, CancelDisplay As Boolean)
        MsgBox (Description & Str(Number))
    End Sub
```

© Xephon 1999

A full screen console interface – part 15

Editor's note: the following article is an extensive piece of work which will be published over several issues of VM Update. *It was felt that readers could benefit from the entire article and from the individual sections. Any comments or recommendations would be welcomed and should be addressed either to Xephon or directly to the author at fernando_duarte@vnet.ibm.com.*

```
*
* Release user buffers and UID block
*
*        Output R2 addresses previous UID block or SSSPTR
*               R8 contains zero
*
CSCUSARL RELOC                          Release block (external call)
         BAS   R14,RELEASE              Perform function
         BACK                           Return to caller
         SPACE
RELEASE  EQU   *                        Release user storage
         ST    R14,RELSV14
         LR    RØ,R8                    Save UID address
         LA    R8,SSSPTR                Scan active sessions
REL1ØØ   LR    R2,R8                    Keep address of previous entry
         L     R8,UIDFWD                Address next entry
         CR    RØ,R8
         BNE   REL1ØØ
         L     R1,UIDFWD                Found
         ST    R1,Ø(,R2)                Alter chain pointer
         LR    R8,RØ                    Restore UID entry address
         LA    RØ,UIDSCRSZ              Release screen
         L     R1,UIDSCRN
         LINK  RELEASE
         LA    RØ,UIDBUFSZ
         L     R1,UIDBUFF               Release user buffer
         LINK  RELEASE
         TM    UIDOPT1,UIDCONN          Is user connected?
         BZ    REL2ØØ
         TM    UIDOPT3,UIDCREQ          Really connected?
         BO    REL2ØØ                   No, just waiting to connect
         LA    RØ,UIDCSCSZ              Yes, release work CSCBUFF
         L     R1,UIDCSC
         LINK  RELEASE
         TM    UIDOPT1,UIDRMTE          Is user also remote?
         BO    REL2ØØ
         LA    RØ,UIDSCRSZ              No, release alternative screen
         L     R1,UIDSCRNA
         LINK  RELEASE
```

```
REL2ØØ   LA    RØ,UIDSIZE             De-allocate UID block
         LR    R1,R8
         LINK  RELEASE
         SR    R8,R8                  Clear UID pointer
         L     R14,RELSV14
         BR    R14
         SPACE 3
         DS    ØD
APPC$SS  DC    C'<CSC>$SS'            Start Session
APPC$SC  DC    C'<CSC>$SC'            Session created
APPC$SD  DC    C'<CSC>$SD'            Session data to connect node
APPC$SE  DC    C'<CSC>$SE'            End Connected Session
APPC$SR  DC    C'<CSC>$SR'            Session rejected
APPC$SU  DC    C'<CSC>$SU'            Session data to user
APPC$TC  DC    C'<CSC>$TC'            Terminate connected sessions
COMMCNN  DC    C'<CSC>CNN'            Display Connect Node
         SPACE
USASVØ5  DS    F                      Save R5  - CSCUSA
DSESVØ5  DS    F                                 DSESSION
CLNSVØ5  DS    F                                 CSCUSACL
SNDSVØ5  DS    F                                 SEND
ESESV14  DS    F                      Save R14 - ESESSION
SNDSV14  DS    F                                 SEND
DSPSV14  DS    F                                 DISPLAY
RELSV14  DS    F                                 RELEASE
         SPACE 3
         CSCDATA
         CSCDS (UID,RND,APP,USR)
         REGEQU
         END
```

CSCOPQ ASSEMBLE

This module adds support for the operator QUERY command. It is used mainly for debugging purposes.

```
         TITLE 'CSCOPQ - CSC Process Operator Query Command'
CSCOPQ   START X'Ø1DD2Ø'
         PRINT NOGEN
         CSCHDR                       Process Operator Query Command
*
* Process Query command
*
*
         LA    RØ,OPQTABLE
         GO    CSCSCN                 Scan subcommand
         BNZ   OPQ1ØØ                 Nothing, display error message
         LTR   R15,R15                Is it valid?
         BZ    OPQ2ØØ                 No, another error message
         GO                         , Yes, process subcommand
         B     OPQ9ØØ
         SPACE
OPQ1ØØ   MSG   Ø6Ø6,CC                 Missing subcommand
         B     OPQ9ØØ
         SPACE
OPQ2ØØ   MSG   Ø6Ø4,CC                 Invalid subcommand
         B     OPQ9ØØ
         SPACE
OPQ9ØØ   BACK                          All done, go back
         SPACE 3
*
*   Query Links
*
LINKS    EQU   *                 LINKS subcommand
         USING RNDSECT,R5              RND Table
         ST    R14,CMDSV14
         SR    RØ,RØ
         GO    CSCSCN                  No table to search
         BZ    LINK5ØØ                 Something found, bad news
         TM    CSCFLGØ1,CSCAPPC        Is APPC/VM enabled?
         BZ    LINK6ØØ                 No, command not valid
         MSG   Ø643
         LA    R5,RNDPTR               Address RND table
LINK1ØØ  L     R5,RNDFWD               Scan table
         TM    RNDOPT1,RNDOLCL         Locate Local node
         BZ    LINK1ØØ                 Not this one
         LA    R2,RNDNODE              Address node name for messages
         LA    R3,RNDRSRC              Address resource name
         LA    R4,ACTIVE               Assume link is active
         TM    RNDOPT1,RNDOSND         Is it really active?
         BO    LINK11Ø                 Yes, good guess
         LA    R4,INACTIVE             Try inactive
         BZ    LINK11Ø                 Not bad
         LA    R4,PENDING              Could not be worse
LINK11Ø  LA    R5,LOCAL                Assume this is a local resource
         TM    RNDOPT1,RNDOLRS         Is it really local?
         BZ    LINK12Ø                 Another guessing game
         LA    R5,GLOBAL               Not local, it must be global
LINK12Ø  MSG   Ø644,SPACE
         LA    R5,RNDPTR               Address RND table
LINK2ØØ  L     R5,RNDFWD               Scan table
         LTR   R5,R5                   Check for end of table
         BZ    LINK3ØØ                 Node not found
         TM    RNDOPT1,RNDOLCL         Locate Local node
         BO    LINK2ØØ                 Not this one
         TM    RNDOPT1,RNDOTMP         Temporary RND entry?
         BO    LINK2ØØ                 Yes, ignore it
         LA    R2,RNDNODE              Address node name for messages
         LA    R3,RNDRSRC              Address resource name
         LA    R4,ACTIVE               Assume link is active
         TM    RNDOPT1,RNDOSND+RNDORCV Is it really active?
         BO    LINK21Ø                 Yes, good guess
         LA    R4,INACTIVE             Try inactive
         BZ    LINK21Ø                  Not bad
         LA    R4,PENDING               Could not be worse
LINK21Ø  MSG   Ø645,SPACE
         B     LINK2ØØ
         SPACE
LINK3ØØ  SR    R15,R15                  Set return code
         B     LINK9ØØ
         SPACE
LINK5ØØ  MSG   Ø6Ø5,CC                  Unexpected operand
         B     LINK9ØØ
         SPACE
LINK6ØØ  MSG   Ø642,CC                  APPC/VM not enabled
         B     LINK9ØØ
         SPACE
LINK9ØØ  L     R14,CMDSV14
         BR    R14
         SPACE
         DROP  R5
         SPACE 3
*
*   Query SEssions
*
*
SESSIONS EQU   *               SESSIONS subcommand
         USING UIDSECT,R8               UID (user) Block
         ST    R14,CMDSV14
         SR    R4,R4                    Initialize counter
         LA    R8,UIDPTR                Address Pending sessions table
SESS1ØØ  L     R8,UIDFWD                Address entry
         LTR   R8,R8                    End of list?
         BZ    SESS5ØØ                  Yes, check active sessions
         LA    R4,1(,R4)                Increment counter
         LA    R2,UIDVMID               Address user-id
         LA    R3,UIDORIG               Address originating APPC node
         CLC   UIDORIG,CSCLOCAL         Is it a local user?
         BNE   SESS11Ø
         LA    R3,BLANKS                Yes, do not display node name
SESS11Ø  LA    R4,PENDING
         MSG   Ø646,SPACE
         B     SESS1ØØ                  List all sessions
         SPACE
SESS5ØØ  LA    R8,SSSPTR                Address Active sessions table
SESS6ØØ  L     R8,UIDFWD                Address entry
         LTR   R8,R8
         BZ    SESS8ØØ                  End of list, all done
         LA    R4,1(,R4)
         LA    R2,UIDVMID               Address user-id
         LA    R3,UIDORIG               Address originating APPC/VM node
         CLC   UIDORIG,CSCLOCAL         Is it a local user?
         BNE   SESS61Ø
         LA    R3,BLANKS                Yes, do not display node name
SESS61Ø  LA    R4,ACTIVE
```

```
         MSG   Ø646,SPACE
         B     SESS6ØØ                  List all sessions
         SPACE
SESS8ØØ  LTR   R4,R4
         BNZ   SESS89Ø
         MSG   Ø647
SESS89Ø  SR    R15,R15                  Zero return code
SESS9ØØ  L     R14,CMDSV14
         BR    R14
         SPACE
         DROP  R8
         SPACE 3
*
*   Query Storage
*
STORAGE  EQU   *                STORAGE subcommand
         ST    R14,CMDSV14
         SR    RØ,RØ
         GO    CSCSCN                   No table to search
         BNZ   STOR1ØØ                  Nothing, that's good news
         MSG   Ø6Ø5,CC                  Display error message
         B     STOR9ØØ
         SPACE
STOR1ØØ  L     R2,FSALLDW               Double words allocated
         SLL   R2,3                     Convert to bytes
         L     R3,FSALL                 Total allocations
         MSG   Ø64Ø
         SRL   R2,3                     Convert bytes back to dwords
         S     R2,FSRELDW               Subtract from dwords released
         S     R2,FSINIDW               Dwords used for initialization
         SLL   R2,3                     Convert to bytes
         S     R3,FSREL                 Same thing for total allocations
         S     R3,FSINI
         MSG   Ø641
         SR    R15,R15                  Zero return code
STOR9ØØ  L     R14,CMDSV14
         BR    R14
         SPACE 3
*
*   Query Users
*
USERS    EQU   *                  USERS subcommand
         ST    R14,CMDSV14
         SR    R15,R15                  Zero return code
 MSG 3333
USER9ØØ  L     R14,CMDSV14
         BR    R14
         SPACE 3
CMDSV14  DS    F                        Save area for input commands
         DS    ØD
ACTIVE   DC    C'Active  '
INACTIVE DC    C'Inactive'
```

```
PENDING  DC    C'Pending '
LOCAL    DC    C'Local   '
GLOBAL   DC    C'Global  '
         SPACE
OPQTABLE CMMD  (I,ØØ,Ø1,LINKS,LINKS),  Query subcommands             *
               (I,ØØ,Ø2,SESSIONS,SESSIONS),                          *
               (I,ØØ,Ø1,STORAGE,STORAGE),                            *
               (I,ØØ,Ø1,USERS,USERS)
*
* OPTIONS PREFIXES
*
         SPACE
LNKTABLE CMMD  (B,ØØ,Ø1,ALL,LNKALL),   LINK operands                 *
               (B,ØØ,Ø1,ACTIVE,LNKACT),                              *
               (B,ØØ,Ø1,INACTIVE,LNKINACT)
         SPACE
LNKALL   EQU   LNKACT+LNKINACT
LNKACT   EQU   X'8Ø'
LNKINACT EQU   X'4Ø'
         SPACE
         CSCDATA
         CSCDS (RND,UID)
         REGEQU
         END
```

CSCOPA ASSEMBLE

This module adds support for the operator START and STOP commands. Note that this code has not been fully tested.

```
         TITLE 'CSCOPA - CSC Remote Node Operator Commands'
CSCOPA   START X'Ø1D118'
         PRINT NOGEN
         CSCHDR                        APPC/VM Operator commands
*
* Process APPC/VM Operator Commands
*
*        USING IPARML,R9               IUCV Parameter List
*        USING UIDSECT,R8              UID (user) Block
*        USING CCHSECT,R7              CCH (cache) Block
         USING RNDSECT,R5              RND Table
         SPACE
*
* Start an inactive link
*
*    STArt nodeid
*          *
*
         TM    CSCFLGØ1,CSCAPPC        Is APPC/VM enabled?
         BZ    STRT6ØØ                 No, command not valid
         LA    R5,RNDPTR               Address RND table
```

```
STRT100  L      R5,RNDFWD                Scan table
         TM     RNDOPT1,RNDOLCL          Locate Local node
         BZ     STRT100                  Not this one
         LA     R2,RNDNODE               Address node name for messages
         TM     RNDOPT1,RNDOSND          Is Local node active
         BZ     STRT620                  No, display error message
         SR     R0,R0                    No table to search
         GO     CSCSCN                   Get node name
         BNZ    STRT700                  Nothing found, display error
         LA     R0,8                     Maximum length is 8
         CR     R0,R1                    Check against entered data
         BL     STRT720                  Too long... too bad
         SR     R0,R0                    No table to search
         GO     CSCSCN                   Check for extra operands
         BZ     STRT740                  Too bad, it is invalid
         LA     R6,SCANUPP               Address entered data for message
         CLC    SSALL,SCANUPP            Is it STArt *
         BE     STRT300                  Yes, that's a different game
         L      R1,RNDPTR                Address RND table
STRT200  LTR    R5,R1                    Check for end of table
         BZ     STRT760                  Node not found
         L      R1,RNDFWD                Address following entry
         LA     R2,RNDNODE               Address node name for messages
         CLC    RNDNODE,SCANUPP          Compare node names
         BNE    STRT200                  Not this one
         TM     RNDOPT1,RNDOLCL          Is it the Local node name
         BO     STRT800                  Yes, command not valid for Local
         TM     RNDOPT1,RNDOSND+RNDORCV  Is link already up?
         BO     STRT820                  Yes, cannot start again
         L      R0,RNDPIDS               Load possible IUCV PATHID
         LTR    R0,R0                    Is it zero
         BNZ    STRT840                  No, activation is in progress
         GO     CSCRNCST                 Activate link
         B      STRT900                  All done
         SPACE
STRT300  SR     R0,R0                    Process STArt *
         ST     R0,SSCNT                 Zero counter
         L      R1,RNDPTR                Address RND table
STRT400  LTR    R5,R1                    Check for end of table
         BZ     STRT500                  Found it, all done
         L      R1,RNDFWD                Address following entry
         TM     RNDOPT1,RNDOLCL          Is it the Local node?
         BO     STRT400                  Yes, skip it
         TM     RNDOPT1,RNDOSND+RNDORCV  Is link already active?
         BO     STRT400                  Yes, skip it
         L      R0,RNDPIDS               Is activation in progress?
         LTR    R0,R0
         BNZ    STRT400                  Yes, skip it
         GO     CSCRNCST                 Start link
         LA     R0,1                     Increment counter by one
         A      R0,SSCNT
         ST     R0,SSCNT
         L      R1,RNDFWD                 Address following entry
         B      STRT4ØØ                   Scan all RND table
         SPACE
STRT5ØØ  L      R2,SSCNT                  Load counter
         LTR    R2,R2                     Any link started?
         BZ     STRT51Ø
         MSG    Ø91Ø                      Yes, display info message
         B      STRT9ØØ
         SPACE
STRT51Ø  MSG    Ø911                      No, display a different message
         B      STRT9ØØ
         SPACE
STRT6ØØ  MSG    Ø9ØØ                      APPC/VM not enabled
         B      STRT9ØØ
         SPACE
         SPACE
STRT62Ø  MSG    Ø9Ø1                      Local node not active
         B      STRT9ØØ
         SPACE
STRT7ØØ  MSG    Ø9Ø2                      Missing operand
         B      STRT9ØØ
         SPACE
STRT72Ø  MSG    Ø9Ø3                      Operand too long
         B      STRT9ØØ
         SPACE
STRT74Ø  MSG    Ø9Ø4                      Unexpected operand
         B      STRT9ØØ
         SPACE
STRT76Ø  MSG    Ø9Ø5                      Node not defined
         B      STRT9ØØ
         SPACE
STRT8ØØ  MSG    Ø9Ø6                      Command not valid for Local node
         B      STRT9ØØ
         SPACE
STRT82Ø  MSG    Ø9Ø7                      Link already active
         B      STRT9ØØ
         SPACE
STRT84Ø  MSG    Ø9Ø8                      Activation pending
*        B      STRT9ØØ
         SPACE
STRT9ØØ  BACK
         SPACE 3
*
* Stop an active link
*
CSCOPASP RELOC                            Stop a link
         TM     CSCFLGØ1,CSCAPPC          Is APPC/VM enabled?
         BZ     STOP6ØØ                   No, command not valid
         LA     R5,RNDPTR                 Address RND table
STOP1ØØ  L      R5,RNDFWD                 Scan table
         TM     RNDOPT1,RNDOLCL           Locate Local node
         BZ     STOP1ØØ                   Not this one
         LA    R2,RNDNODE             Address node name for messages
         TM    RNDOPT1,RNDOSND        Is Local node active
         BZ    STOP62Ø                No, display error message
         SR    RØ,RØ                  No table to search
         GO    CSCSCN                 Get node name
         BNZ   STOP7ØØ                Nothing found, display error
         LA    RØ,8                   Maximum length is 8
         CR    RØ,R1                  Check against entered data
         BL    STOP72Ø                Too long... too bad
         SR    RØ,RØ                  No table to search
         GO    CSCSCN                 Check for extra operands
         BZ    STOP74Ø                Too bad, it is invalid
         LA    R6,SCANUPP             Address entered data for message
         CLC   SSALL,SCANUPP          Is it STOP *
         BE    STOP3ØØ                Yes, that's a different game
         L     R1,RNDPTR              Address RND table
STOP2ØØ  LTR   R5,R1                  Check for end of table
         BZ    STOP76Ø                Node not found
         L     R1,RNDFWD              Address following entry
         LA    R2,RNDNODE             Address node name for messages
         CLC   RNDNODE,SCANUPP        Compare node names
         BNE   STOP2ØØ                Not this one
         TM    RNDOPT1,RNDOLCL        Is it the Local node name
         BO    STOP8ØØ                Yes, command not valid for Local
         TM    RNDOPT1,RNDOSND+RNDORCV Is link up?
         BO    STOP21Ø                Yes, stop it
         L     RØ,RNDPIDS             Is activation pending?
         LTR   RØ,RØ
         BZ    STOP82Ø                No, already down
         MSG   Ø9Ø8                   Display warning message
STOP21Ø  GO    CSCRNCSP               Stop link
         B     STOP9ØØ
         SPACE
STOP3ØØ  SR    RØ,RØ                  Process STOP *
         ST    RØ,SSCNT               Zero counter
         L     R1,RNDPTR              Address RND table
STOP4ØØ  LTR   R5,R1                  Check for end of table
         BZ    STOP5ØØ                Found it, all done
         L     R1,RNDFWD              Address following entry
         TM    RNDOPT1,RNDOLCL        Is it the Local node?
         BO    STOP4ØØ                Yes, skip it
         TM    RNDOPT1,RNDOSND+RNDORCV Is link up?
         BO    STOP41Ø                Yes, stop it
         L     RØ,RNDPIDS             Is activation in progress
         LTR   RØ,RØ
         BZ    STOP4ØØ                No, skip it
         MSG   Ø9Ø8                   Display warning message
STOP41Ø  GO    CSCRNCSP               Stop link
         LA    RØ,1                   Increment counter by one
         A     RØ,SSCNT
         ST    RØ,SSCNT
         L     R1,RNDFWD              Address following entry
         B     STOP4ØØ                 Scan all RND table
         SPACE
STOP5ØØ  L     R2,SSCNT                Load counter
         LTR   R2,R2                   Any link stopped?
         BZ    STOP51Ø
         MSG   Ø916                    Yes, display info message
         B     STOP9ØØ
STOP51Ø  MSG   Ø917                    No, display a different message
         B     STOP9ØØ
         SPACE
STOP6ØØ  MSG   Ø9ØØ                    APPC/VM not enabled
         B     STOP9ØØ
         SPACE
STOP62Ø  MSG   Ø9Ø1                    Local node not active
         B     STOP9ØØ
         SPACE
STOP7ØØ  MSG   Ø9Ø2                    Missing operand
         B     STOP9ØØ
         SPACE
STOP72Ø  MSG   Ø9Ø3                    Operand too long
         B     STOP9ØØ
         SPACE
STOP74Ø  MSG   Ø9Ø4                    Unexpected operand
         B     STRT9ØØ
         SPACE
STOP76Ø  MSG   Ø9Ø5                    Node not defined
         B     STOP9ØØ
         SPACE
STOP8ØØ  MSG   Ø9Ø6                    Command not valid for Local node
         B     STOP9ØØ
         SPACE
STOP82Ø  MSG   Ø9Ø9                    Link not active
         B     STOP9ØØ
         SPACE
STOP9ØØ  BACK
         SPACE 3
         DS    ØD
SSALL    DC    C'*        '            Generic operand (all)
SSCNT    DS    F                       Counter
         CSCDATA
         CSCDS (RND)
         PUSH  PRINT
         PRINT OFF
*        COPY  IPARML
         POP   PRINT
         REGEQU
         END
```

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

Fernando Duarte
Analyst (Canada)

VM news

IBM has announced the IBM Java Port for VM/ESA, Developer Release 1.1.4, as a port for Sun Microsystems' Java Development Kit (JDK) to the System/390 platform.

Developer Release 1.1.4, developed to run with the OpenEdition Shell and Utilities feature, has executed and passed the majority of the Java Compatible test suite with the exception of certain graphical function tests for the AWT. The full range of Java graphical functionality is not supported by the VM/ESA operating system so this Java implementation does not claim to be fully Java Compatible. It has been made available as a Developer Release rather than as a GA product to facilitate development of Java applications designed for server environments that do not provide a graphical capability (also known as 'headless servers'). A subsequent GA release will support the running of Java applications that assume a screen is available.

Developer Release 1.1.4 includes a compiler, a debugger, the Java Virtual Machine (a Java byte-code interpreter), Sun Microsystem's JDK 1.1.4 class libraries, and Java Native Interface (JNI). It does not include a Just-In-Time compiler (JIT) or use of the Abstract Windowing Toolkit (AWT) classes for execution on VM/ESA.

An alternative execution environment, also available as a Developer Release, removes the need for users to purchase the OpenEdition Shell and Utilities feature by offering a 'shell-less' Java implementation.

Code changes necessary to fix the '29 February 2000' date format bug found in JDK Version 1.1.4 and 1.1.5 have been implemented so that the Developer Release 1.1.4 is Year 2000 ready.

For further information contact your local IBM representative.

* * *

Mirasoft has announced Distributed Devices for VM/ESA, enabling devices on one VM/ESA system to be used by virtual machines on another. Using the ISFC (Inter-System Facility for Communications) feature of the VM/ESA Control Program, it allows virtual machine-based servers such as VTAM and TCP/IP to support users across VM/ESA system boundaries. This allows hardware and software costs to be reduced, as well as easing system management in the multiple-system environment.

For further information contact:
Mirasoft, 60 Alban Street, Boston, MA 02124-3709, USA.
Tel: (617) 825 9121.
Jemasys, 37 Ridgeway, Wargrave, Berkshire, RG10 8AS, UK.
Tel: (01189) 404878.
URL: http://www.mirasoft.com.