December 2002

In this issue

3  Implementing an SNA MCS console
10  How to talk 3270
23  Dynamically creating a NERD chart
55  Command line ftp
62  Setting up a corporate portal – getting it right
72  Information point – reviews
76  TCP/SNA news

© Xephon plc 2002
Subscriptions and back-issues
A year’s subscription to TCP/SNA Update, comprising four quarterly issues, costs £130.00 in the UK; $190.00 in the USA and Canada; £136.00 in Europe; £142.00 in Australasia and Japan; and £140.50 elsewhere. In all cases the price includes postage. Individual issues, starting with the March 1999 issue, are available separately to subscribers for £33.00 ($49.50) each including postage.

Editorial panel
Articles published in TCP/SNA Update are reviewed by our panel of experts. Members include John Bradley (UK), Carlson Colomb (Canada), Anura Gurugé (USA), Jon Pearkins (Canada), and Tod Yampel (USA).

Contributions
When Xephon is given copyright, articles published in TCP/SNA Update are paid for at £170 ($260) per 1000 words for original material. To find out more about contributing an article, please download a copy of our Notes for Contributors from http://www.xephon.com/index/nfc

TCP/SNA Update on-line
Code from TCP/SNA Update, and complete issues in Acrobat PDF format, can be downloaded from http://www.xephon.com/tcpsna; you will need to supply a word from the printed issue..
Implementing an SNA MCS console

z/OS 1.1 supports a new type of console, SNA Multiple Console Support, a VTAM application which can use VTAM-controlled devices for MCS console support. This eliminates the need for a non-SNA 3174 Terminal Control Unit, and also means that you can easily implement a remote MCS console, allowing a distant data centre to interact with your local MVS systems.

SMCS consoles are MCS consoles that use VTAM services for input and output. SMCS consoles provide most of the same functions as MCS consoles, with the following exceptions:

- SMCS consoles are not available during NIP. The system console or an MCS console must be used instead.
- VTAM must be active for SMCS to be active. The system console and MCS consoles do not rely on VTAM, and can be used before VTAM is active.
- The activation process depends on the console definitions, but VARY CONSOLE and VARY CN, ONLINE don’t work for SMCS.

Because an SMCS console is connected through a network and uses VTAM services, network problems and the VTAM VARY NET and HALT NET commands can affect console operations.

Although an SMCS console can be a real 3270 type device, it’s usually a 3270 emulator such as IBM Personal Communications. SMCS supports VTAM LU Type 0 or Type 2, and SMCS consoles must support Extended Data Stream and the Read Partition Query function.

INSTALLING SMCS SUPPORT

Installing SMCS consoles requires some VTAM and SYS1.PARMLIB definitions. SMCS is implemented using a client/server architecture: on the MVS image, an SMCS
application server connected to VTAM interacts with the CONSOLE address space. On the client side, the 3270 SNA terminal logs on to the SMCS application to receive the console data.

VTAM definitions

Defining the VTAM major node for the SMCS application

To define the SMCS application to VTAM, you must create a VTAMLST member that defines the SMCS application id (APPLID). You could write the SMCS application definition as follows:

```
SMCS    VBUILD TYPE=APPL
SMCS&SYSNAME. APPL
```

Note that each system within the sysplex that will run an SMCS application must have a unique VTAM resource name.

Defining the VTAM major node for SMCS permanent consoles

If certain devices are always used for SMCS, they can be defined to automatically log on to the SMCS application when the device becomes active using the LOGAPPL keyword on the LOCAL or LU statements:

```
LO20885  LOCAL CUADDR=885,                      CU ADDRESS                      X
       ISTATUS=ACTIVE,           INITIAL ACTIVE                  X
       TERM=3277,                3270 DISPLAY TERMINAL           X
       FEATUR2=MODEL2,           DEFAULT SCREEN SIZE             X
       MODETAB=MTTABLE,                               X
       DLOGMOD=M2BSCQ,                              X
       LOGAPPL=SMCS&SYSNAME.,      <- automatic logon          X
       USSTAB=USSTAB00
```

SYS1.PARMLIB definitions

CONSOL00 definitions

To indicate that the SMCS application is to be started, you must define the SMCS APPLID on the INIT statement of CONSOL00:
In order to define an SMCS console, you must specify DEVNUM (SMCS). You also have to specify a NAME for this console.

CONSOLE DEVNUM(SMCS) ROUTCODE(ALL) <- SNA MCS console
NAME(SMCSS00) <- Name of the console
MSCOPE(*ALL)
RBUF(15) PFKTAB(PFKTAS00)
AUTH(ALL)
MONITOR(JOBNAMES-T)
CON(N) SEG(16) DEL(R) RNUM(19) RTME(1) MFORM(S,J,T) AREA(NONE)

Note that if you omit APPLID, SMCS will not be available for the life of the system. You can change the APPLID once the system is active, but only if an APPLID was specified in CONSOL00 during IPL. The following command can be used to change a system's SMCS APPLID:

K M,APPLID=SMCS012
IEE821E SMCS APPLID VALUE HAS BEEN CHANGED ON S012 - SMCS MUST BE RECYCLED
IEE712I CONTROL PROCESSING COMPLETE

SMCS will continue to use the old APPLID until it's deactivated with the VARY NET,INACT command. Once the old APPLID is deactivated, the new one may need to be activated using the V NET,ACT command. During the time that the old APPLID is still in use, message IEE821E will be issued as a reminder that SMCS needs to be recycled on that system.

You can issue D C,SMCS to verify your actions and display the status of the SMCS application:

D C,SMCS
IEE047I 16.39.34 CONSOLE DISPLAY 872
GENERIC=SMCS
SYSTEM APPLID SMCS STATUS APPLID* GENERIC*
S012 SMCSS012 ACTIVE S012 *NONE*
* CURRENT NAME IN USE BY SYSTEM
Use of VTAM generic resources in a parallel sysplex

SMCS supports the use of VTAM generic resources. In a parallel sysplex, this allows an operator who logs on to be connected transparently to one of the active systems of the sysplex rather than being connected to a specific system.

To use generic resources, you should specify the GENERIC parameter on the INIT statement. You should supply one generic name for the entire sysplex:

```
INIT CMDDELIM(:)
  MLIM(1500)
  MONITOR(DSNAME)
  AMRF(N) MPF(00)
  MMS(NO)
  PFK(00)
  RLI(M10)
  UEXIT(N)
  APPLID(SMCS&SYSNAME)       <– SNA specific ACB for the SMCS X
                             application
  GENERIC(SMCSXCF)           <– VTAM Generic resource
```

The following command can be used to change the SMCS GENERIC name:

```
K M,GENERIC=generic
```

**IEE820E** SMCS GENERIC VALUE HAS BEEN CHANGED - SMCS MUST BE RECYCLED ON SOME SYSTEMS.

**IEE712I** CONTROL PROCESSING COMPLETE

Each SMCS application in the sysplex will continue to use the old GENERIC until that SMCS application is recycled, using the V NET,INACT and V NET,ACT commands.

The SMCS GENERIC can be deactivated as follows:

```
K M,GENERIC=*NONE*
```

USING SMCS CONSOLES

Starting the SMCS application

The SMCS application is designed to start and restart automatically, and will attempt to connect to VTAM using the SMCS APPLID every 15 seconds. During the IPL process, the CONSOLE address space automatically starts the SMCS
If the APPLID is deactivated, the SMCS application will attempt to restart and reconnect to VTAM every 15 seconds. As before, you can issue a D C,SMCS to check the status of the SMCS application:

```
D C,SMCS
```

If the APPLID is deactivated, the SMCS application will attempt to restart and reconnect to VTAM every 15 seconds. As before, you can issue a D C,SMCS to check the status of the SMCS application:

```
D C,SMCS
```

Logging on to the SMCS application

Once the SMCS application is active, you can log on to the SMCS application using a LOGON APPLID(...) command. The SMCS Console Selection screen is then displayed:

```
SMCS CONSOLE SELECTION

Enter the Console Name you want to access and press ENTER.

CONSOLE NAME ===>          (Required. This name must have been defined as an SMCS console in CONSOLxx at IPL).

You are attempting to access:

SYSPLEX:  YXCF     SYSTEM:  SØ12

Licensed Materials - Property of IBM
"Restricted Materials of IBM"
5694-AØ1 (C) Copyright IBM Corp. 2ØØ1
All rights reserved.
```

Providing security for SMCS consoles

Now that operator consoles can be located anywhere, each installation must ensure that operator access is properly controlled. SMCS consoles support the LOGON keyword on the CONSOLE statement:
• LOGON (OPTIONAL) indicates that the console doesn’t need to be logged on.
• LOGON (AUTO) indicates that the console is automatically logged on. The userid will be the console name in EBCDIC format.
• LOGON (REQUIRED) indicates that the console must be logged on before commands can be issued.

```
17.42.19 5012  logoff
17.42.19 5012  IEE185i LOGOFF SMCS02  COMPLETE FOR LU=TCP05001
               CN=SMCS02

IEE187i ENTER LOGON PARAMETERS
LOGON          PASSWORD
GROUP          SECLABEL
IEE163i MODE= R
```

Predefined LU
Controlling which physical SNA terminals can act as an SMCS console is one way to implement security, and you can specify in the CONSOL00 parmlib member that a particular console name should always be associated with a particular LU.

```
CONSOLE DEVNUM(SMCS) ROUTCODE(ALL) <- SNA MCS console
       NAME(SMCS01) <- Name of the console
       LU(L020885) <- Predefined VTAM LU name
       MSCOPE(*ALL)
       RBUF(15) PFKTAB(PFKTA00)
       AUTH(ALL)
       MONITOR(JOBNAME5-T)
       CON(N) SEG(16) DEL(R) RNUM(19) RTME(1) MFORM(S,J,T) AREA(NONE)
```

Once the LU is logged on to the SMCS application, the console becomes active, bypassing the SMCS selection screen.

You can turn off the predefined LU of an SMCS console using the VARY CN command:

```
VARY CN(consname),LU=*NONE*
```

SMCS permanent consoles
The LOGAPPL VTAM parameter in the definition of a 3270 terminal indicates that this particular LU automatically logs on
to a particular application when the LU becomes active.

L020885 LOCAL CUADDR=885, CU ADDRESS X
I STATUS=ACTIVE, INITIAL ACTIVE X
TERM=377, 3270 DISPLAY TERMINAL X
FEATUR2=MODEL2, DEFAULT SCREEN SIZE X
MODETAB=MTTABLE, X
DLOGMOD=M2BSCQ, X
LOGAPPL=SMCS&SYSNAME., <- automatic logon X
USSTAB=USSTAB00

By indicating that a particular 3270 LU should automatically log
on to the SMCS application, a console can be activated
automatically once VTAM is active, in much the same way that
MCS consoles activate automatically during IPL.

In the same way, when you deactivate/activate that 3270 LU,
the associated SMCS console is automatically deactivated/ activated:

V NET, INACT, ID=L020885, FORCE

IST097I VARY ACCEPTED
IST129I UNRECOVERABLE OR FORCED ERROR ON NODE L020885 - VARY INACT
SCHED
IEE057I ACCESS TO CONSOLE: SMCS01 LU: L020885 LOST 937
RSN: 00000018 CODE: LT01
IST105I L020885 NODE NOW INACTIVE
IEE055I CONSOLE SMCS01 (LU: L020885) IS INACTIVE

V NET, ACT, ID=LM020880

IST097I VARY ACCEPTED
IST093I LM020880 ACTIVE
IEE055I CONSOLE SMCS01 (LU: L020885) IS ACTIVE

The result of the DISPLAY CONSOLE command shows a new
type of console (SM):

D C
IEE889I 16.12.26 CONSOLE DISPLAY 947
MSG: CURR=0 LIM=1500 RPLY: CURR=0 LIM=10 SYS=S012 PFK=00
CONSOLE/ALT ID ---------------- SPECIFICATIONS ----------------
SYSLOG COND=H AUTH=CMDS NBUF=0 UD=N
ROUTCDE=ALL
...
SMCS01 03 COND=A,SM AUTH=AL NBUF=0 UD=N <-
type SM = SMCS
L020885 AREA=Z MFORM=T, S, J
How to talk 3270

In our era of PCs and graphical user interfaces (GUIs), a 3270 terminal emulator may sound a strange concept – a whole generation of young people has never known anything but Windows or X-windows. But it’s not really that unusual.

The Unix environment has a terminal character environment (a VT100 or similar), which is also emulated within a window in a GUI environment. The same is true for a DOS session. However, the 3270 is the most powerful of these three environments. It’s not just a means to display characters, but has many features that make it nearly graphical. And, with the appropriate software (Graphical Data Display Manager – GDDM), it can be a truly graphical environment, in the sense that you can manipulate each pixel individually. I’m not going to go into GDDM here, however, because what this article aims to do is introduce you to the fun of playing with a 3270 environment.

This article was not written for people already used to the 3270, although they might find some points useful; nor does it cover all its possibilities. It was written for people who have a curious mind and who like to investigate and learn to do new things. It was written for people who have seen those drop-down boxes in ISPF and want to know how that’s done. Or who’ve seen blinking characters or reverse video, or even the hilite feature of the ISPF editor (have you tried typing ‘hilite auto’ in the command prompt while editing a program?). These are the types of thing I’ll be covering here.
This article doesn’t describe ISPF or any other high-level (CICS BMS, for example) method of playing these tricks. It explains the raw commands that are behind them. In other words, it explains how to talk 3270 – the same language that a real 3270 terminal understands, and that a PC emulator mimics into something more or less identical.

The 3270 language is independent of the environment in which it’s used. It can be used in TSO, in CICS, in VM, in ICCF, whatever; it’s just a stream of bytes with a special meaning. That stream of bytes – a 3270 datastream – can be created in many ways: with an editor, written to a file by a program, and so on. Once it’s been created, you need to use some function of the environment in which you’re working to send it to the display device without any modification. If you want to do it in TSO, you must use an IBM-supplied Assembler macro (TPUT); if you’re in CICS, you can use the SEND FROM command to display your data (not SEND TEXT or SEND MAP, because these will interfere with your datastream).

HOW TO BUILD A DATASTREAM

The first character that must be present in a datastream is a Write Control Character (WCC). This is a single byte that performs a set of initializations, such as unlocking the keyboard, sounding the alarm, and so on. For practical purposes, I use only two WCCs: x’F0’ for no-alarm, and x’F5’ for alarm. Note that in the examples below, this WCC character is not present. This is because, if you work under TSO and use the TPUT macro to send the datastream, you must insert the WCC character at the beginning. But if you work under CICS with the SEND command, you don’t need it, because CICS automatically builds the WCC for you, based on the SEND options you specify: ALARM, FREEKB, ERASE, and so on.

There are several formats for 3270 terminals. The most commonly used is the 24*80 lines/columns, and that’s the one we’ll concentrate on here. The addresses in the terminal are specified in terms of an absolute position, starting at position 0.
(the upper left corner), going down to the lower right corner, or position $24 \times 80 - 1 = 1919$. After that, the position wraps back to the beginning of the screen.

A datastream consists of bytes that contain both the text to display and some special bytes that are interpreted as orders. In general terms, those orders consist of a byte that corresponds to a specific order, followed by one or more bytes that are the ‘arguments’ for that order. Anything other than those orders and their arguments is considered displayable stuff and will be written to the screen.

Let’s take an address order as an example. If you want to display a sentence in a specific line and column on your terminal, you can either stuff your datastream with spaces until the desired relative position is reached or you can send an order that makes that position the current one, and add your sentence right after it. So, to write ‘Here I am’ at line 16, column 30, you can either lead it with $15 \times 80 + 30 - 1$ or 1229 spaces, or simply create the following text (shown in hexadecimal and in character) immediately followed by your sentence.

$$x'11534D' \text{Here I am}$$

where hexadecimal ‘11’ is the ‘set buffer address’ (SBA) order, and the two bytes that follow represent position 1229.

First, write 1229 in binary, with a total of 12 bits: 010011001101. Now, separate those twelve bits into two blocks of six, and to each block add on the left side two more bits, for example, ‘01’:

$$01 \ 010011 \ 01 \ 001101 = x'534D'$$

Now you have 16 bits, or two bytes, that form the argument of the address order. These two bytes mean ‘absolute position 1229’. This special way of representing an address is known as 12-bit addressing.

In practice, you can develop a very simple algorithm to calculate the two bytes that represent a given address. You can either do this by bit shifting, ANDs, and ORs, or by using a mathematical implementation. If you look closely, you’ll see that the bit transformation operated above corresponds to the following:
12bitaddr = \( (\text{position} \mod 64 + 64) \times 256 + \text{position} \div 64 + 64 \)

where \( \mod \) represents an integer division and \( \div \) represents the remainder. I use a REXX procedure to calculate addresses, but you can do it any way you like.

As we’ve seen, each of the orders in a datastream takes up three bytes, and you can have as many as you like within your datastream, intermixed with the characters you want to display. However, if you don’t do anything else, your display will have only the default settings – that means regular text and an entirely unprotected screen.

The next level of sophistication is to define fields, which basically means creating protected and unprotected areas in the screen. There are two ways to do this, depending on the level of characteristics you want to assign. The most simple field definition consists of a ‘start field’ (SF) order, indicated by hexadecimal x’1D’, followed by a byte where each bit (or combination of bits) represents an attribute or a characteristic of that field – for example, should the field be protected, unprotected, or numeric only? Should the display be normal, dark (invisible), or bright? Should it be light-pen detectable?

The combination of these characteristics results in a byte called the attribute byte. Since the two high-order bits of this byte aren’t used for characteristics definition, they can be chosen in such a way that an attribute byte is always represented by a character above space. Some common attribute characters are shown in Figure 1.

If you’re familiar with CICS and BMS, you’ll notice that these attributes are the same as those found in the ATTRB parameter of the DFHMDF macro. CICS also provides copybooks (DFHBMSCA) for several programming languages, where attribute byte characters are equated to words like DFHBMPRO etc.

Each field order, along with its characteristics, is valid until another field order is reached. In practice, field orders are normally placed immediately following an address order and before any text to display in that field.
So far, then, a datastream will consist of one or more sequences of

```
set-buffer-address address    start-field attribute   text to display
```

You can of course omit addresses if you don’t need to reposition your current address. Or you can omit the text if you want an open input field without anything written on it. This is better explained by example. Imagine that I want to create the following screen, consisting of a prompt arrow, an input field 30 bytes long, and a small text afterwards. And I want it centred on the screen, beginning at line 12, position 5:

```
=>                               (Enter your name)
```

First, let’s calculate the address of our first field, the arrow: it works out as 11*80+4 = 884, which corresponds in 12-bit addressing mode to x’4D74’. Now let’s imagine that we want the arrow to be protected, bright, and autoskip, which corresponds to a byte attribute ‘8’ or x’F8’. The first field therefore consists of x’114D741DF8’, followed by the arrow characters.

The second field is placed right after the arrow, so there’s no need to reposition it, which means that no address order is needed. All that’s required is a new field attribute to unprotect the screen. A valid byte attribute for that purpose is the space, or x’40’. So, we just add to the stream x’1D40’.

The third field marks the end of the input, so it must again be a protected and autoskip field. If we also choose normal display instead of bright, then we can choose a ’0’ or x’F0’ as the attribute byte. But we also need to specify the new position, since there’s no text in the input area, and we don’t want to stuff

<table>
<thead>
<tr>
<th>Attribute byte</th>
<th>Hex</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Space</td>
<td>X’40’</td>
<td>Unprotect, normal display</td>
</tr>
<tr>
<td>(</td>
<td>X’4D’</td>
<td>Unprotect, dark</td>
</tr>
<tr>
<td>0</td>
<td>X’F0’</td>
<td>Protect, normal display, autoskip</td>
</tr>
<tr>
<td>8</td>
<td>X’F8’</td>
<td>Protect, bright, autoskip</td>
</tr>
<tr>
<td>J</td>
<td>X’D1’</td>
<td>Numeric, normal display</td>
</tr>
</tbody>
</table>

*Figure 1: Common attribute characters*
it with spaces. So, calculating the address for column 40, we get the address x'4E58'. The third field will therefore consist of x'114E581DF0', followed by the text. The datastream will therefore be:

\[
\text{x'114D741DF8' ==> x'1D4Ø114E581DFØ'}(\text{Enter your name})
\]

Correct? Well, almost, but not entirely. Since the screen is unprotected by default, what happens between the upper left corner (position 0) and our first protected field, the arrow, situated in line 12? It’s all an open area where we can type at will, which is not something we want. So, to protect the screen from the left up to our first field, we must lead the above sequence with a ‘set buffer address’ for position zero and a start field protected order: x'1140401DF0'.

Finally, the last detail: the cursor. If we issue no command, it will appear at position zero. Since we want it positioned in our unprotected field, in front of the arrow, we must use an ‘insert cursor’ order, or IC, which consists of a single byte, x'13', that we can place in the datastream anywhere we like; it can appear in the middle of the text, or after an address or field order. So our final datastream will be as follows:

\[
\text{x'114Ø4Ø1DFØ114D741DF8' ==> x'1D4Ø13114E581DFØ'}(\text{Enter your name})
\]

As we saw above, the start field order x'1D' is the simplest way of defining a field. But if you want more sophistication, like colour or reverse video, you must use the ‘start field extended’ (SFE) order instead.

The hexadecimal code for an SFE is x'29', followed by a byte that indicates the number of byte pairs that follow it. A byte pair consists of two bytes, where the first indicates the characteristic to define (for example, colour) and the second its value. The number of byte pairs that follow an SFE order is variable, depending on how we want the field to be. We therefore need to say how many byte pairs we specify.

The characteristics that can be defined and their possible values are summarized in Figure 2. Note that the code that indicates the basic attributes used in the simple SF orders is now x'C0', instead of x'1D'.
So, if we want our arrow in the above example to be yellow, we need to specify a colour code x'42' followed by the yellow value x'F6'. And we must also indicate the basic attribute that makes the field protected (X'F8') preceded by the attribute indicator x'C0'. This gives us two byte pairs in our SFE order, which means the order will be:

```
x'290242F6C0F8' (SFE: 2 byte pairs) (colour: yellow) (attrb: prot, skip)
```

This SFE order replaces the simple SF order (x'1DF0') that we had previously, which means that our stream becomes:

```
x'114D001D0114D74290242F6C0F8' => x'1D4013114E581DF0' (Enter your name)
```

If we also want our arrow in reverse video, then we add another byte pair – hilight (X'41') with the desired value x'F2' – and our SFE, now with three byte pairs, becomes X'290342F641F2C0F8'. The sequence in which the byte pairs are specified is not important, as long as they correspond to the total number indicated.

The last type of attribute mentioned above – the outline – is probably less known and less used than the others. Indeed, I only discovered it quite recently, and not all 3270 emulators can display outline.

Field outlining consists of drawing a thin line above, below, or at the sides of a field, in any combination. The above and below lines are drawn just in between the text rows, and don’t occupy a character cell. The left line is drawn in the ‘dead’ byte (the

---

**Table: Characteristics that can be defined and their possible values**

<table>
<thead>
<tr>
<th>Type</th>
<th>Code</th>
<th>Possible values and meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATTRIBUTE</td>
<td>X'C0'</td>
<td>Same attribute bytes used with SF order (X'1d')</td>
</tr>
<tr>
<td>COLOR</td>
<td>x'42'</td>
<td>X'F1' to x'F7': Blue, Red, Pink, Green, Turq, Yellow, White</td>
</tr>
<tr>
<td>HILIGHT</td>
<td>x'41'</td>
<td>X'00' (No hilight) x'F1' (blink) x'F2' (reverse) x'F4' (underline)</td>
</tr>
<tr>
<td>OUTLINE</td>
<td>X'C2'</td>
<td>From X'00' to X'0F', in any combination of the following values: x'01' (under), x'02' (right), x'04' (over), x'08' (left)</td>
</tr>
</tbody>
</table>

---

*Figure 2: Characteristics that can be defined and their possible values*
position just before a field, where the cursor never stops) that precedes the field, and the right line goes into the next field’s dead byte; both left and right lines occupy a character cell.

The field outline code is x'C2', and the value is a byte in which the four left bits are zero and the four right bits each indicate a line position, in any desired combination, so the byte can range from x'00' (no outlining) to x'15' (full box).

If we wanted our example input field to be fully outlined, so that it appeared with a box drawn all around it, we would start by replacing the start field (SF) order with an SFE order. We would then choose the byte pairs: the outline pair (x'C20F'), the attribute pair (X'C040'), and perhaps also a colour pair, to avoid the default colour. Let’s imagine, however, that in this case we aren’t bothered about colour specification. Our SFE will be: x'2902C20FC040', and our stream becomes:

x'114Ø4Ø1DFØ114D74290242F6CØF8' ===> x'29Ø2C2ØFCØ4Ø13114E581DFØ'(Enter your name)

Note that outline should not be confused with underline (which is part of the ‘hilight’ feature). An underlined field has a thicker line than a lower outline. It’s also possible to have a field both outlined and underlined, and the two lines will be clearly distinct. Note also that the drop-down boxes of ISPF (and DITTO) menus are not made with outlining (this is discussed in more detail later).

The last order to discuss is the ‘set attribute’ (SA) order, which modifies the characteristics of a field starting at the point where it is inserted. To undo this modification and restore the field to its previous characteristic, issue another SA order for the same characteristic with the default value x'00'. It consists of three bytes: the SA order code (x'28') followed by a single byte pair, identical to the byte pairs used in SFE. An SA can be inserted anywhere in a stream: after a set buffer address order, in the middle of text, and so on. Unlike start field orders that take up a ‘dead’ byte, SA orders don’t occupy the screen, so you can assign a different colour to each letter of a word, or make it appear as reverse, or blinking, etc. By the way, this is how the ISPF editor creates those effects when you FIND text or when
you HILIGHT the syntax of a program source.

Let’s say that you want the last sentence of our example stream to appear with the parentheses blinking, and a different colour to each letter of the word ‘Enter’. The SAs to use would be x’2841F1’ (blink), x’284100’ (undo the blink), and x’2842Fn’ for the colours, where \( n \) is the colour number.

That part of the stream would become:

\[
x'2841F1'( '284100 2842F4'E'2842F7'n'2842F3't'2842F5'r your name x'2841F1')
\]

which reads: set blink, ‘(’, undo blink, set colour green, ‘E’, set colour white, ‘n’, etc.

I suggest that you pause here and practise a little based on the above examples, until you become familiar with the whole process, starting simply and gradually increasing the degree of complexity. The most common causes of error are malformed addresses, incorrect values in byte pairs, incorrect number of byte pairs specified in SFE, and so on.

ALTERNATIVE CHARACTER SETS

Once you feel at ease with datastreams, you can progress to the next level: alternative character sets. Let’s look now at how those drop-down boxes are made.

As you know, in PCs, the ASCII codepages sometimes contain a set of symbols that allow all types of boxes and rectangles to be drawn; they even have two versions of them: a thin one with only a single line, and a thicker one with a double line. Well, the 3270 has a similar possibility if we use an alternative character set instead of the standard EBCDIC one. This alternative EBCDIC is meaningful only above x’40’ or space. Instead of letters, this set contains the symbols necessary to draw boxes, in a thin version (the one used by ISPF drop-down menus) and a thicker version (a single trace but larger) that is not so commonly used. The remaining codes of this alternative EBCDIC consist of Greek letters, mathematical symbols, and so on.
It's very easy to access this alternative character set. If we want a given character to be displayed by its alternative value, we simply precede it with hexadecimal '08', the order for 'graphic escape' (GE). This order applies only to a single character.

The EBCDIC characters for a thin version of the box are shown in Figure 3, where a complete box is drawn, including an inner cross to show all the possibilities. Hex codes A2 and 85 represent straight lines, horizontal and vertical. The remaining codes represent corners and line intersections. The thicker version must be built in a slightly different manner – there are no ‘corners’ and no ‘intersection’ characters. Instead, these are formed in the image by putting together two or more different characters. Figure 4 shows how the full box would be made.

For example, let's draw a simple thin rectangle, three lines high by ten characters wide, in yellow, and protected. For simplicity, let’s start it at the upper left-hand corner of the screen. The following code is separated into three lines to make it easier to read; each line begins with a set buffer address plus the characteristics yellow and protected:

```
114040290242F5C0F0 08C508A208A208A208A208A208A208A208A208D5
114150290242F5C0F0 088511415A0885
114260290242F5C0F0 08C408A208A208A208A208A208A208A208A208D4
```
Note that the middle line has the left side address (line2, column 1) followed by the vertical line, plus the right side address (line2, column 10), followed by another vertical line. The other two lines contain the corners and the horizontal lines.

This introduces us to another possibility: the ‘repeat to address’ order, or RTA. Instead of repeating ‘08A2’ several times, I could also find that the last occurrence of ‘08A2’ is at column 10, line 1, and write the first line as:

```
x'114040290242F5C0F0 08C5 3C404A 08A2 08D5
```

which reads 3C (repeat to address) 404A (line 1 col 10) character ‘08A2’.

The RTA order has the format ‘3C <stop address> <character>’. Character can either be a ‘graphic escape’ plus character, as above, or a single regular EBCDIC character.

Similarly, the bottom line in our rectangle could be written:

```
x'114261290242F5C0F0 08C4 3C426A 08A2 08D4
```

A datastream doesn’t need to be created or sent in address sequence. This means that you can send orders to the lower corner of the screen, then to the top, then to the middle, etc. Bear in mind, however, that what you send may overlay something you’ve previously sent.

This means that you can send a screen containing ordinary text fields, and then add boxes to it. One interesting possibility that I’ve exploited several times is in CICS, where you can have ordinary ‘Send Map’ commands, and follow them with a ‘send’ of a 3270 datastream containing boxes or lines that will enhance the BMS map look.

Figure 5 summarizes all the orders we’ve discussed, with their codes and values.

**HOW TO SEND DATASTREAMS**

The easiest way to send a datastream to a terminal is in CICS, where you need only create a data area in a program containing
the stream and then issue a ‘SEND’ command pointing at that area with the correct length. Don’t forget that in this case you don’t need the ‘WCC’ initial character because the SEND command will build it for you.

Under TSO, you need to do it in Assembler, and your data must start with a WCC. You must create a program containing the following macros, where R3 points to the datastream and R4 points to a fullword containing its length.

```
STFSMODE ON, INITIAL=YES
STTMPMD ON
STLINENO LINE=1
TPUT (R3),(R4),FULLSCR,,HOLD
TGET (R3),(R4), ASIS
STFSMODE ON
```

The TPUT macro sends the data to the terminal, and your program then waits for your keyboard action. When you do something (press Enter, or a PF), TGET receives your input to the area indicated by R3, and for a maximum length of R4. In this example, I’ve used the same area for output and input; for a real application, the TGET area would be a specific area to receive your input.

For testing purposes, I suggest that you oversize both your sending buffer and your sending length. What I mean by this is that you can create your datastream and add a few hundred

<table>
<thead>
<tr>
<th>Order</th>
<th>Hex code</th>
<th>Arguments</th>
</tr>
</thead>
<tbody>
<tr>
<td>SBA – set buffer address</td>
<td>X'11'</td>
<td>Address in 12-bit format</td>
</tr>
<tr>
<td>SF – start field</td>
<td>X'1D'</td>
<td>Attribute byte</td>
</tr>
<tr>
<td>SFE – start field extended</td>
<td>X'29'</td>
<td>Number of attribute pairs,</td>
</tr>
<tr>
<td>attributes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SA – set attribute</td>
<td>X'28'</td>
<td>Attribute pair</td>
</tr>
<tr>
<td>RTA – repeat to address</td>
<td>X'3C'</td>
<td>Address, character or address, X'08, character</td>
</tr>
<tr>
<td>GE – graphic escape</td>
<td>X'08'</td>
<td>Character</td>
</tr>
<tr>
<td>IC – Insert cursor</td>
<td>X'13'</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 5: Resume of all orders, codes, and values*  

low-values to the end of it, and declare a generous size, as long as that size fits anywhere within those low-values. This way, you won’t have to worry about counting the exact number of bytes to send, because sending some extra low-values won’t interfere with your screen.

REAL TERMINALS AND PC EMULATORS

If you’re working with a real 3270 terminal, you must be sure, before you send a datastream, that it supports all the features you put in it. There are terminals that can’t do hilight (blinking, reverse, underscore), and others won’t do outline. If that’s the case and you send data containing these features, you’ll get incorrect displays, or even a terminal error (eg ‘Prog 402’ in the status line). To avoid these situations, ‘intelligent’ programs, like ISPF, will query the terminal characteristics before building the datastream and sending it.

PC emulators aren’t so particular – you can send them anything, and they will simply discard what they can’t deal with. There are some emulators that will do everything, others will ignore outline, and so on. Emulators that can’t deal with graphic escape orders will display the standard EBCDIC character instead of the graphical equivalent. But even those that do display GE will not always get things right. You don’t even need to create a datastream with graphic data to see how your emulator will behave: try opening an ISPF menu and see if you get a perfect box or if it’s somehow broken, with non-contiguous lines.

The problem is that some font types and sizes won’t join the 3270 character cells perfectly, leaving gaps between them. If this happens, try changing the font or size until you get a better display.

An interesting feature in some emulators is that you can trace the datastream being sent or received into a PC file and look at what’s there. This is a good way to learn a few tricks, when you want to know how a particular screen is built. You can also use
this feature to study the receiving data (the bytes that the
terminal sends back to your program when you hit a transmitting
key). As a bonus, I’ll tell you that the receiving data contains the
key code (Enter, PA, PFxx, etc), the cursor address at that
moment, and the field addresses and contents that should be
transmitted. I’ll leave the rest to your curiosity.

For more information than is contained in this article, I suggest
that you refer to the 3270 Datastream Application Programming
Reference, and the CICS Application Programming Guide.

Luis Paolo Ribeiro
Systems Engineer, Edinfor (Portugal) © Xephon 2002

Dynamically creating a NERD chart

The VTAM systems programmer who maintained our tele-
communications network kept what everyone in his group
referred to as a NERD chart – an elemental pictograph of the
TITAN’s network. It accurately depicted each host system’s
connection into sundry 3745s, and provided the names of
cross-domains, SSCP's, NCP's, and lines associated with SNI
connections, along with the numbers of the subareas associated
with them. When he left, the TITAN’s NERD chart quickly got
out-of-date, so I created PPGMAPVR.

PPGMAPVR dynamically generates a NERD chart for the
DOMAIN on which it executes, containing information gleaned
mostly from control blocks anchored in VTAM’s ATCVT. My key
objective when creating PPGMAPVR was to reproduce a
NERD chart, but my efforts were extended to include a map of
virtual and explicit routes and cross-domain definitions as well.
If you prefer to do without this, you can insert a branch
instruction to bypass its creation. (The anchor for the virtual
route control blocks is in a field named ATCVRNDX within
VTAM’s ATCVT; for explicit route queues, it’s in a field named
ATCERTP; and for SSCP entries in a field named ATCSSCPT.)
PPGMAPVR has been executed on OS/390 release 2.9 with NCP 7.6 and VTAM 4.3. As well as SYS1.MACLIB, two other datasets are required in order to assemble PPGMAPVR: SYS1.AMODGEN and a version of AHASMAC, V2R5M0 or V2R8M0. It must be link-edited into an authorized library with an option of AC=1 specified. PPGMAPVR can be invoked with the following JCL:

```plaintext
//CHART   EXEC PGM=PPGMAPVR
//SYSPRINT DD SYSOUT=*  
```

### SOURCE

**TITLE 'PPGMAPVR - MAP A NETWORK'S SUBAREAS'**

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* THE PURPOSE OF THIS ROUTINE IS TO CONSTRUCT A "NERD" CHART AND, *
* ALSO, TO MAP VIRTUAL AND EXPLICIT ROUTES IN A NETWORK. *

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SPACE 2

PPGMAPVR CSECT

PPGMAPVR AMODE 31

PPGMAPVR RMODE 24

SPACE

PRINT NOGEN

SPACE

USING PPGMAPVR, R13, R12

ESTABLISH PPGMAPVR ADDRESSABILITY

USING PSA, RØ

ESTABLISH PSA ADDRESSABILITY

SPACE

BAKR R14, RØ

PRESERVE ENVIRONMENT AT ENTRY

LR R13, R15

PRIME BASE REGISTER

SPACE 1

* SIGH - RAN OUT OF ADDRESSABILITY AND NEEDED ANOTHER BASE REGISTER

LA R12, 2Ø48(R13)

CONSTRUCT SECOND

LA R12, 2Ø48(R12)

BASE REGISTER FOR PPGMAPVR

EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* PROCESS PARAMETERS SPECIFIED ON THE EXEC STATEMENT. *
* IF FIRST DIGIT IS NOT A ZERO, THEN IT'S A PASSWORD. *

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SPACE 1

L R1, Ø(R1)

POINTER TO PARM LENGTH FIELD.

LH R2, Ø(R1)

LENGTH INTO 2.

LTR R2, R2

TEST IF ANY PARMS.

BZ PPGNOPRM

BRANCH IF NONE.

SPACE

CLI 2(R1), C 'Ø'

TEST IF ERROR MESSAGES DESIRED
**BL**  PPGCALYN  BRANCH IF NOT
**BE**  PPGSETSW  BRANCH IF SO

**PPGPRMER**  WTO  '01R202E PARM ERROR - MUST BE Ø, PSWD(<=8), OR ØPSWD'
**LA**  R15,8  SET AN UNSUCCESSFUL RETURN CODE
**PR**  R14  BACK TO DUST

**SPACE**

**PPGSETSW**  MVI  PPGSW,Ø  INDICATE VTAM ERROR CODES DESIRED
**BCTR**  R2,Ø  REDUCE LENGTH BY ONE
**LA**  R1,1(R1)  PSEUDO START OF PASSWORD
**SPACE**

**LTR**  R2,R2  TEST IF ANY ADDITIONAL PARAMETERS
**BZ**  PPGNOPRM  BRANCH IF NONE

**SPACE**

**PPGCALYN**  C  R2,=F'8'  TEST IF PASSWORD EXCEEDS EIGHT BYTES
**BH**  PPGPRMER  BRANCH IF SO
**MVC**  PPGPSWD+1(8),PPGHTICS  BLANK PASSWORD
**SPACE**

**STC**  R2,PPGPSWD  STOW LENGTH OF PASSWORD IN PARM
**BCTR**  R2,RØ  REDUCE LENGTH OF PASSWORD FOR MOVE
**EX**  R2,PPGETAID  COPY PASSWORD TO PARAMETER AREA
**SPACE**  1

**PPGNOPRM**  OPEN  (PPHDCB,OUTPUT)  PREPARE DATA SET FOR USE
**SPACE**

**LOAD**  EP=PPGRDFEP,ERRET=PPGLDERR  LOAD PPGRDFEP (PERHAPS)
**ST**  RØ,PPGCPSUB  STOW ITS ADDR FOR FUTURE REFERENCE
**SPACE**

**PPGLDERR**  LA  R1Ø,58  SET LINES-PER-PAGE
**SPACE**  1
**MODESET**  MODE=SUP,KEY=ZERO  PRETEND TO BE GEORGE
**SPACE**

**ESAR**  R1  GET SECONDARY ASID OF THIS TASK
**ST**  R1,PPHCASID  SAVE IT
**EJECT**

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* OBTAIN THIS SYSTEM'S SMF IDENTIFICATION AND NODE NAME *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

**SPACE**  1

**L**  R1,CVTPTR  ADDRESS OF CVT
**USING**  CVTMAP,R1  ESTABLISH CVT ADDRESSABILITY
**SPACE**  1

**ICM**  R8,15,CVTJESCT  ADDRESS OF JES2 COMMUNICATION TABLE
**BE**  PLRNOJES  ASSUME JES2 IF NOT AVAILABLE
**SPACE**

**USING**  JESCT,R8  ESTABLISH JESCT ADDRESSABILITY
**MVC**  PPHJES2,JESPJESN  STOW TRUE NAME OF SUBSYSTEM
**SPACE**

**L**  R2,JESSSCT  RETRIEVE ADDR OF 1ST SUBSYS COMM TBL
**USING**  SSCCT,R2  ESTABLISH SSCCT ADDRESSABILITY
**SPACE**

**CPTRYAGN**  CLC  SSCTSNAM,JESPJESN  TEST IF THIS ONE BELONGS TO JES2
**BE**  CPGOTJES  BRANCH IF SO
ICM R2,15,SSCTSCTA FETCH ADDRESS OF NEXT SSCT
BNE CPTYAGN CONTINUE SEARCHING FOR JES2
B PLRNOJES PROCESS CONTROL BLOCKS ANYWAY
SPACE
CPGOTJES L R3,SSCTSUS2 GET ADDRESS OF $HCCT
USING HCCT,R3 ESTABLISH HCCT ADDRESSABILITY
SPACE
SR R15,R15 CLEAR A VOLATILE REGISTER
IC R15,CCTNDENL SET LENGTH OF NODE'S NAME
EX R15,PCXFIONA COPY ITS NAME INTO A HOLD AREA
SPACE
LH R15,CCTTONOD GRAB IDENTIFIER OF THIS NODE
CVD R15,PPGTWICE ALTER ITS RADIX
LA R1,PPHJESID+3 POINT TO LAST SPOT FOR 'N' + 1
EMD R1,PPHJESID+3,PPGTWICE+6 CONVERT ID TO EBCDIC
BCTR R1,Ø POSITION BACK ONE CHARACTER
MVI Ø(R1),C'N' STOW CHARACTER 'N' FOR NODE
SPACE 1
L R1,CVTTPTR ADDRESS OF CVT
PLRNOJES L R3,CVTSMCA ADDRESS OF SMF CONTROL AREA
USING SMCABASE,R3 ESTABLISH SMF ADDRESSABILITY
MVC PPHSYSID,SMCASID STOW SMF ID
SPACE
DROP R2,R3,R8 FORGET SMF AND JES2 STUFF
EJECT
**********************************************************************
*          PROVIDE ENVIRONMENTAL INFORMATION                          *
**********************************************************************
SPACE 1
L R1,CVTTPTR FETCH ADDRESS OF CVT
SH R1,PATH256 POINT TO BEGINNING OF PREFIX
USING CVTFIX,R1 ESTABLISH ADDRESSABILITY TO PREFIX
SPACE 1
MVC PATPRODN,CVTPRODN PRODUCT NAME OF OPERATING SYSTEM
MVC PATPRODI,CVTPRODI FMID OF OPERATING SYSTEM
MVC PATNUMB,CVTNUMB RELEASE NUMBER
MVC PATLEVEL,CVTLEVEL LEVEL OF RELEASE
UNPK PATMODEL,CVTMDL(3) CONVERT TO EYE-
TR PATMODEL,PPHTRANS-24Ø READABLE FORMAT
MVI PATMODEL+4,C' ' CLEAR DE TRASH FROM MESSAGE
SPACE 1
L R1,CVTEX2 FETCH ADDRESS OF CVT'S EXTENSION
DROP R1 FORGET CVT
USING CVT2RØØØ,R1 ESTABLISH CVT2RØØØ ADDRESSABILITY
MVC PATNUC,CVTNUCLS IDENTIFICATION OF MEMBER NAME OF NUC
MVC PATHCD,CVTCI OCD IDENTIFICATION OF ACTIVE I/O CONFIG
DROP R1 FORGET CVT
WTO MF=(E,PATWTOOP)
EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* LOCATE AND ESTABLISH ADDRESSABILITY TO NET'S ADDRESS SPACE *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SPACE 1
L    R7, PSAATCVT ADDRESS OF VTAM'S VECTOR TABLE
L    R9, ATCONFT(R7) ADDRESS OF VTAM CONFIGURATION TABLE
MVC  PPGNETID, ATCASID(R7) STOW ASID OF VTAM'S ADDRESS SPACE
MVC  PPHJ NAME, CONIDENT(R9) SET THE NAME OF VTAM'S TASK
L    R15, CONAREA(R9) POINT TO CONFT AREA
MVC  PPHGLIST, CONLIST(R15) SET LIST= OPERAND OF START COMMAND
MVC  PPHVTVVER(4), ATCVTLVL(R7) RELEASE LEVEL OF VTAM
MVC  PPHTVVER+4(4), X'B14'(R7) ** TEMPORARY-RESERVED FIELD **

SPACE 1
TRT  ATCNQNAM(17, R7), PPGPRSTB SEPARATE NETID FROM SSCP NAME
BZ   PPGNSSCP BRANCH IF IMPOSSIBLE
MVC  PPGCLLN1M, 1(R1) STOW NAME OF SSCP INTO MESSAGE
LA   R15, ATCNQNAM(R7) POINT TO NETWORK IDENTIFIER
SR   R1, R15 COMPUTE THE SIZE OF ITS NAME
BCTR R1, RØ DECREMENT BY ONE FOR EX INSTRUCTION
EX   R1, PPGMVPPG COPY NETID TO MESSAGE AREA
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* ENABLE SELECTED REGISTERS TO ACCESS DATA IN VTAM'S ADDRESS SPACE *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SPACE
PPGNSSCP LAM  R4, R6, PPHONE INITIALIZE ACCESS REGISTERS
LAM  R11, R11, PPHONE INITIALIZE ACCESS REGISTER
LAM  R2, R2, PPHONE INITIALIZE ANOTHER ACCESS REGISTER
SPACE 1
LA  R1, R1 SET AUTHORIZATION
AXSET AX=(R1) INDEX TO ONE
EJECT

******************************************************************************

* LOCATE CATALOG'S ASID *
******************************************************************************

SPACE
L    R3, CVTPTTR ADDRESS OF CVT
USING CVTMAP, R3 ESTABLISH CVT ADDRESSABILITY
SPACE 1
L    R5, CVTASVT FETCH ADDRESS OF ASVT
DROP R3 FORGET CVT
SPACE 1
USING ASVT, R5 ESTABLISH ASVT ADDRESSABILITY
L    R4, ASVTMAXU MAXIMUM NUMBER OF ADDRESS SPACES
SPACE 1
PAPLOC TM ASVTENTY, ASVTAVA1 TEST IF ENTRY IS AVAILABLE
BO   PAPGRUVE BRANCH IF SO
SPACE 1
L    R6, ASVTENTY RETRIEVE ADDRESS OF ASCB
USING ASCB, R6 ESTABLISH ASCB ADDRESSABILITY
SPACE 1

ICM  R1,15,ASCBJ  POINTER TO INITIATED JOBNAME
BZ  PAPJBNI   BRANCH IF NONEXISTENT
SPACE 1
CLC  Ø(8,R1),PPHCNAME  TEST IF CORRECT JOB
BNE  PAPGRUVE  BRANCH IF NOT
B    PAPGOTIT  ELSE CONTINUE
SPACE 1
PAPJBNI  ICM  R1,15,ASCBJ  POINTER TO START/MOUNT/LOGON TASK
BZ  PAPGRUVE  FORMAT IT
SPACE 1
CLC  Ø(8,R1),PPHCNAME  TEST IF CORRECT JOB
BE  PAPGOTIT  BRANCH IF SO
SPACE 1
PAPGRUVE  LA  R5,4(R5)  NEXT ENTRY
BCT  R4,PAPLOC  LOOP POWER
B    PCPGNTNET  NO PROBLEM; VOLSER OF CAT VOL N/A
EJECT
*******************************************************************************
*        LOCATE SERIAL NUMBER OF VOLUME THAT CONTAINS THE MASTER         *
*        CATALOG BY SEARCHING THE CAX CHAIN FOR ITS UCB ADDRESS.         *
*******************************************************************************
SPACE
PAPGOTIT  LH  R1,ASCBASID  OBTAIN ASID OF CATALOG'S ADDR SPACE
ST  R1,PPGCATID  STOW ASID OF CATALOG'S ADDR SPACE
BAS  R8,PPHBECAT  ACCESS DATA IN CATALOG'S ADDR SPACE
SPACE
L  R3,CVTPTR  ADDR OF COMMUNICATIONS VECTOR TABLE
USING CVTMAP,R3  ESTABLISH CVT ADDRESSABILITY
L  R4,CVTCBSP  ACCESS METHOD CNTL BLK STRUCTURE BLK
DROP  R3  FORGET CVT
L  R4,CBSCAXCN(,R4)  ADDRESS OF THE CAXWA CHAIN
*  USING IGGCAXWA,R4  SET ADDRESSABILITY TO CAT AUX WRK AREA
SPACE 1
PPGLNCAT  TM  CAXFLGS(R4),CAXMCT  TEST IF MASTER CATALOG
BO  PPGLSLAVE  BRANCH IF SO
ICM  R4,15,CAXCHN(R4)  FETCH ADDRESS OF NEXT WORK AREA
BNZ  PPGLNCAT  CONTINUE TO HUNT FOR MASTER CATALOG
B  PPGETNET  BRANCH IF UNABLE TO LOCATE MASTER
SPACE 2
PPGLSLAVE  L  R2,CAXUCBA(,R4)  FETCH ADDRESS OF UCB
USING UCB0B,R2  PROVIDE UCB ADDRESSABILITY
MVC  PHPCATVL(6),UCBVOLI  STOW NCLOAD'S VOLUME SERIAL NUMBER
SPACE
DROP  R2,R5,R6  FORGET ADDRESSABILITIES
EJECT
*******************************************************************************
*        LOCATE VTAM'S ADDRESS SPACE CONTROL BLOCK                        *
*******************************************************************************
SPACE 1
PPGETNET  BAS  R8,PPHRESET  ENTER SOLO MIO ROLE
SPACE 1
PCPGTNET LH R2, PPGNETID OBTAIN ASID OF VTAM'S ADDRESS SPACE
BAS R8, PPSET ACCESS MULTIPLE ADDRESS SPACES
L R3, CVTPT R ADDRESS OF CVT
USING CVTMAP, R3 ESTABLISH CVT ADDRESSABILITY
SPACE 1
L R5, CVTASVT FETCH ADDRESS OF ASVT
DROP R3 FORGET CVT
SPACE 1
USING ASVT, R5 ESTABLISH ASVT ADDRESSABILITY
LA R6, ASVTFRST POINT TO FIRST ENTRY
LH R1, PPGNETID FETCH ASID OF VTAM
MH R1, PPGH4 COMPUTE OFFSET TO ADDR OF VTAM'S ASCB
L R6, Ø(R1, R6) THEN RETRIEVE ADDRESS OF ASCB
USING ASCB, R6 ESTABLISH ASCB ADDRESSABILITY
L R6, ASCBASXB OBTAIN ADDRESS OF ASCB EXTENSION
USING ASXB, R6 ESTABLISH ASXB ADDRESSABILITY
L R6, ASXBFTCB ADDRESS OF FIRST TCB ON TCB CHAIN
USING TCB, R6 ESTABLISH TCB ADDRESSABILITY
SPACE 1
DROP R5 FORGET ASVT
PATCGLNE L R4, TCBRB P CURRENT RB ADDRESS
USING RBBASIC, R4 ESTABLISH RB ADDRESSABILITY
PATGETRB CLM R6, 7, RBLINKB TEST IF FIRST RB ON CHAIN
BE PATGOTRB BRANCH IF SO
ICM R4, 7, RBLINKB ADDRESS OF PREVIOUS RB
BNE PATGETRB RETRY IF AVAILABLE
B PATDKNOW MUST BE OUT IN LIMBO, AGAIN
SPACE 1
PATGOTRB ICM R5, 15, RBCDE CURRENT CDE ADDRESS
BE PATDKNOW BR IF UNKNOWN CONDITION ENCOUNTERED
USING CDENTRY, R5 ESTABLISH CDE ADDRESSABILITY
CLC CDNAME, PPGISTIN TEST IF CORRECT TCB
BE PATISTCB BRANCH IF SO
ICM R6, 15, TCBLTC ADDRESS OF NEXT TCB ON CHAIN
BE PATDKNOW WHY?
B PATCGLNE PROCESS IT
EJECT
***********************************************************************
* SCAN TIOT ENTRIES FOR A DD STATEMENT WITH A NAME OF NCPLOAD *
***********************************************************************
SPACE 1
PATISTCB L R2, TCBTIO TIOT ADDRESS
USING TIOT1, R2 ESTABLISH TIOT ADDRESSABILITY
SR R15, R15 ZERO INDEX REGISTER
LR R1, R15 SET ZEROES FOR COMPARE
PPGTFINI C R1, TIENTRY TEST IF END OF TIOT
BE PATDKNOW BRANCH IF SO
CLC TIODEDDNM, PPGNCPLD SCAN FOR 'NCPLOAD' DD STATEMENT
BE PHAVEALT BRANCH IF LOCATED
IC    R15, TIOELNGH    LENGTH OF THIS DD ENTRY
LA    R2, Ø(R15, R2)   NEXT DD ENTRY
B     PPGTFIN    CONTINUE SEARCH
SPACE 1
USING UCBDBB, R3    SET UCB ADDRESSABILITY

PHAVEALT    SR    R3, R3    CLEAR VOLATILE REGISTER
ICM    R3, 7, TIODEFSRT    FETCH ADDRESS OF NCPLOAD'S UCB
MVC    PHPLVVOL(6), UCBVOLI    STOW NCPLOAD'S VOLUME SERIAL NUMBER
SPACE 1
DROP    R2, R3, R4, R5    FORGET ADDRESSABILITIES
SPACE 1
PATDNOW    DS ØH
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

* MAP VIRTUAL ROUTES
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* PSA + X'408'(PSAATCVT) ===> ATCVT
* ATCVT + X'69C'(ATCVRNDX) ===> VRIT
* VRIT + 4*(DESTSA #) ======> VRBLK
* VRBLK + 2 ==================> VRBVRN (VIRTUAL ROUTE NUMBER)
* VRBLK + X'58' ===========> FIRST OF THREE CONTIGUOUS VRBFSTS'
* EACH OF WHICH IS X'30' BYTES LONG.

SPACE 1
L    R5, ATCVRNDX(R7)    ADDRESS OF VIRTUAL ROUTE QUEUES
SPACE
LA    R9, 80    SET MAXIMUM AMOUNT OF PROCESSING
SPACE
L    R3, ATCHOSTA(R7)    NUMBER OF HOST SUBAREA
CVD   R3, PPGTWICE    ALTER RADIX OF SUBAREA NUMBER
ED    PPHSUBAH, PPGTWICE+6    BEAUTIFY NUMBER OF THIS SUBAREA
SPACE

PPGISLUV    ICM    R2, 15, Ø(R5)    FETCH ADDRESS OF VIRTUAL ROUTE BLOCK
BNE    PPGCAPLS    BRANCH IF IT'S AVAILABLE
SPACE

PPHNXTSA    LA    R5, 4(R5)    POINT TO NEXT VRBLK POINTER
BCT    R9, PPGISLUV    PROCESS IT
B    PCPMAPER    PROCESS EXPLICIT ROUTES
SPACE

PPHFORM    ICM    R2, 15, VRBFXCHN(R2)    FETCH ADDRESS OF NEXT VRBLK
BE    PPHNXTSA    BRANCH IF NONEXISTENT
SPACE

PPGCAPLS    CLI    VRBTYPE(R2), VRBCID    TEST IF 'TIS A TRUE VRBLK
BE    PPGISAGO    BRANCH IF SO
BAS    R8, PPHRESET    ENTER SOLO MIO ROLE
WTO    'INVALID VRBLK'
B    PPHCLOS2
EJECT

PPGISAGO    MVI    PPHCC, C'    INITIAL BLANK
MVC    PPHVRBLK(PPHDLEN-1), PPHCC CLEAR DISPLAY AREA
MVC PPHSUBA#, PPHSUBAH  STOW SUBAREA'S NUMBER IN OUTPUT AREA

SR R1, R1  REMOVE DETRITUS FROM GPR #1
IC R1, VRBVRN(R2)  OBTAIN VIRTUAL ROUTE NUMBER
CVD R1, PPGTWICE  ALTER RADIX OF VR NUMBER
MVC PPHVR#, PPGPATSA  SET EDIT PATTERN IN OUTPUT AREA
ED PPHVR#, PPGTWICE+6  BEAUTIFY NUMBER OF THIS VIRTUAL ROUTE

SPACE

IC R1, VRBIER(R2)  OBTAIN INITIAL EXPLICIT ROUTE NUMBER
CVD R1, PPGTWICE  ALTER RADIX OF ER NUMBER
MVC PPHER#, PPGPATSA  SET EDIT PATTERN IN OUTPUT AREA
ED PPHER#, PPGTWICE+6  BEAUTIFY NUMBER OF THIS VIRTUAL ROUTE

SPACE

L R1, VRBADJSA(R2)  OBTAIN VIRTUAL ROUTE NUMBER
CVD R1, PPGTWICE  ALTER RADIX OF VR NUMBER
MVC PPHAJSA#, PPGPATSA  SET EDIT PATTERN IN OUTPUT AREA
ED PPHAJSA#, PPGTWICE+6  MAKE IT PRETTY

SPACE

L R1, VRBDSTSA(R2)  OBTAIN DESTINATION SUBAREA NUMBER
CVD R1, PPGTWICE  ALTER ITS RADIX
MVC PPHDEST#, PPGPATSA  SET EDIT PATTERN IN OUTPUT AREA
ED PPHDEST#, PPGTWICE+6  MAKE IT PRETTY
EJECT

LA R11, VRBFSTS(R2)  POINT TO VRBI BASE
LA R15, PPHTP1  POINT TO FIRST ENTRY
USING PPGBASE, R15  ESTABLISH PPGBASE ADDRESSABILITY
LA R1, 3

PPHDOBAS TM VRBFCFSM(R11), 3  TEST FLOW CONTROL STATE
BO PPGOPEN  BRANCH IF OPEN
BM PPGHELD  BRANCH IF HELD
MVC PPHFCFSØ, PPGSET  SHOW BLOCKED VR

PPHCKFSM SR R14, R14  CLEAR A VOLATILE REGISTER
IC R14, VRBVRFSM(R11)  FETCH STATE OF VR
MH R14, PPGH4  COMPUTE OFFSET OF ENTRY
LA R14, PPVGRVAL(R14)  POINT TO CORRECT ENTRY FOR STATUS
L R14, Ø(R14)  RETRIEVE ADDRESS OF CONSTANT
MVC PPHSTATØ, Ø(R14)  STOW STATUS IN OUTPUT AREA

SPACE

CLI VRBVRFSM(R11), 5  TEST IF STATE OF VR IS ACTIVE
BE PPGVRACT  BRANCH IF SO

SPACE

PPHDOFS SR R14, R14  REMOVE DETRITUS FROM GPR #1
IC R14, VRBTP1(R11)  OBTAIN TRANSMISSION PRIORITY
CVD R14, PPGTWICE  ALTER RADIX OF TP NUMBER
MVC PPHTPØ, PPGPATSA  SET EDIT PATTERN IN OUTPUT AREA
ED PPHTPØ, PPGTWICE+6  BEAUTIFY TRANSMISSION PRIORITY NUMBER

SPACE

MVC PPHRTØ, PPGPRI  ASSUME PRIMARY ROUTE
TM VRBVRFSM+1(R11), VRBPRI  TEST IF PRIMARY ROUTE
BO  PPGLSNXT
MVC  PPHRTO, PPGSEC  SET SECONDARY SPACE

PPGLSNXT LA  R11, 48(R11)  POINT TO NEXT VRBFSTS
LA  R15, PPGBASEL(R15)  POINT TO NEXT PRINT AREA
BCT R1, PPFDDBAS  PROCESS ALL THREE VRBFSTS'S
BAS R1, PCPSCRIB  TRANSCRIBE DATA
B  PPHFORM  PROCESS NEXT ENTRY SPACE

PPGOPEN MVC  PPHFCFSØ, PPGOPENC  SHOW ACTIVE VR
B  PPHCKFSM  CONTINUE...

SPACE

PPGHELD MVC  PPHFCFSØ, PPGHELDC  SHOW HELD VR
B  PPHCKFSM  CONTINUE...

SPACE

PPGVRACT LH  R14, VRBSECNT(R11)  OBTAIN COUNT OF SESSIONS ON THIS VR
CVD R14, PPGTWCIE  ALTER RADIX OF COUNT
MVC PPH#LUSØ, PPGPATLU  SET EDIT PATTERN IN OUTPUT AREA
ED  PPH#LUSØ, PPGTWCIE+5  BEAUTIFY NUMBER OF SESSIONS
B  PPFD OFS  BRANCH PERIOD EJECT

****************************************************************
*       TRANSCRIBE FORMATTED DATA                                *
****************************************************************

SPACE 1

PCPSCRIB ST  R1, PCPRETRN  STOW RETURN ADDRESS
BAS R8, PPHRESET  ENTER SOLO MIO MODE
MODESET MODE=PROB, KEY=NZERO BECOME MORTAL ONCE AGAIN

SPACE 1

C R1Ø, PPGF58  TEST IF TOP-OF-PAGE
BNE PCPPUT  BRANCH IF NOT
L  RØ, PCPATITL  POINT TO TITLE
PUT PPHDCB, (Ø)  TRANSCRIBE IT
MVI PPHCC, C'Ø'  DOUBLE SPACE AFTER TITLE

SPACE

PCPPUT PUT  PPHDCB, PPHCC  TRANSCRIBE DATA
MVI PPHCC, C' '  SINGLE SPACE DATA LINES
MVC PPHVRBLK(PPHDLEN-1), PPHCC  REFRESH OUTPUT AREA
BCT R1Ø, PCPSMODE  CONTINUE...
LA  R1Ø, 58  SET BEGINNING LINE COUNT

SPACE

PCPSMODE MODESET MODE=SUP, KEY=ZERO PRETEND TO BE GEORGE
BAS R8, PPHSET  ENTER UNIVERSAL MODE
L  R1, PCPRETRN  RETRIEVE RETURN ADDRESS
BR R1  PROCESS NEXT VRBLK

SPACE 2

PPHCLOSE BAS R8, PPHRESET  CLEAN UP ENVIRONMENT
PPHCLOS2 MODESET MODE=PROB, KEY=NZERO BECOME MORTAL ONCE AGAIN
PUT PPHDCB, PPGCLAM  PRINT ENVIRONMENTAL INFORMATION
CLOSE (PPHDCB)  CLEAN UP ENVIRONMENT
SR R15, R15           INDICATE SUCCESS
PR R14                 RETURN TO DUST
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* MAP EXPLICIT ROUTES                                          *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

SPACE 1
PCPMAPE R5, ATCERTP(R7) ADDRESS OF EXPLICIT ROUTE QUEUES
LA R1, PCPTITLE POINT TO HEADINGS
ST R1, PCPATITL ALTER TITLE OF DATA
SPACE
LA R9, 8Ø SET MAXIMUM Amount OF PROCESSING
LA R10, 5Ø SET BEGINNING LINE COUNT
SPACE
PCPI SLUV ICM R2, 15, 0(R5) FETCH ADDR OF EXPLICIT ROUTE BLOCK
BNE PCPCAPLS BRANCH IF IT'S AVAILABLE
SPACE
PCPNXTSA LA R5, 4(R5) POINT TO NEXT ERT POINTER
BCT R9, PCPI SLUV PROCESS IT
B PPHDOCQR PROCESS CDR ENTRIES
SPACE
PCPFORM ICM R2, 15, ERTPTR(R2) FETCH ADDRESS OF NEXT ERT
BE PCPNXTSA BRANCH IF NONEXISTENT
SPACE
PCPCAPLS CLI ERTBID(R2), ERTIDCON TEST IF 'TIS A TRUE VRBLK
BE PCPI SAGO BRANCH IF SO
BAS R8, PPHRESET ENTER SOLO MIO ROLE
WTO 'INVALID ERT' ISSUE ERROR MESSAGE
B PPHCLOS2 DEPART - TRASH MAY APPEAR IN OUTPUT
SPACE
PCPI SAGO MVI PPHCC, C' ' INITIAL BLANK
MVC PPHVRBLK(PPHDLEN-1), PPHCC CLEAR DISPLAY AREA
SPACE
LA R15, PPHVRBLK POINT TO FIRST ENTRY
USING PCPBASE, R15 ESTABLISH PPGBASE ADDRESSABILITY
SPACE
MVC PCPSA, PHSUBAH STOW SUBAREA'S NUMBER IN OUTPUT AREA
SPACE
SR R1, R1 REMOVE DETRITUS FROM GPR #1
IC R1, ERTERN(, R2) OBTAIN EXPLICIT ROUTE NUMBER
CVD R1, PPGTVICE ALTER RADIX OF VR NUMBER
MVC PCPER#, PPGPATSA SET EDIT PATTERN IN OUTPUT AREA
ED PCPER#, PPGTWICE+6 BEAUTIFY # OF THIS EXPLICIT ROUTE
SPACE
LA R1, PPGNOERS NUMBER OF CONSTANTS FOR FSM
LA R14, PPGFSMØØ FIRST FSM VALUE
PCPDOVAL CLC 0(1, R14), ERTFSM(R2) TEST IF THIS IS THE FSM STATE
BE PCPMVAL BRANCH IF SO
LA R14, PPGLNERS(R14) POINT TO NEXT ENTRY
BCT R1, PCPDOVAL PROCESS NEXT ENTRY

MVC  PCPSTAT, PPHUH    SET UNKNOWN TYPE OF STATUS
B  PCPBESO  CONTINUE PROCESSING...
EJECT

PCPMVAL  MVC  PCPSTAT, 1(R14)    SET FSM STATE IN OUTPUT AREA
SPACE

PCPBESO  MVC  PCPSTAT, 1(R14)    SET FSM STATE IN OUTPUT AREA
SPACE

MVC  PCPADDR(SA+1(7)), PPGPATLU  SET EDIT PATTERN
L  R1, ERTADDR(A, R2)  RETRIEVE NUMBER OF ADJACENT SUBAREA
CVD  R1, PPGTWICE  ALTER ITS RADIX
ED  PCPADDR(SA+1(7)), PPGTWICE+5  MAKE IT PRETTY
SPACE

MVC  PCPDEST+1(7), PPGPATLU  SET EDIT PATTERN
L  R1, ERTDSA(A, R2)  RETRIEVE NUMBER OF DESTINATION SUBA
CVD  R1, PPGTWICE  ALTER ITS RADIX
ED  PCPDEST+1(7), PPGTWICE+5  BEAUTIFY IT
SPACE

MVC  PCPHOPS, PPGPATLU  SET EDIT PATTERN
LH  R1, ERTHOPS(A, R2)  RETRIEVE NUMBER OF TRANSMISSION GRPS
CVD  R1, PPGTWICE  ALTER ITS RADIX
ED  PCPHOPS, PPGTWICE+5  BEAUTIFY IT
SPACE

BASE  R1, PCPSCRIB  TRANSSCRIBE DATA
B  PCPFORM  PROCESS NEXT EXPLICIT ROUTE
SPACE
DROP  R15  FORGET PCPBASE
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        PROCESS ENTRIES IN THE ADJACENT SSCP TABLE                *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SPACE 1

PPHDOCDR  LA  R10, 58    SET LINE COUNT
L  R7, PSAATCVT  REFRESH ADDRESS OF VTAM'S CVT
L  R5, ATCSSCPT(R7)  ADDR OF FIRST ENTRY IN ADJ SSCP TABLE
LA  R1, PCPCTITL  ADDR OF TITLE FOR CDRM DATA
ST  R1, PCPATITL  REVISE POINTER TO DATA'S TITLE
LA  R3, PPHCC  POINT TO OUTPUT AREA
USING  PCPSNI, R3  ESTABLISH PCPSNI ADDRESSABILITY
PPCDOCDR  CLI  ADJID(R5), ADJIVAL  ENSURE THAT THIS IS TRULY AN ENTRY
BNE  PCPGLCSN  BRANCH IF NOT - SOMETHING'S CHANGED
SPACE
MVI  PCPCC, C' '  SET SINGLE SPACE
MVC  PPHWRBLK(PPHDLEN-1), PPHCC  CLEAR DISPLAY AREA
SPACE

MVC  PCPADNET, ADJNETID(R5)  NETWORK IDENTIFIER TO OUTPUT AREA
MVC  PCPACDRM, ADJCDNAM(R5)  NAME OF CDRM TO OUTPUT AREA
SR  R1, R1  CLEAR A WORK REGISTER
ICM  R1, 3, ADJNETENT(R5)  FETCH NUMBER OF CDRM ENTRIES
BE  PCPPCORM  BRANCH IF NONE
C  R1, PPGF14  TEST IF NUMBER OF ENTRIES EXCEEDS 14
BNH  PCPNOK  BRANCH IF NOT
L  R1, PPGF14  LIMIT PROCESSING TO 14
PCPNOK  LA  R15,PCPENTRY  POINT TO FIRST SLOT IN OUTPUT AREA
LA  R2,ADJ ENTRY(,R5)  POINT TO NAME OF FIRST ADJ CDRM
PCPLCS  MVC  B(8,R15),0(R2)  COPY NAME TO OUTPUT AREA
LA  R2,8(,R2)  POINT TO NEXT NAME
LA  R15,10(R15)  POINT TO NEXT AVAILABLE SLOT
BCT  R1,PCPLCS  COPY NAMES TO OUTPUT AREA
PCPPCDRM  BAS  R1,PCPSCRIB  TRANSCRIBE LINE
ICM  R5,15,ADJ NEXT(R5)  POINT TO NEXT ENTRY
BNE  PCPDOCDR  THEN PROCESS IT
SPACE
DROP  R3
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        PROCESS CROSS DOMAIN RESOURCE MANAGERS                       *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SPACE 1
PCPGLCSN  L  R7,PSAATC VT  ADDRESS OF VTAM'S COMM VECTOR TABLE
LA  R10,58  SET LINE COUNT
L  R9,ATCCONFT(R7)  ADDRESS OF VTAM'S CONFIGURATION TABLE
L  R5,CONVTHAA(R9)  ADDRESS OF HOST CDRM DUMMY RDTE
LA  R1,PCPNERDT  POINT TO HEADINGS
ST  R1,PCPATITL  ALTER TITLE OF DATA
LA  R3,PPHCC  PRIME BASE FOR PCPSNI
USING  PCPNERD,R3  ESTABLISH PCPNERD ADDRESSABILITY
SPACE
PCPCLAIR  CLI  RPRENTRY(R5),RPRENTRH  TEST IF A CDRM HEADER
BNE  PCPGXRRN  BRANCH IF NOT
SPACE
CLC  RPRCURST(2,R5),PPGACTIV  TEST IF NODE IS ACTIVE
BNE  PCPGXRRN  BRANCH IF NOT
SPACE
TM  RPRBITAN(R5),8  TEST IF THIS IS THE LAST ENTRY
BO  PCPGXRRN  BRANCH IF AT END
SPACE
ICM  R2,15,RPRELEN(R5)  FETCH OFFSET TO 'RCC'
BE  PCPGXRRN  BRANCH IF AT END
AR  R2,R5  POINT TO ENTRY
SPACE 1
USING  PCPNERD,R3  ESTABLISH PCPNERD ADDRESSABILITY
SPACE
CLI  RPRENTRY(R2),RPRENTRM  TEST IF A CROSS DOMAIN RESRC MNGR
BNE  PCPGXRRN  BRANCH IF NOT
MVI  PCPNCC,C' '  SET SINGLE SPACE
MVC  PPHVRBLK(PPHDLEN-1),PPHCC  CLEAR DISPLAY AREA
SPACE
ST  R5,PPGTWICE
UNPK  PCPNADDR(9),PPGTWICE(5)  ALTER RADIX OF ADDRESS
TR  PCPNADDR,PPHT RANS-240  CONVERT ADDRESS TO EBCDIC
MVI  PCPNADDR+8,C' '  REMOVE DE DETRITUS
EJECT

CLI Ø(R5), C' ' TEST IF NAME OF CDRM MEMBER EXISTS
BL PCPNOCDR BRANCH IF NOT
MVC PCPNCDRM, Ø(R5) COPY NAME OF CDRM MEMBER TO OUTPUT SPACE

PCPNOCDR SR R1, R1 CLEAR A VOLATILE REGISTER
ICM R1, 3, X'14'(R2) FETCH NUMBER OF NCP'S SUBAREA
BE PCPNOSA# BRANCH IF UNAVAILABLE
CVD R1, PPGTWICE ALTER ITS RADIX
MVC PCPNNCP#, PPGPATLU COPY EDIT PATTERN INTO OUTPUT AREA
ED PCPNNCP#, PPGTWICE+5 STOW NCP'S SUBAREA NUMBER IN OUTPUT SPACE

PCPNOSA# L R2, X'B4' (R2), R2 FETCH POINTER TO CROSS-DOMAIN DATA
CLI X'2C'(R2), C' ' TEST FOR THE PRESENCE OF NCP NAME
BL PCPNONCP BRANCH IF UNAVAILABLE
MVC PCPNCP, X'2C'(R2) STOW NAME OF NCP IN OUTPUT AREA

PCPNONCP MVC PCPNDNET, X'14'(R2) COPY REAL NAME OF DESTINATION NETWRK
CLI Ø(R2), C' ' TEST IF NAME OF ADJ SSCP EXISTS
BL PCPNOADJ BRANCH IF NOT
MVC PCPNSSCP, X'1C'(R2) COPY NAME OF ADJACENT SSCP TO OUTPUT SPACE

PCPNOADJ MVC PCPNANET, X'34'(R2) STOW NAME OF LOCAL NULL NETWORK SPACE
LH R1, X'54'(, R2) FETCH # OF DESTINATION'S NUL SUBAREA
CVD R1, PPGTWICE ALTER ITS RADIX
MVC PCPNDSUB, PPGPATLU COPY EDIT PATTERN INTO OUTPUT AREA
ED PCPNDSUB, PPGTWICE+5 STOW NULL SUBAREA NUMBER IN OUTPT SPACE

PCPNDSUB L R1, X'60'(, R2) FETCH # OF LOCAL NULL SUBAREA
CVD R1, PPGTWICE ALTER ITS RADIX
MVC PCPNASUB, PPGPATLU COPY EDIT PATTERN INTO OUTPUT AREA
ED PCPNASUB, PPGTWICE+5 STOW NULL SUBAREA NUMBER IN OUTPT SPACE
BAS R1, PCPScrib TRANSCRIBE LINE

PCPGXRRN ICM R5, 15, RDTFORW(R5) ADDRESS OF NEXT RDTE SPACE
BNE PCPCLAIR BRANCH IF NOT AT END OF RDTE'S SPACE
DROP R3 FORGET PCPNERD
EJECT

* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
CURRENT ENTRY, FROM THE CURRENT ENTRY.

THE STATUS OF AN ENTRY IS AT OFFSET X'3C' - X'0505' = ACTIVE

SPACE
L R7, PSAATCVT ADDRESS OF VTAM'S COMM VECTOR TABLE
L R9, ATCCONFT(R7) ADDRESS OF VTAM'S CONFIGURATION TABL
L R5, CONVTHAA(R9) ADDRESS OF HOST CDRM DUMMY RDTE
LA R1, PCPLINKT POINT TO HEADINGS
ST R1, PCPATIO IT ALTERN TITLE OF DATA
LA R3, PPHEC PRIME BASE FOR PCPSNI
USING PCPLINK, R3 ESTABLISH PCPLINK ADDRESSABILITY
SPACE 1
ICM R15, 15, RDTFORW(R5) TEST IF THIS IS A DUMMY ENTRY
BE PPHCLOSE BRANCH IF SO
EJECT

PROCESS LINES WITH LINK STATIONS

LOCATE A GROUP RDT WITHIN A COMMUNICATION'S CONTROLLER
THEN PROCESS ALL LINES WITHIN IT THAT HAVE LINK STATIONS.

SPACE
PCPCMRLG CLI RPRENTRY(R5), RPRENTRN TEST IF COMMUNICATIONS CONTROLLER
BNE PCPLXRRN BRANCH IF NOT
SPACE
CLC RPRCURST(2, R5), PPGACTIVE TEST IF NODE IS ACTIVE
BNE PCPLXRRN BRANCH IF NOT
SPACE
TM RPRBITAN(R5), 8 TEST IF THIS IS THE LAST ENTRY
BO PCPLXRRN BRANCH IF AT END
SPACE
MVI PCPBGTIT, 1 SHOW THAT BIG TITLES ARE REQUIRED
MVI PPGHTICS, C' ' STOW INITIAL BLANK
MVC PPGHTICS+1(44), PPGHTICS BLANKET TIC ENTRIES WITH BLANKS
EJECT

PROCESS A GROUP'S RDT ENTRIES

SPACE
ICM R11, 15, RPRELEN(R5) FETCH OFFSET TO 'RCC'
BE PCPLXRRN BRANCH IF AT END
AR R11, R5 POINT TO ENTRY
BAS R1, PPGDOTIC LOCATE FIRST FIVE TIC'S
SPACE
PCPNJTCI CLI RPRENTRY(R11), RPRENTGP TEST IF ENTRY IS A GROUP
BNE PCPNJTRN BRANCH IF NOT
SPACE
PCPSAGP ICM R2, 15, RPRELEN(R11) FETCH OFFSET TO 'RCC'
BE PCPLXRRN BRANCH IF AT END
AR R2, R11   POINT TO ENTRY
SPACE
MVC PCPLGRUP, Ø(R11)   GROUP'S NAME TO OUTPUT
SPACE
CLI RPRENTRY(R2), RPRENTRYLN TEST IF ENTRY IS A LINE
BNE PCPNXRDT   BRANCH IF NOT
SPACE
PCPG LINE MVC PCPGLINE, Ø(R2)   COPY NAME OF LINE TO OUTPUT
CLI RLNCUA(R2), C' '   TEST IF UNIT'S NAME AVAILABLE
BL PCPNOLUA   BRANCH IF NOT
MVC PCPLUNIT, RLNCUA(R2) UNIT NAME OF LINE TO OUTPUT
SPACE
PCPNOLUA ST R2, PPGTWICE   STOW VIRTUAL ADDRESS OF LINE
UNPK PCPLADR(9), PPGTWICE(5) ALTER RADIX OF ADDRESS
TR PCPLADR, PPHTRANS-240 CONVERT ADDRESS TO EBCDIC
MVI PCPLADR+8, C' '   REMOVE DE TRASH
SPACE
ICM R1, 15, RPRELEN(R2) FETCH OFFSET TO RCC
BE PCPLXRRN   BRANCH IF NOT AVAILABLE
AR R2, R1   COMPUTE ADDRESS OF LINK STATION
SPACE
CLI RPRENTRY(R2), RPRENTRYLN TEST IF ENTRY IS INTERMEDIATE NODE
BE PCPLRLUV   BRANCH IF SO
CLI RPRENTRY(R2), RPRENTPX TEST IF ENTRY IS A SKELETAL PU
BNE PCPNXRDT   BRANCH IF NOT
SPACE
PCPLRLUV TM X'18'(R2), 1   TEST IF THIS IS A LINK STATION
BNO PCPNXRDT
EJECT
MVC PCPLINKS, Ø(R2)   COPY NAME OF LINK STATION TO OUTPUT
SPACE
CLC RPRCURST(2, R2), PPGACTIV TEST IF LINK STATION IS ACTIVE
BE PCPLSACT   BRANCH IF SO
SPACE
CLI PCPBGTIT, Ø   TEST IF A LARGE TITLE IS REQUIRED
BE PCPMINOR   BRANCH IF NOT
BAS R1, PCPJGTIT OTHERWISE TRANSCRIBE ONE
SPACE
PCPMINOR BAS R1, PCPScriB   TRANSCRIBE LINE
SPACE
PCPNXRDT A R2, RPRELEN(, R2)   POINT TO NEXT ENTRY
TM RPRBITAN(R2), 8 TEST IF THIS IS THE LAST ENTRY
BO PCPLXRRN   BRANCH IF SO
ICM R1, 15, RPRELEN(R2) ADDRESS OF NEXT RDTE
BE PCPLXRRN   BRANCH IF DONE
SPACE
CLI RPRENTRY(R2), RPRENTGP TEST IF ENTRY IS A GROUP
BNE PCPGLWRM   BRANCH IF NOT
LR R11, R2   POINT TO IT
B PCPSAGP PROCESS IT

© 2002. Xephon UK telephone 01635 33848, fax 01635 38345. USA telephone (303) 410 9344, fax (303) 438 0290.
SPACE
PCPGLWRM CLI RPREENTRY(R2),RPRENTLN TEST IF ENTRY IS A LINE
BE PCPGLINE BRANCH IF SO
B PCPNXRDT FIND NEXT RDT ENTRY
SPACE
PCPLSACT L R4,R1 INCPPT(R2) POINT TO NAME OF ADJACENT NODE
CLI RPUB8(R2),C'A' TEST IF NAME OF LINK STATION'S AVAILABLE
BL PCPAMG BRANCH IF NOT
MVC PCPLALNK,RPUB8(R2) NAME OF ADJACENT LINK STATION TO OUTPUT AREA
PCPAMG MVC PCPLANOD,RPNAME(R4) ADJACENT NODE'S NAME TO OUTPUT AREA
MVC PCPLDNET,RRNNETID(R4) COPY NODE'S NETID TO OUTPUT AREA
SPACE
ICM R6,15,RRNSFPTR(R5) POINT TO FIRST "SUFFIX" ENTRY
BE PCPPGLUV BRANCH IF UNAVAILABLE
EJECT
PCPGLUV ICM R6,15,Ø(R6) POINT TO AN SNI ENTRY
BE PCPPGLUV BRANCH IF NOT FOUND
SPACE 1
CLC PCPLDNET,8(R6) TEST FOR A MATCHING ENTRY
BNE PCPGLUV BRANCH IF ENTRIES DON'T MATCH
SPACE 1
MVC PCPLDNET,8(R6) STOW NAME OF SNI IN OUTPUT AREA
L R15,X'14'(,R6) FETCH SUBAREA'S NUMBER
CVD R15,PPGTWICE ALTER RADIX OF SUBAREA NUMBER
MVC PCPLSA,PPGPATSA SET EDIT PATTERN
ED PCPLSA,PPGTWICE+6 BEAUTIFY NUMBER OF THIS SUBAREA
SPACE
PCPPGLUV LH R1,X'14'(,R4) FETCH NUMBER OF ADJACENT SUBAREA
CVD R1,PPGTWICE
MVC PCPLASA,PPGPATSA SET EDIT PATTERN IN OUTPUT AREA
ED PCPLASA,PPGTWICE+6 BEAUTIFY # OF THIS ADJACENT SUBAREA
SPACE
CLI PCPBGTIT,Ø TEST IF A LARGE TITLE IS REQUIRED
BE PPGMIOR BRANCH IF NOT
BAS R1,PCPJGTIT OTHERWISE TRANSCRIBE ONE
SPACE
PPGMIOR BAS R1,PCPSCLIB TRANSCRIBE LINE
B PCPNXRDT PROCESS NEXT RDT ENTRY
SPACE
PCPNJTRN TM RPRBITAN(R11),8 TEST IF THIS IS THE LAST ENTRY
BO PCPLXRRN BRANCH IF SO
ICM R1,15,RPRELEN(R11) ADDRESS OF NEXT RDTE - PERHAPS
BE PCPLXRRN BRANCH IF DONE
A R11,RPRELEN(R11) SET ADDRESS OF NEXT RDTE
B PCPNJTCL PROCESS IT
SPACE
PCPLXRRN ICM R5,15,RDTFORW(R5) ADDRESS OF NEXT RDT
BNE PCPMRGL BRANCH IF NOT AT END OF RDT'S
B PPHCLOSE PROCESSING COMPLETED
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* ENTER UNIVERSAL ACCESS MODE                                  *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SPACE 1
PPHSET LH R1,PPGNETID ALTERNATE ADDRESS SPACE'S IDENTIFIER
PPHBECAT SSAR R1 USE DATA IN VTAM'S ADDRESS SPACE
SPACE 1
SAC 512 SET UNIVERSAL ACCESS MODE
SPACE 1
BR R8 RETURN TO CALLER
SPACE 2
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
* ENTER SOLO ACCESS MODE                                       *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
SPACE 1
PPHRESET L R1,PPHCASID OBTAIN ACTUAL SECONDARY ASID
SSAR R1 SET SECONDARY TO CURRENT
SPACE 1
SAC Ø ACCESS DATA ONLY WITHIN THIS ASID
SPACE 1
BR R8 RETURN TO CALLER
EJECT
***********************************************************************
* PROVIDE GENERAL INFORMATION REGARDING THIS NCP               *
* NOTE: THE PCPTITLE AREA IS REUSED AT THIS JUNCTURE.          *
***********************************************************************
SPACE
PCPJGTIT ST R1,PCPFTIT STOW RETURN ADDRESS
LA R1,PCPLINKT POINT TO HEADINGS
ST R1,PCPATITL ALTER TITLE OF DATA
MVC PCPTITLE,PPHCC PRESERVE DATA
SPACE
LA R10,7 SET PSEUDO LINE COUNTER
MVI PPHCC,C' ' SET INITIAL BLANK
MVC PPHVRBLK(PPHDLEN-1),PPHCC CLEAR DISPLAY AREA
SPACE
MVC PCPLCC(PCPNSTUF),PCPNTITL COPY TITLE TO OUTPUT AREA
BAS R1,PCPSCRIB TRANSCRIBE TITLE
SPACE
L R1,ATCHOSTA(R7) NUMBER OF HOST SUBAREA
CVD R1,PPGTWICE ALTER RADIX OF SUBAREA NUMBER
MVC PCPLHOST,PPGPATSA COPY EDIT PATTERN TO DESIGNATED SPOT
ED PCPLHOST,PPGTWICE+6 BEAUTIFY NUMBER OF THIS SUBAREA
SPACE
L R6,CONDCBBA(R9) FETCH POINTER TO VTAMLIB DCB
USING HADCB,R6 ESTABLISH DCB ADDRESSABILITY
L R6,DCBDEBAD FETCH ADDRESS OF VTAMLIB DEB
ICM R6,8,PPGXØ DESTROY HIGH-LEVEL BYTE OF ADDRESS
USING DEBASIC,C,R6 ESTABLISH DEB ADDRESSABILITY
L R6,DEBSUCBA FETCH ADDRESS OF UCB
ICM   R6, 8, PPGXØ   DESTROY HIGH-LEVEL BYTE OF ADDRESS
USING  UCBOB, R6   ESTABLISH UCB ADDRESSIBILITY
MVC   PCPLVOLI, UCBVOLI   COPY VOLUME SERIAL NUMBER TO OUTPUT
DROP  R6   FORGET ADDRESSABILITIES
SPACE
LH   R1, X'14' (, R5)   NUMBER OF HOST NCP'S SUBAREA
CVD   R1, PPGTWICE   ALTER ITS RADIX
MVC   PCPLNSA#, PPGPATSA   COPY EDIT PATTERN INTO OUTPUT AREA
ED   PCPLNSA#, PPGTWICE+6   STOW NCP'S SUBAREA NUMBER IN OUTPUT
SPACE
MVI   PCPLCC, C'Ø'   SET DOUBLE SPACE FOR DATA
MVC   PCPLRNN, RPRNAME(R5)   SAVE FOR VIEWING
MVC   PPGNCPID, RPRNAME(R5)   SAVE FOR EXTRACTING DATA FROM FEB
EJECT
ICM   R6, 15, RRNSFPTR(R5)   POINT TO FIRST SUFFIX ENTRY
BE   PPGLUVCP   BRANCH IF UNAVAILABLE
MVC   PCPLNET, 8(R6)   SET NETWORK IDENT. IN OUTPUT AREA
PPGLUVCP
MVI   PCPLREL, C'R'   DESIGNATE NUMBER AS RELEASE LEVEL
MVC   PCPLREL+1(2), RRNREL(R5)   STOW NCP'S RELEASE LEVEL IN OUTPUT
MVI   PCPLMOD, C'M'   DESIGNATE NUMBER AS MODIFICATION LEVEL
MVC   PCPLMOD+1(2), RRNMODL(R5)   STOW NCP'S MODIFICATION LEVEL
MVC   PCPCUNIT, RRNRNCUA(R5)   STOW UNIT ADDRESS OF NCP IN OUTPUT
MVC   PCPLSID, PPHSYSID   COPY NAME OF SYSTEM TO OUTPUT AREA
MVC   PCPLSSCP, PPGCLLMN   COPY NAME OF SSCP TO OUTPUT
MVC   PCPLNAME, PPHJESNM   COPY NAME OF NJE TO OUTPUT
MVC   PCPLNODE, PPHJESID   COPY NUMBER OF NJE TO OUTPUT
MVC   PCPLTIC(27), PPGHTICS   COPY NAME(S) OF TIC LINE(S) TO OUTPUT
MVC   PCPLVOL, PHPLVVOL   COPY NCPLOAD'S VOLUME SERIAL NUMBER
MVC   PCPCATVL, PHPCATVL   COPY CATALOG'S VOLUME SERIAL NUMBER
BAS   R1, PCPSCRIB   TRANSCRIBE GENERAL INFORMATION
SPACE
TM   PPGSW, 1   TEST IF FEB MAY BE READ
BO   PPGNOBOX   BRANCH TO AVOID PROVERBIAL ESTUARY
SPACE
ICM   R6, 15, PPGCPSUB   RETRIEVE ADDRESS OF PPGRDSEP
BE   PPGNOBOX   BRANCH IF UNAVAILABLE
BAS   R1, PCPSCRIB   SEPARATE WITH A BLANK LINE
SPACE
BAS   R8, PPHRESET   ENTER SOLO MIO ROLE
LR   R15, R6   POINT TO PPGRDSEP
L   R0, PPGLOVE   INDICATE THAT THIS IS A CALL REQUEST
LA   R1, PPGRAMS   POINT TO LIST OF PARAMETERS
BASSM   R14, R15   RETRIEVE NCP DATA
BAS   R8, PPHSET   BECOME OMNISCIENT ONCE AGAIN
SPACE
TM   PPGSW, 1   TEST IF READ WAS SUCCESSFUL
BO   PPGNOBOX   BRANCH IF NOT
BAS   R1, PCPSCRIB   TRANSCRIBE GENERAL INFORMATION
SPACE
PPGNOBOX
MVC   PPHCC(PPHDLEN), PCPLINKT CoPT TITE TO OUTPUT AREA

MVI PCPLCC, C'Ø' SET DOUBLE SPACE FOR SubTitle
BAS R1, PCPSCRIB TRANSCRIBE IT
SPACE
MVC PPHCC(PPHDLEN), PCPTITLE RESTORE DATA
LA R10, 5Ø SET NEW LINE COUNT
MVI PCPLCC, C'Ø' DOUBLE SPACE 1ST LINE AFTER SubTitle
L R1, PCPFTIT RETRIEVE RETURN ADDRESS
MVI PCPBGTIT, Ø SHOW THAT NO Title IS Required
BR R1 RETURN TO Caller
SPACE
DROP R3 FORGET PCPNERD
EJECT
**********************************************************************
* Locate AND Save the name(s) of lines used for                      *
* Token ring interface cards.                                      *
**********************************************************************
SPACE
PPGDOTIC ST R1, PCPFTIT STOW Return ADDRESS
LA R1, PPHTICS POINT TO HOLD AREA FOR tic's
LA R15, 5 SET maximum number of slots for tic's
LR R6, R11 ditto a general purpose register
SPACE
ICM R4, 15, RPRELEN(R6) FETCH OFFSET TO 'RCC'
BE PPGDPART branch if end
AR R4, R6 point to entry
SPACE 1
CLI RPREENTRY(R6), RPRENTGP TEST IF ENTRY IS A Group
BNE PPGDPART branch if not
SPACE
TM RGPTIC(R6), 1 TEST IF TOKEN RING
BNO PPGDPART branch if not
SPACE
PPGDOAGP ICM R4, 15, RPRELEN(R6) FETCH OFFSET TO 'RCC'
BE PPGDPART branch if end
AR R4, R6 point to entry
SPACE
CLI RPREENTRY(R4), RPRENTLN TEST IF ENTRY IS A LINE
BNE PPGDPRTD branch if not
SPACE
MVC Ø(8, R1), Ø(R4) LINE'S NAME TO OUTPUT
LA R1, 9(R1) POINT TO NEXT SLOT
BCT R15, PPGETRDT AND ATTEMPT TO FILL IT
B PPGDPART fun is done; back to work
SPACE
PPGETRDT A R4, RPRELEN(), R4 point to next entry
TM RPRB1TAN(R4), 8 TEST IF THIS IS THE LAST ENTRY
BO PPGDPART branch if so
ICM R0, 15, RPRELEN(R4) ADDRESS OF NEXT RDTE
BE PPGDPART branch if done
SPACE

CLI   RPRENTRY(R4),RPRENTGP TEST IF ENTRY IS A GROUP
BNE   PPGETRDT BRANCH IF NOT
LR    R6,R4   POINT TO IT
TM    RGPTIC(R6),1 TEST IF TOKEN RING
BNO   PPGETRDT BRANCH IF NOT
   B     PPGDOAGP PROCESS IT
   SPACE
PPGDPART L   R1,PCPFTIT RETRIEVE RETURN ADDRESS
BR    R1 UTILIZE IT
EJECT
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*        CONSTANTS AND OTHER JUNK                                    *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
   SPACE 1
   PPGTWICE DS    D
   PCPFTIT DS    F
   PCPATITL DC    A(PPHTITLE) DATA'S TITLE
   PCPRETRN DC    F'0' VTAM'S ASID
   PPGNETID DC    F'0' VTAM'S ASID
   PPGF58 DC    F'58'
   PPGF14 DC    F'14'
   PPGH4 DC    H'4'
   PPGACTIV DC    XL2'0505'
   PPGXØ DC    X'00'
   SPACE
   PPGNCPID DC    CL8'NCPE3'
   PPGPSWD DC    AL1(4),CL8'CLAM'
   PPGISTIN DC    CL8'ISTINMØ1' PROGRAM'S NAME EXECUTED BY VTAM PROC
   PPGNCPPLD DC    CL8'NCLOAD'
   PPGRSET DC    CL4'RSET'
   PPGRSIT DC    CL4'PRIS'
   PPGSEC DC    CL4'SEC'
   PPGOPENC DC    CL4'OPEN'
   PPGHELDC DC    CL4'HELD'
   SPACE 3
*        DEFINITIONS OF ENTRIES IN VTAM'S CONFIGURATION TABLE
   CONIDENT EQU   X'100' OFFSET WITHIN CONFT TO VTAM'S JOB ID
   CONDCBBA EQU   X'5C' OFFSET WITHIN CONFT TO VTAMLIB DCB
   CONAREAA EQU   X'E0' OFFSET TO CONFT AREA'S POINTER
   CONLIST EQU   20 OFFSET TO LIST=ID OPERAND ON START
   SPACE
*        DEFINITIONS OF ENTRIES IN VTAM'S COMMUNICATIONS VECTOR TABLE
   ATCVTTLVL EQU   0 OFFSET WITHIN ATC TO VTAM'S RELEASE
   ATCVRNDX EQU   X'69C' OFFSET WITHIN ATC TO VRIT ENTRIES
   ATCASID EQU   X'756' OFFSET WITHIN ATC TO VTAMLIB DCB
   ATCERTP EQU   X'5D0' OFFSET WITHIN ATC TO ERT ENTRIES
   ATCSSCPT EQU   X'554' OFFSET WITHIN ATC TO ADJ SSCP TABLE
   ATCHOSTA EQU   1172 OFFSET TO THIS HOST'S SUBAREA NUMBER
   ATCNQNAM EQU   2412 OFFSET TO NETID.SSCPNAME
EJECT

* DEFINITIONS OF ENTRIES IN ADJACENT SSCP TABLE

\[
\begin{align*}
\text{ADJID} & \quad \text{EQU} \quad 0 \quad \text{OFFSET TO CONTROL BLOCK ID}.
\text{ADJIDVAL} & \quad \text{EQU} \quad \text{X}'77' \quad \text{CONTROL BLOCK'S IDENTIFIER}
\text{ADJNEXT} & \quad \text{EQU} \quad 8 \quad \text{OFFSET TO POINTER TO NEXT ENTRY}
\text{ADJNETID} & \quad \text{EQU} \quad 16 \quad \text{OFFSET TO DESTINATION'S NETWORK ID}
\text{ADJCDNAM} & \quad \text{EQU} \quad 24 \quad \text{OFFSET TO CDRM NAME}
\text{ADJENTRY} & \quad \text{EQU} \quad 32 \quad \text{OFFSET TO NAME OF FIRST ADJCDRM}
\end{align*}
\]

SPACE 3

* DEFINITIONS OF ENTRIES IN EXPLICIT ROUTE TABLE

\[
\begin{align*}
\text{ERTBID} & \quad \text{EQU} \quad 0 \quad \text{OFFSET TO ERT CONTROL BLOCK ID}.
\text{ERTIIDCON} & \quad \text{EQU} \quad \text{X}'14' \quad \text{ERT CONTROL BLOCK IDENTIFIER}
\text{ERTPTR} & \quad \text{EQU} \quad 4 \quad \text{OFFSET TO POINTER TO NEXT ERT}
\text{ERTERN} & \quad \text{EQU} \quad 8 \quad \text{OFFSET TO EXPLICIT ROUTE NUMBER}
\text{ERTFSM} & \quad \text{EQU} \quad 9 \quad \text{OFFSET TO FINITE STATE MACHINE}
\text{ERTHOPS} & \quad \text{EQU} \quad 20 \quad \text{OFFSET TO # OF TRANSMISSION GROUPS}
\text{ERTADJSA} & \quad \text{EQU} \quad 24 \quad \text{OFFSET TO ADJACENT SUBAREA NUMBER}
\text{ERTDSA} & \quad \text{EQU} \quad 28 \quad \text{OFFSET TO DESTINATION SUBAREA NUMBER}
\end{align*}
\]

SPACE

* DEFINITIONS OF ENTRIES IN VIRTUAL ROUTE BLOCK TABLE

\[
\begin{align*}
\text{VRBVRFSM} & \quad \text{EQU} \quad 0 \quad \text{STATE OF VIRTUAL ROUTE}
\text{VRBTYPE} & \quad \text{EQU} \quad 0 \quad \text{OFFSET TO VRBLK TYPE FIELD}
\text{VRBFCSM} & \quad \text{EQU} \quad 1 \quad \text{FLOW CONTROL-FINITE STATE MACHINE}
\text{VRBTP} & \quad \text{EQU} \quad 2 \quad \text{TRANSMISSION PRIORITY INDICATOR}
\text{VRBVRN} & \quad \text{EQU} \quad 2 \quad \text{NUMBER OF VIRTUAL ROUTE}
\text{VRBIXCHN} & \quad \text{EQU} \quad 4 \quad \text{CHAIN POINTER FOR NEXT VRBLK}
\text{VRBCID} & \quad \text{EQU} \quad 5 \quad \text{VRBLK TYPE IDENTIFIER}
\text{VRBADJSA} & \quad \text{EQU} \quad 8 \quad \text{ADJACENT SUBAREA NUMBER}
\text{VRBPRI} & \quad \text{EQU} \quad \text{X}'80' \quad \text{PRIMARY ROUTE INDICATOR}
\text{VRBFS} & \quad \text{EQU} \quad 88 \quad \text{OFFSET TO FIRST STATUS AREA}
\text{VRBDRC} & \quad \text{EQU} \quad 232 \quad \text{INITIAL EXPLICIT ROUTE NUMBER}
\text{VRBDSTSA} & \quad \text{EQU} \quad 236 \quad \text{DESTINATION SUBAREA}
\text{VRBESC} & \quad \text{EQU} \quad 24 \quad \text{OFFSET TO COUNT OF SESSIONS FOR VR}
\end{align*}
\]

SPACE 3

* EDIT PATTERNS

SPACE 1

\[
\begin{align*}
\text{PPGPATSA} & \quad \text{DC} \quad \text{XL4'40202120'}
\text{PPGPATLU} & \quad \text{DC} \quad \text{XL7'4020206202120'}
\text{PCPBGTIT} & \quad \text{DC} \quad \text{X'00'}
\end{align*}
\]

SPACE

\[
\begin{align*}
\text{PPGSW} & \quad \text{DC} \quad \text{X'02'} \quad \text{VTAM RESPONSES ARE NOT PARROTED}
\text{PPHDCB} & \quad \text{DCB} \quad \text{LRECL=137, BLKSIZE=137, DSORG=PS, MACRF=PM, RECFM=FA,}
\text{DDNAME=SYSPRINT, DCBE=PHDCBGL}
\end{align*}
\]

SPACE 1

\[
\begin{align*}
\text{PPHDCBGL} & \quad \text{DCBE}
\text{PPHGAREA} & \quad \text{DS} \quad \text{F}
\text{PPHCASID} & \quad \text{DS} \quad \text{F}
\text{PPGFSMST} & \quad \text{DS} \quad \text{H}
\end{align*}
\]
PPHONE DC 3F'1'
PPHSYSID DS CL4

SPACE

PPGCATID DC F'0'
PPHCNAME DC CL8'CATALOG'

SPACE

PPHJESNM DC CL8'?'
PPHVVOL DC CL6'?'
PPPCATVL DC CL6'?'

SPACE

PPHJES2 DC CL4'JES2'

PACE

PPHJESI D DC CL4'JES2'

PACE

PPHJES2 DC CL4'JES2'

PACE

PPHJESID DC XL4'4Ø2Ø212Ø'
PATH256 DC H'256'

PACE

PPHSUBAH DC XL4'4Ø2Ø212Ø'

PACE

PPGMVPPG MVC PPGNMNET(*-*),ATCNQNAM(R7) ===> EXEC ONLY <===
PPGETAID MVC PPGPSWD+1(*-*),2(R1) ===> EXEC ONLY <===

PACE

USING HCCT,R3 ESTABLISH HCCT ADDRESSABILITY
PCXFIONA MVC PPHJESNM(*-*),CCTNDENM **** EXEC ONLY ****

PACE

DROP R3 FORGET HCCT SPACE 2

PACE

PPGHTICS DC 5CL9'

PACE

PPHTRANS DC C'0123456789ABCDEF'

PACE

PPGFSMØØ DC X'00', CL6'RESET' RESET - NOT DEFINED

PACE

PPGLNERS EQU (*-PPGFSMØØ)

PACE

PPGFRS DC CL8'RESET'

PACE

PPFGFIN DC CL8'INACTIVE'

PACE

PPGFPI DC CL8'PEND-ACTV'

PACE

PPGFPA DC CL8'PEND ACTV'

PACE

PPGFAC DC CL8'PEND ACTV'

PACE

PPGFIF DC CL8'PEND ACTV'

PACE

PPGFRS DC CL8'RESET'

PACE

PPFGFIN DC CL8'INACTIVE'

PACE

PPGFPI DC CL8'PEND ACTV'

PACE

PPGFPA DC CL8'PEND ACTV'

PACE

PPGFAC DC CL8'PEND ACTV'

PACE

PPGFIF DC CL8'PEND ACTV'

PACE

PPGHTICS DC 5CL9'

PACE

PPHTRANS DC C'0123456789ABCDEF'

PACE

PPGFSMART DC X'00', CL6'RESET' RESET - NOT DEFINED

PACE

PPGLNERS EQU (*-PPGFSMART)

PACE

PPGNOERS EQU (((-PPGFSMART)/(PPGLNERS))

PACE

PPGHUH DC CL6'??????' UNKNOWN

PACE

PPGVRVAL DC A(PPGFRS)

PACE

PPGVFIN DC A(PPFGFIN)

PACE

PPGFPI DC A(PPGFPI)

PACE

PPGFPA DC A(PPGFPA)

PACE

PPGFAC DC A(PPGFAC)

PACE

PPGFIF DC A(PPGFIF)

PACE

PPGFRS DC CL8'RESET'

PACE

PPFGFIN DC CL8'INACTIVE'

PACE

PPGFPI DC CL8'PEND ACTV'

PACE

PPGFPA DC CL8'PEND ACTV'

PACE

PPGFAC DC CL8'PEND ACTV'

PACE

PPGFIF DC CL8'PEND ACTV'

PACE

PPGHTICS DC 5CL9'

PACE

PPHTRANS DC C'0123456789ABCDEF'

PACE

PPGFSMART DC X'00', CL6'RESET' RESET - NOT DEFINED

PACE

PPGLNERS EQU (*-PPGFSMART)

PACE

PPGNOERS EQU (((-PPGFSMART)/(PPGLNERS))

PACE

PPGHUH DC CL6'??????' UNKNOWN
PPGFAC DC CL8'ACTIVE'
PPGFI F DC CL8'DACTVR-F'
SPACE
CONVTHAA EQU X'94'  POINTER TO VTAM RDT HEADER AREA
RGP Tic EQU X'7B'  POINTER TO VTAM RDT HEADER AREA
SPACE 1
CAXCHN EQU 4
CAXMCT EQU 4
CAXFLGS EQU 8
CAXUCBA EQU X'1C'
CBSCAXCN EQU 20
SPACE
ATCCONFT EQU X'440'  POINTER TO VTAM CONFIG TABLE
SPACE 3
LTORG
EJECT
* NAMES OF RDTE OFFSETS
SPACE 1
RPRNAME EQU 0  NAME OF RDTE ET AL
RPRENTRN EQU 1  ENTRY IS A PU 4/5
RPENTRY EQU X'0D'  ENTRY TYPE
RPRENRH EQU X'06'  CDRM HEADER
RPRELEN EQU X'24'  LENGTH OF CURRENT ENTRY
RPRECURST EQU X'3C'  CURRENT-STATE BYTES(SEE FSM)
RPRENTRE EQU X'40'  ENTRY IS A CDRM
RPRENTAN EQU X'41'  FLAG BITS
RPRENLN EQU X'50'  LINE ENTRY
RPRENTGP EQU X'30'  GROUP ENTRY
RPRENTPX EQU X'72'  SKELETAL PHYSICAL UNIT (PUX)
RPRENTIN EQU X'82'  INTERMEDIATE NODE
SPACE
RDTFORW EQU X'70'  POINTER TO NEXT SEGMENT
RLNCUA EQU X'90'  OFFSET WITHIN RLN TO CHANNEL UNITADR
RPUB8 EQU X'B8'  OFFSET WITHIN RPU TO NAME OF ADJ LNK
RINNCPT EQU X'EC'  " " RIN TO RIN'S NAME ON ID STMNT
RRNNETID EQU X'120'  " " RRN TO NET ID OF DUMMY NCP
SPACE
RRNRELL EQU X'EB'  RELEASE LEVEL OF NCP
RRNMODL EQU X'ED'  NCP'S MODIFICATION LEVEL
RRRNCUA EQU X'104'  ACTUAL ADDRESS OF NCP'S CHANNEL UNIT
RRNSFPTR EQU X'128'  POINTER TO FIRST SUFFIX ENTRY
EJECT
***********************************************************************
* LIST FORM OF ENVIRONMENTAL INFORMATION'S WTO                      *
***********************************************************************
SPACE
PATWTOOP WTO '02022001  15112000  1234  12.12  IEANUC00  HCD - ',MF=L
PATPRODN EQU PATWTOOP+4,8
PATPRODI EQU PATWTOOP+4+10,8
PATMODEL EQU PATWTOOP+4+10+10,5
PATNUMB  EQU  PATWTOOP+4+10+10+6, 2
PATLEVEL EQU  PATWTOOP+4+10+10+6+3, 2
PATNUC  EQU  PATWTOOP+4+10+10+6+3+11, 1
PATHCD  EQU  PATWTOOP+4+10+10+6+3+11+9, 2

* TITLE USED FOR MAPPING OF VIRTUAL ROUTES

SPACE 1

PPHTITLE DC  CL137' '
ORG  PPHTITLE
DC  '1'
DC  CL5' SA #'
DC  CL5' DEST'
DC  CL5' VR #'
DC  CL7' ADJSUB'
DC  CL5' ER #'
DC  CL5' PRIO'
DC  CL4' RTE'
DC  CL9' VR-STATE'
DC  CL5' STAT'
DC  CL10' LU-COUNT'
DC  CL5' PRIO'
DC  CL4' RTE'
DC  CL9' VR-STATE'
DC  CL5' STAT'
DC  CL10' LU-COUNT'
DC  CL5' PRIO'
DC  CL4' RTE'
DC  CL9' VR-STATE'
DC  CL5' STAT'
DC  CL10' LU-COUNT'
DC  CL2' '

ORG
EJECT

* TITLE USED FOR ENVIRONMENTAL INFORMATION

SPACE 1

PPGCLAM DC  CL137' '
ORG  PPGCLAM
DC  '1'
DC  C' NETWORK IDENTIFIER IS: '

PPGNMNET DC  CL8' '
DC  CL2' '
DC  C' NAME OF SSCP IS: '

PPGCLLMN DC  CL8' '
DC  CL2' '
DC  C' VTAM' S NAME IS: '

PPHJNAME DC  CL8' VTAM' S HANDLE
DC  CL2' '
DC  C' VTAM' S VERSION IS: '

PPHVTVER DC  CL8' VTAM' S HANDLE
DC  CL2' '

DC   C' STARTED: LIST='PPHGLIST DC   CL2' VTAM'S HANDLE
ORG
EJECT
*       TITLE USED FOR ENTRIES IN THE ADJACENT SSCP TABLE
SPACE 1
PCPCTITL DC   CL137' ORG
DC   C' 1'
DC   CL10' DEST-NETID'
DC   C' '
DC   CL8' CDRMNAME'
DC   C' '
DC   C' NAMES OF ADJACENT CDRMS'
ORG
EJECT
PCPNITTL DC   C' 1'
*       TITLE USED FOR GENERALIZED INFORMATION REGARDING AN LPAR
SPACE 1
DC   CL4' SID'
DC   C' '
DC   CL8' NET-ID.'
DC   C' '
DC   CL8' NCP-LOAD'
DC   C' '
DC   CL6' NCP-SA'
DC   C' '
DC   CL6' HOSTSA'
DC   C' '
DC   CL8' VERSION'
DC   C' '
DC   CL8' SCPNAME'
DC   C' '
DC   CL8' SSCPNAME'
DC   C' '
DC   CL8' NODENAME'
DC   C' '
DC   CL8' NODE'
DC   C' '
DC   CL4' UNIT'
DC   C' '
DC   CL7' VTMLIB'
DC   C' '
DC   CL7' NCPLOAD'
DC   C' '
DC   CL7' CAT-VOL'
DC   C' '
DC   CL27' TOKEN RING LINE(S)'
PCPNSTUF EQU   *-PCPNITTL
EJECT
*       TITLE USED FOR GENERALIZED INFORMATION REGARDING AN SNI
SPACE 1
PCPLINKT DC CL137 ' '
ORG PCPLINKT
DC C '1'
DC CL8'GROUP'
DC C ' '
DC CL8'LINENAME'
DC C ' '
DC CL8'LIN-STA'
DC C ' '
DC CL4'ADDR'
DC C ' '
DC CL7'SUBAREA'
DC C ' '
DC CL6'DESTSA'
DC C ' '
DC CL8'ADJ-NODE'
DC C ' '
DC CL8'ADJ-LKSTA'
DC C ' '
DC CL8'DEST-NET'
DC C ' '
DC CL8'CBLKADDR'
ORG
EJECT

PCPNERDT DC CL137 ' '
* TITLE USED FOR CROSS-DOMAIN RESOURCE MANAGERS
SPACE 1
ORG PCPNERDT
DC C '1'
DC CL8'NCP-LOAD'
DC C ' '
DC CL7'NCP-SA#'
DC C ' '
DC CL8'CDRMNAME'
DC C ' '
DC CL8'ADJ-SSCP'
DC C ' '
DC CL8'ADJ-NET'
DC C ' '
DC CL8'DEST-NET'
DC C ' '
DC CL7'SUBAREA'
DC C ' '
DC CL7'DEST-SA'
DC C ' '
DC CL8'CBLKADDR'
ORG
EJECT
* TITLE USED FOR MAPPING EXPPLICIT ROUTES
SPACE 1
PCPTITLE DC CL137 ' ' ORG PCPTITLE DC C'1' DC CL4' SA # ' DC C' ' DC CL4' ER # ' DC C' ' DC CL6' STATUS' DC C' ' DC CL8' ADJ-SUBA' DC C' ' DC CL8' DESTSUBA' DC C' ' DC CL7' HOPS' ORG EJECT * TITLE USED FOR MAPPING VIRTUAL ROUTES SPACE 1 PPHCC DC C'1' PPHVRBLK EQU * PPHSUBA# DC CL4' ' DC C' ' PPHDEST# DC CL4' ' DC C' ' PPHVR# DC CL4' VR # ' DC CL3' ' PPHAJSA# DC CL4' ' DC C' ' PPHER# DC CL4' ' DC C' ' PPHTP1 DC CL4' TP # ' DC C' ' PPHTP2 DC CL4' TP # ' DC C' ' PPHTP2 DC CL4' TP # ' DC C' ' PPHRT1 DC CL3' PRI ' DC C' ' PPHRT2 DC CL3' PRI ' DC C' ' PPHSTAT1 DC CL8' VR STATE' DC C' ' PPHSTAT2 DC CL8' VR STATE' DC C' ' PPHFCFS1 DC CL4' STAT' DC C' ' PPHFCFS2 DC CL4' STAT' DC C' ' PPHLUS1 DC CL8' SESS-CNT' DC CL2' ' PPHLUS2 DC CL8' SESS-CNT' DC CL2' '
PCPLNET DC CL8' '
DC C' '
PCPLRRN DC CL8' '
DC CL3' '
PCPLNSA# DC CL4' '
DC CL3' '
PCPLHOST DC CL4' '
DC C' '
PCPLREL DC CL3' '
PCPLMOD DC CL3' '
DC CL3' '
PCPLSSCP DC CL8' '
DC C' '
PCPLNAME DC CL8' '
DC C' '
PCPLNODE DC CL4' '
DC C' '
PCPCUNIT DC CL4' '
DC C' '
PCPLVOLI DC CL6' '
DC CL2' '
PCPLVVOL DC CL6' '
DC CL2' '
PCPCATVL DC CL6' '
DC CL2' '
PCPLTIC DC 3CL9' '
SPACE
ORG PCPLSID
PCPLGRUP DC CL8'GROUP'
DC C' '
PCPLINE DC CL8'LINENAME'
DC C' '
PCPLINKS DC CL8'LINK-STA'
DC C' '
PCPLUNIT DC CL4'ADDR'
DC CL4' '
PCPLSA DC CL4'SA-#'
DC CL3' '
PCPLASA DC CL4'DSA#'
DC C' '
PCPLANOD DC CL8'ADJ-NODE'
DC C' '
PCPLALNK DC CL8'ADJ-LKSTA'
DC C' '
PCPLDNET DC CL8'DEST-NET'
DC C' '
PCPLADR DC CL8'ADDRESS'
TITLE 'ESA CONTROL BLOCKS'
***********************************************************************
*                        GENERATE REQUIRED OS CONTROL BLOCKS             *
***********************************************************************
**********************************************************************************************************
SPACE 1
DSECT
IEFUCBOB
SPACE 1
PPGTIOT DSECT ,
IEFTIOT1
SPACE 1
IEESMCA
SPACE 1
IKJTCB TASK CONTROL BLOCK
SPACE 1
IHAXTLST EXTENT LIST
SPACE 1
IHARB REQUEST BLOCK
SPACE 1
IHACDE CONTENTS DIRECTORY ENTRY
SPACE 1
IHASDWA SYSTEM DIAGNOSTIC WORK AREA
SPACE 1
IHAPSA PREFIXED SAVE AREA (LOW CORE)
SPACE
IHAAASC ADDRESS SPACE CONTROL BLOCK
SPACE
IHAAAXB ADDRESS SPACE CONTROL BLOCK XTENSION
SPACE
IHAAASVT ADDRESS SPACE CONTROL BLOCK XTENSION
SPACE
IEFJESCT JES CONTROL TABLE
SPACE 1
IEZDEB DATA EXTENT BLOCK
SPACE 1
DCBD DATA CONTROL BLOCK FOR -SAM AND BPAM
SPACE 1
IEFJSCVT JES CONTROL TABLE
SPACE 1
CVT DSECT=YES, PREFIX=YES ANCHOR AFTER ANCHOR
TITLE 'JES2 CONTROL BLOCKS'
**********************************************************************************************************
* GENERATE REQUIRED JES2 CONTROL BLOCKS *
**********************************************************************************************************
SPACE 1
COPY $HASPGBL
&SGI HASU(1) SETB 0
&XITMASK(6) SETA 1
SPACE
$BUFFER
SPACE
$HASPEQU
SPACE
Command line ftp

This article follows on from last issue's round-up of the most common tools used to ftp to the mainframe ('Accessing the mainframe from ftp software products', TCP/SNA Update 47, September 2002, pp 39-59). Here, we look at the command line ftp that still comes with Windows XP, virtually unchanged over the years.

Despite its ‘so DOS’ feel, command line ftp is undoubtedly a
viable ftp alternative, especially when you have only a single file to move between mainframe and workstation. But, arguably, its greatest value is as an educational tool. With it, you can gain a detailed understanding of exactly how the z/OS ftp server works, and of what the developers of the ftp clients described in the last issue were up against.

RUNNING FTP

Like all other command line programs, ftp is best initiated from what Windows NT/2000/XP calls the Command Line, and Windows 9x/ME calls MS-DOS. From the Windows XP Start button menu, select All Programs/Accessories/Command Prompt. ftp will not work directly from Start/Run, though you can run CMD from there to initiate the Command Prompt.

Type FTP followed by a blank, and then either the mainframe’s IP host name or numeric IP address. You’ll then be prompted to enter User and password, typically your TSO ID and password as administered by RACF or equivalent. To replicate the simple example in the ‘ftp on the mainframe’ section of last issue’s article, you would then type:

```
quote syst
dir
quit
exit
```

The complete session would look as follows:

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Armand Minet>ftp 209.217.251.162
Connected to 209.217.251.162.
220 Connection will close if idle for more than 50 minutes.
User (209.217.251.162:(none)): aminet
331 Send password please.
Password:
230 AMINET is logged on. Working directory is "AMINET."
ftp> quote syst
215 MVS is the operating system of this server. FTP Server is running on
OS/390 UNIX System Services.
```
ftp> dir
200 Port request OK.
125 List started OK

Volume Unit    Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
ISPW89 3390  2002/06/04  1  15  FB   80 27920  PO  ACTIVE.ASM
ISPW89 3390  2002/06/04  1  1  VB  255 27998  PO  ACTIVE.CLIST
ISPW89 3390  2002/06/23  4  13  FB   80 27920  PO  ACTIVE.CNTL
ISPW89 3390  2002/06/03  2  6  FB  80 27920  PO  ACTIVE.COBOL
ISPW89 3390  2002/06/23  8  36  FB  80 27920  PO  ACTIVE.DATA
ISPW89 3390  2002/06/23  2  2  VB  255 27998  PO  ACTIVE.EXEC
ISPW89 3390  2002/06/23  1  1  VBA 133 27998  PS  ACTIVE.LIST
ISPW81 3390  2002/06/03  2  11  U   27998 27998  PO  ACTIVE.LOAD
ISPW81 3390  2002/06/30  1  15  FB   80 27920  PO  ACTIVE.MACLIB
ISPW89 3390  2002/02/22  1  1  FB  80 27920  PO  ACTIVE.PLI
ISPW89 3390  2001/10/21  1  1  FB  80 27920  PO  ARCHIVE.CNTL
ISPW89 3390  2002/02/02  1  7  VB  255 27998  PO  ARCHIVE.EXEC
ISPW89 3390  2002/06/23  2  8  FB  80 3120  PO  ISPF.ISPPROF

250 List completed successfully.
ftp: 1449 bytes received in 0.305 seconds 4.81Kbytes/sec.
ftp> quit
221 Quit command received. Goodbye.

C:\Documents and Settings\Armand Minet>exit

VIEW OF THE MAINFRAME FILE SYSTEM

As the response to the DIR command shows, the initial view is of just the datasets with the High Level Qualifier (HLQ) matching the userid with which you log on. Each DataSet Name (DSN) is shown without its HLQ — add your user ID as HLQ before processing.

To emulate the hierarchical Unix style directory file structure on which ftp is based, you can navigate through the DSN qualifiers. If you’re familiar with abbreviated DOS directory commands, you’ll be at home with the ftp client. In the example below, change directory is CD, and is shown going to the next qualifier for all catalogue entries beginning ‘AMINET.ACTIVE.’:

ftp> cd active
250 "AMINET.ACTIVE." is the working directory name prefix.
ftp> dir
200 Port request OK.
125 List started OK

Volume Unit    Referred Ext Used Recfm Lrecl BlkSz Dsorg Dsname
ISPW89 3390  2002/06/04  1  15  FB   80 27920  PO  ASM
ISPW89 3390  2002/06/04  1  1  VB  255 27998  PO  CLIST
A Partitioned DataSet (PDS), indicated by the Dsorg field value of PO, is yet another level of the directory hierarchy:

```
ftp> cd cobol
250 "AMINET.ACTIVE.COBOL" partitioned data set is working directory
ftp> dir
200 Port request OK.
125 List started OK

Name     VV.MM   Created       Changed      Size  Init   Mod   Id
CALLUID   01.01 2002/05/28 2002/05/28 10:36    12    12     Ø AMINET
CALLUJC   01.02 2002/05/30 2002/06/03 10:14    12    12     Ø AMINET
DB2USQL   01.03 2002/05/30 2002/06/03 15:07    13    12     Ø AMINET
SYSUPARM  01.05 2002/05/30 2002/06/03 10:02    19    12     Ø AMINET
```

250 List completed successfully.
ftp: 356 bytes received in 0.04Seconds 8.90Kbytes/sec.

Note how the z/OS ftp server responds to the CD command, indicating that you’ve entered a PDS. Note also how the fields displayed by the DIR command have changed from the previous examples, now that you’re inside a PDS.

Even CD .. is supported, moving up one level of the directory hierarchy. Unfortunately, however, you can’t get high enough to see the full DSN (with HLQ):

```
ftp> cd ..
250 "AMINET.ACTIVE." is the working directory name prefix.
ftp> cd ..
250 "AMINET." is the working directory name prefix.
ftp> cd ..
250 "" is the working directory name prefix.
ftp> dir
200 Port request OK.
550 No data sets found.
```

A SIMPLE SESSION
The simplest and most common ftp sessions transfer a single
text file either from mainframe to workstation or vice versa. Once you’ve started command line ftp, connected to the mainframe host and logged on, select the correct directory on both workstation (LCD) and host (CD), set the transfer type to Text (ASCII), transfer the file (GET or PUT) and then terminate the ftp session (QUIT) and the Command Prompt session (EXIT).

```plaintext
230 AMINET is logged on. Working directory is "AMINET.".
ftp>lcd "My Documents"
Local directory now C:\Documents and Settings\Armand Minet\My Documents.
ftp> cd activecntl
250 "AMINET.ACTIVE.CNTL" partitioned data set is working directory
ftp>ascii
200 Representation type is Ascii NonPrint
ftp>get jobcard
125 Sending data set AMINET.ACTIVE.CNTL(JOBCARD) FIXrecfm 80
250 Transfer completed successfully.
ftp: 246 bytes received in 0.03Seconds 8.20Kbytes/sec.
ftp>cd..
250 "AMINET.ACTIVE." is the working directory name prefix.
ftp>cdasm
250 "AMINET.ACTIVE.ASM" partitioned data set is working directory
ftp>put jobcard
125 Storing data set AMINET.ACTIVE.ASM(JOBCARD)
250 Transfer completed successfully.
ftp: 246 bytes sent in 0.00Seconds 246000.00Kbytes/sec.
ftp>quit
221 Quit command received. Goodbye.
C:\Documents and Settings\Armand Minet>exit
```

TWO COMMAND LANGUAGES

The most confusing thing about ftp, especially this command line version, is that there are ftp client commands and server commands. Because the two sets have very few commands in common, many ftp clients attempt to simplify matters by accepting both, without forcing the user to differentiate. This is not the case with command line ftp. Here, client commands are entered normally, but server commands must be preceded by the QUOTE command. For example, as shown in the first sample session in this article, SYST is a server command:
Use the HELP command to display the client ftp commands, and REMOTEHELP to display the server commands. The commands, which may be abbreviated, are as follows:

```
!               delete          literal         prompt          send
?               debug           ls              put             status
append          dir             mdelete         pwd             trace
ascii           disconnect      mdir            quit            type
bell            get             mget            quote           user
binary          glob            mkdir           recv            verbose
bye             hash            mls             remotehelp
cd              help            mput            rename
close           lcd             open            rmdir
```

```
ftp> remotehelp
214-The server-FTP commands are:
214-ABOR,*ACCT,*ALLO, APPE, CDUP, CWD, DELE, HELP, LIST, MKD, MODE
214-NLST, NOOP, PASS, PASV, PORT, PWD, QUIT, REIN, REST, RETR, RMD
214-RNFR, RNTO, SITE,*SMNT, SYST, STAT, STOR, STOU, STRU, TYPE, USER
214-The commands preceded by '*' are not implemented
214-The data representation type may be ASCII, EBCDIC or IMAGE, or may be
214-one of the following Double Byte Character Sets:
214-EBCDIC IBM Kanji, Shift JIS Kanji, Extended Unix Code Kanji,
214-JIS 1983 Kanji, JIS 1978 Kanji, Hangeul, Traditional Chinese,
214-or Korean Standard Code KSC-5601, 1989 Version
214-The data structure may be File or Record.
214-The mode may be Stream, Block, or Compressed.
214-If the connection to this server is inactive for more than
214-3000 seconds, the connection will be closed.
214-Data set names are represented as either a valid MVS data set name
214-or a valid HFS file name.
214-For information about a particular command, type
214 HELP SERVER command or QUOTE HELP command
```

**COMMANDS**

As a command line utility, ftp basically requires you to have access to some sort of Command Reference documentation. As we saw above, you can use the HELP and REMOTEHELP commands to list the available ftp client (workstation) and server (mainframe) commands. You can also get specific help about a particular ftp server command:

```
ftp> quote help syst
214 SYST: Returns the name of server's operating system
```
Likewise, for an ftp client command:

```
ftp> help dir
dir               List contents of remote directory
```

However, this provides only a description of the purpose of the command, not an explanation of any parameters.

**PARAMETER DETAILS**

Windows XP’s Help and Support Centre is the best source of usage details for both command line ftp and its client-based commands. However, it is listed in neither the Contents nor Index. Use the Search field in the upper left corner, type FTP, and hit the green right arrow just to the right of the Search field. There, you’ll see six suggested topics. Choose one of the following:

- ftp subcommands – all ftp client commands and their parameters.
- ftp – the parameters you can type on the same line as FTP at the DOS C:> prompt.

The z/OS ftp client commands are well documented in *z/OS Communications Server IP User’s Guide and Commands*. For z/OS 1.4, the order number is SC31-8780-02 and the URL is:

http://publibz.boulder.ibm.com/cgi-bin/bookmgr_OS390/BOOKS/F1A1B920

Each command is individually documented in Chapter 5. Just add /5.0 to the URL above to get there directly.

IBM doesn’t document the commands for the z/OS ftp server, beyond the output of the HELP command, as shown in response to the REMOTEHELP client command above. Note how some commands are preceded by an asterisk, with a note indicating that they’re not implemented. In fact, the z/OS ftp server supports most of the commands defined in the Internet standard for ftp servers that can be found at ftp://ftp.isi.edu/in-notes/rfc959.txt

The best place to start is Section 4.1, ftp commands, which begins on page 24. Parameters for each command are explained
Setting up a corporate portal – getting it right

The first article in this two-part series on setting up a corporate portal (TCP/SNA Update 47, September 2002, pp 32-38) concentrated on security issues. Here, we focus on the platform and software choices available, and on the TCP/SNA application integration options.

The platform you select for your portal will inevitably dictate many of its eventual operational characteristics – including performance, scalability (the number of concurrently active users that can realistically be supported), resilience, and upgradability. However, if you intend to use a portal server, the choice of portal platforms is currently limited by the operating systems supported. Many of the leading portal servers are Java-based – and so theoretically platform-neutral – but the most commonly supported operating systems are Windows 2000, Windows NT, IBM AIX, Sun Solaris, and Red Hat Linux.

Although you can find portal servers that support other flavours of Unix, such as HP-UX and Compaq Tru64, the OS choices still boil down to Windows NT/2000, Unix, and Linux. If you’re an IBM shop, you may be surprised by the absence of mainframe operating systems and OS/400, given that, since around 1998, IBM has been making a big effort to make these powerful,
J2EE-compliant Java platforms. Especially since Java application servers, a prerequisite for Java-based portal servers, are readily available on mainframes and AS/400s from IBM as well as other vendors (eg BEA). But the real issue here is not technical; it has to do with testing and support.

Java’s proud claim of ‘write-once, run anywhere’ is to be applauded. However, as Java developers well know, this supposed platform independence is only realized via a ‘write-once, test on all platforms and then make necessary modifications’ process. This is the rub. Java, in the end, still has some inevitable platform dependencies and idiosyncrasies. If you have both Apple Macs and Windows clients you’ll already know this – to your cost. A Java applet that works immaculately on Windows may not work as well on a Mac. So much so that many Java-applet-based client-side solutions are supported only on Windows because the applet vendors don’t have the resources to test and support the applets on Mac, Unix, and Linux clients. The same issue applies to portal servers.

Given the perceived complexity of any type of mainframe deployment, a mainframe may not be your first choice as a portal platform. If you’re already a mainframe shop, however, and plan to have a high-volume portal, a mainframe could be an ideal portal platform. Today’s ‘on-the-fly’ capacity-upgrade-on-demand features – whereby you can just pay for additional processing capacity when you need to handle peak loads – make this option even more attractive from a portal perspective. Right now, however, there are only two ways to leverage a mainframe as a portal platform. One approach, which is both attractive and viable, is to use Linux. The other is to opt for a customized solution.

WINDOWS NT/2000 OR UNIX?

With Windows NT/2000, Unix, and Linux as the three readily available options, the portal platform decision basically hinges on a choice between Windows NT and Unix. Since few people can agree on which of these operating systems is better suited for enterprise use, the choice will usually be based more on
emotional factors than on technical ones. If you already use Unix for some of your mission-critical applications – in particular your Web servers – you’ll probably opt for a Unix/Linux approach. If, on the other hand, you currently have no Unix servers in your company, the chances are that you’ll favour an NT/2000 approach, at least to begin with.

However, security and scalability – two key issues when it comes to portals – should be given considerable thought. The recent spate of attacks by hackers and virus unleashing on NT family servers have severely damaged NT’s security credentials – despite Microsoft’s efforts to come up with fixes. So much so that respected consulting firms have advised clients not to use NT servers where they can be accessed over the Internet (eg as the basis for a Web server). A corporate portal server needs an Internet interface, so you need to think carefully about security if you want to opt for an NT approach.

Scalability is only an issue if you plan to have a high-traffic, high-volume portal; if your portal will be supporting 4,000 or more concurrent users in the near future, you need to do some careful testing and reference-checking before you decide on an NT approach. Although Windows 2000 is much more scalable than NT 4.0, Unix may give you better and more predictable scalability when dealing with high traffic volumes and lots of users.

There’s also the issue of hardware. Performance and scalability aren’t just OS issues: the underlying hardware plays a big role in this too. Unix servers, like mainframes, have always aimed for good performance, with RISC, parallel processing, and now clustering. If your company is already using medium to large Unix servers or mainframes to handle your current user loads, the chances are that you’ll start with a Unix server for your portal.

Another important consideration when selecting a portal platform is your current in-house expertise and experience base. If you’re predominantly an NT server shop, bringing in a Unix server may just complicate things. On the other hand, if you want to migrate towards Unix/Linux, a portal can be the killer application to justify this transition. And while you could outsource
the installation, deployment, and initial maintenance of a Unix server, the mission-critical nature of a corporate portal means that in the long-term you should have in-house resources to manage, maintain, and upgrade your Unix server(s).

Finally, there’s always the possibility of changing horses in mid-stream. Although not ideal, this can be a valid approach because the same portal servers will work both on NT and Unix. So, you could start with an NT implementation to cut your teeth on the intricacies of managing portals – on a platform that you know and already have lots of experience with. Then, as your user base and transaction volumes grow, you can think about moving to a Unix implementation.

JAVA OR DE-CAF?

Rather like the Windows NT versus Unix debate, the Java versus de-caf conundrum is ‘emotional’ rather than overtly technical. Windows NT/2000 is a proven, popular, and practical platform for Java, so the issue is not whether you can successfully run a Java-based portal server on a Windows NT server. Rather, it’s whether your partialities lie with Java or Microsoft .NET. It’s another spin on the old ‘open’ versus proprietary debate, except that now Microsoft, rather than IBM, is the purveyor of proprietary, platform-specific solutions. The Java camp, in the context of portals, is packed with most of the super heavyweights, including IBM, Sun/Netscape (iPlanet), Oracle, BEA, and PeopleSoft. Others, like SAP, support both J2EE and .NET. Microsoft, in turn, offers its own portal server, SharePoint.

Java has been around since 1995, so most corporations have a well honed view on it. If you’re already a committed Java shop, the portal decision is a no-brainer. There are lots of powerful and proven Java-based portal servers on the market; evaluate some from names that you’re comfortable with, and off you go.

If your company is already committed to .NET, your options are also cast in stone. You won’t have as many choices from big names as you would with Java, but there are enough .NET-oriented portal servers to ensure that you do have a choice.
If you’re still on the fence, however, it’s time for some serious thinking. It’s often said, justifiably, that Java hasn’t lived up to its once exalted expectations, but there’s no denying the fact that Java is very popular, and has enviable backing, market traction, and lots of momentum. On the other hand, Microsoft totally dominates corporate desktops and departmental servers. Much depends on your company’s philosophy, vision, and aspirations. Whether you like it or not, Java carries with it a connotation and aura of ‘big company’. On the other hand (despite Windows being the desktop of choice for the Fortune 500), saying that you are Microsoft-centric all-the-way projects an impression of ‘smallness’. There are always exceptions, but this ‘big’ versus ‘small’ perception is something that you should ponder. If you envisage your corporate portal as the next big thing to hit your company, as it most likely will be, you should give some serious thought to Java. Remember, going with Java doesn’t mean that you can’t use the Microsoft platform.

Fortunately, both Java and .NET promote the development and deployment of Web services, and it’s unlikely to be the deciding factor between them. Yet again, this is simply an issue of what you and your company are comfortable with.

| IBM’s WebSphere portal family (eg Portal Enable) | PeopleSoft’s PeopleTools 8.1 Portal |
| mySAP Enterprise Portals | Sybase Enterprise Portal |
| Microsoft’s SharePoint Portal Server 2001 | Brio Portal |
| BEA’s WebLogic Portal | Abilizer Web Engine |
| Plumtree’s Corporate Portal | Viador E-Portal |
| iPlanet’s Portal Server | Bowstreet Factory |
| Hummingbird’s EIP | Epicentric Foundation Server |
| Iona’s Netegrity Interaction Server | Corechange’s Coreport |
| Oracle9i Application Server Portal | Verity K2 Enterprise |
| Tibco’s ActivePortal | BroadVision InfoExchange Portal |
| CA’s CleverPath Portal (née Jasmine Portal) | Enfish (once KnowledgeTrack) |

**Figure 1: Portal servers (in no particular order)**
PORTAL SERVER SOFTWARE

There are two ways to set about implementing a corporate portal. You can either do it the difficult way by synthesizing ad hoc, à la carte programs, customized scripts, and individual services on top of a Web server. Or you can do it the easy way, using one of the popular, off-the-shelf portal servers.

Opting for a portal server-based solution doesn’t lock you into a rigid regime; the major portal servers provide many ways to customize, enhance, and augment corporate portal implementations via plug-ins, APIs, and adapters. Web services provide another way to extend a portal server’s scope, functionality, and reach.

Today’s conventional wisdom is therefore not to build a corporate portal from scratch, unless, of course, you have a budget to burn. Instead, the preferred approach is to start with a good portal server (see Figure 1) as the underlying foundation and then build on that.

Note that the list in Figure 1 is by no means exhaustive. It simply illustrates the range of representative, off-the-shelf solutions available. A growing number of portal server vendors emphasize the role of Web services in future portals, with nearly all already offering some level of support for Web services. There are two important messages to take away from this. The first is that there is near-universal concurrence that Web services, whether Java-centric or .NET-based, will play an increasingly significant role within corporate portals in the coming years. The other is that, far from impeding the potential deployment of Web services, using a portal server will most likely facilitate the adoption of this new methodology for Web applications.

In an effort to simplify portal development and maintenance as well as to differentiate themselves from each other, portal server vendors have introduced many innovative concepts and features over the last few years. Noteworthy among these are portlets, digital dashboards with Web parts, gadgets, breadcrumbs, skins, roles, domains, and iViews. Of these portal facilitating schemes, the notion of portlets (or related
concepts such as PeopleSoft’s ‘pagelets’) is probably the most pervasive, endorsed and supported by, among others, IBM, BEA, Oracle, Sybase, Viador, and Verity.

From a user’s perspective, a portlet is a content channel or an application ‘window’. On a Windows 9x desktop, each application interacts with the user via a window, with each window being a self-contained workplace with its own title bar, menu selections, and, if necessary, up-and-down ‘elevators’ for scrolling through the window. A portlet offers a similar self-contained workplace, complete with the necessary controls for an overall portal view.

<table>
<thead>
<tr>
<th>Category</th>
<th>Thin-client</th>
<th>Host publishing</th>
<th>Host integration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Host application access via portal</td>
<td>Host application access via portal</td>
<td>Develop new portal-centric applications or Web Services</td>
</tr>
<tr>
<td>Available from</td>
<td>Host access vendors</td>
<td>Host access vendors</td>
<td>Host access vendors and portal server vendors</td>
</tr>
<tr>
<td>Examples</td>
<td>IBM’s Host On-Demand, SEAGULL’s WinJa, WRQ’s Reflection for the Web</td>
<td>IBM’s Host Publisher, ResQNet ResQPortal, Hummingbird e-Gateway</td>
<td>IBM’s Host Publisher, SEAGULL’s Transidiom, BEA’s Java Adapter for mainframes</td>
</tr>
<tr>
<td>Characteristics</td>
<td>Host emulator in Java or ActiveX</td>
<td>Host-to-HTML/ WML/XML converter</td>
<td>Programmatic connections to the host via JavaBeans COM or APIs</td>
</tr>
<tr>
<td>Executes on</td>
<td>Client</td>
<td>Server</td>
<td>Server</td>
</tr>
<tr>
<td>Primary output</td>
<td>Client system specific – eg Window on desktop</td>
<td>HTML, WML or XML</td>
<td>Depends on application or Web service – but in this case portal oriented</td>
</tr>
<tr>
<td>Web browser disposition</td>
<td>Invoked via a Web browser; output could be within browser</td>
<td>Browser-centric. Data displayed within browser in HTML</td>
<td>Application or Web service but normally browser centric</td>
</tr>
</tbody>
</table>

*Figure 2 (part 1): Portal-to-data centre access techniques*
TCP/SNA APPLICATION INTEGRATION

If your company relies on IBM (or compatible) mainframes, IBM AS/400s (now iSeries), other minicomputers, or Unix servers for some or all of its IT needs, your portal must have access to the mission-critical applications and data located on these machines. The 150+ solutions for linking corporate portals with data centres, from close to 100 vendors, can be divided into three main categories.

Two of these categories, namely thin-client and host publishing, are designed for channelling existing interactive data centre access via the portal. The other category, ‘host integration’, is geared towards the development of new portal-centric

### Figure 2 (part 2): Portal-to-data centre access techniques (cont)

<table>
<thead>
<tr>
<th>User interface modernization</th>
<th>Thin-client</th>
<th>Host publishing</th>
<th>Host integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best suited for</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Primary advantages</th>
<th>1 Continue terminal emulation paradigm</th>
<th>1 Zero footprint. No host access software at the client</th>
<th>1 Enables host application logic to be reused</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Minimize installation, maintenance and upgrade costs</td>
<td>2 Easily adaptable for tight integration with portals</td>
<td>2 Totally extensible and permits synthesis of data from multiple sources</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Key disadvantages</th>
<th>1 Requires software on client systems</th>
<th>1 Still at heart a terminal emulation scheme</th>
<th>1 Requires some development effort</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 Difficult to ensure seamless portal integration</td>
<td>2 Not meant for combining functions from multiple applications</td>
<td>2 Requires the continuing presence of host application</td>
</tr>
</tbody>
</table>

| Best use vis-à-vis portal  | Tactical, short-term ‘stop-gap’ | Mid-term, portal integration with host data embellished with developing new new HTML-formatted Web services content | Strategic long-term, especially for portal data |

applications or Web services that wish to leverage data centre resources. The data centre and application adapters available with most portal servers fall into this third category, since they require some level of scripting or programming to ensure that they have an appropriate portal-compatible user interface. Figure 2 summarizes the key characteristics of these three techniques.

It’s worth noting that some data centre access is probably best done outside of the corporate portal, despite the desire to route all corporate IT access through it. Data entry operators are invariably the most cited example when it comes to this ‘bypass-the-Web altogether’ approach. Data entry operators are normally paid by the volume of transactions they complete in a day, and therefore want the fastest, most efficient, and least intrusive means of getting this done. Any intermediary processing, even host-to-HTML conversion, could slow them down.

HOST PUBLISHING AND HOST INTEGRATION

‘Thin client’-based terminal emulation is, at best, a tactical, stop-gap solution when it comes to data centre access via a corporate portal. For a start, a thin client relies on having host access-specific software on the client system – albeit software that can be invoked from a browser via a Web page link or button. Although this software can be dynamically downloaded from a Web server and cached (to remove the need for repeated downloads until a new version of the software is installed on the server), it is still counter to the notion of requiring just a standard Web browser to deal with a corporate portal.

The need for client-side software also makes this approach unsuitable for general, Internet-based public access scenarios, causing a delay while the software is downloaded and installed on the user’s system. Besides, many users will be uncomfortable with the idea of having large amounts of software dynamically installed on their system.

Host publishing is more appropriate for realizing portal-based host access because it works via dynamic, bi-directional host-
to-HTML, host-to-XML, or host-to-WML conversion, and therefore dovetails nicely with the portal paradigm. This server-side approach, whose roots go back to 1996, was called ‘publishing’ because it enables host data to be published on a Web page — now a portal view. Furthermore, the output of a host publishing solution can be easily ‘plugged into’ a portal view, because the output can be in HTML or XML form.

Some of the leading host publishing systems even enable you to alter the screen input/output sequence of a host application by allowing you to skip screens or combine I/O fields from different screens into a new consolidated view. You could even combine screen data from multiple applications — though at this point it’s best to start looking at host integration techniques. Some host publishing schemes, such as IBM’s WebSphere Host Publisher and NetManage OnWeb, can do both publishing and integration. They therefore provide an attractive migration path. Start by using publishing to portal-enable all the requisite data centre applications. Then look at host integration as a means to reuse the functionality of some of these applications to develop new portal applications or Web services.

From a portal integration perspective, host publishing is a compelling solution for short- to medium-term data centre integration. It’s also a very convenient way to generate XML representations of host screen data ‘on-the-fly’, without manual intervention — although this capability isn’t offered by all host publishing schemes. The downside of host publishing is that, in the end, it’s still a screen-oriented, terminal emulation scheme. It doesn’t have a role per se when it comes to enabling the business logic of existing host applications to be reused to build new applications — other than its ability to generate XML renditions of host screen data. This is where host integration comes in.

Host integration enables the proven business logic found within existing mission-critical applications to be reused when building new portal-specific applications or Web services. This slashes development costs, compresses testing schedules, and enhances the resilience of the new application.
However, it doesn’t allow you to just extract the necessary execution logic from the original application and then embed that ‘code’ within the new application. This wouldn’t be practical, for a variety of reasons – key among them being programming language incompatibility. Instead, it works by allowing the new application or Web service to make run-time calls to the original application, which then executes the transaction on behalf of the new application. The original host application, in essence, becomes a subordinate task that’s running on a different platform and passing relevant data back to the new application.

SUMMARY

The good news is that there is plenty of off-the-shelf software to enable you to quickly realize a powerful corporate portal. The lack of mainframe-specific solutions, other than via the Linux route, can still be somewhat frustrating, but host publishing and host integration offer more than enough options for quickly and easily integrating mainframe or AS/400 resources with portals implemented on Unix, Linux, or Windows NT platforms. There are no longer any technological reasons to delay the implementation of a corporate portal. All you need is a will, a mandate, and an adequate budget.

Anura Gurugé
Strategic Consultant (USA) © Xephon 2002

Information point – reviews

REDBOOKS – http://www.redbooks.ibm.com

Redbooks are an often-overlooked source of great information that goes well beyond the formal manuals that IBM publishes. A Redbook is the documented results of a Residency – an IBM staff and customer experience installing IBM hardware and/or software. They’re published by IBM's International Technical
Support Organization (ITSO), which, as the name implies, operates worldwide.

There are a number of ways to find Redbooks covering your areas of interest. The most obvious is the Search box in the upper right corner of every Redbook Web page. Just type a keyword – which needn’t appear in the Redbook title – and hit the Search button. For example, a search for AnyNet yielded 16 Redbooks, of which only the first three had ‘AnyNet’ in the title. Because the default presentation order is by relevance, the three with AnyNet in the title appear first.

The results from some network-relevant keyword searches are listed in Figure 1, to illustrate just how many Redbooks there are. As you’ll see from the entry for IP, a maximum of 250 results is returned for any search.

Note, however, that this includes Redbooks with no mainframe content, as well as Redpieces, Redpapers, and even an occasional Hint and Tip. (Redpieces are Redbooks-in-progress, while Redpapers are technical documents that have been written to address a specific topic but do not qualify to be a Redbook.) You may even see Residencies and Workshops listed, but not included in the count. Residencies are the two to eight week process where IBM staff, partners, and/or customers get together to produce a Redbook. Workshops use the materials developed during a Residency as the basis for a public course.

PORTALS

Redbook Portals provide an alternative approach to finding

<table>
<thead>
<tr>
<th>APPC</th>
<th>38</th>
<th>SNI</th>
<th>2</th>
<th>VTAM</th>
<th>45</th>
</tr>
</thead>
<tbody>
<tr>
<td>APPN</td>
<td>49</td>
<td>SNMP</td>
<td>21</td>
<td>X.25</td>
<td>15</td>
</tr>
<tr>
<td>FTP</td>
<td>32</td>
<td>SSL</td>
<td>35</td>
<td>XML</td>
<td>50</td>
</tr>
<tr>
<td>IP</td>
<td>250+</td>
<td>SSP</td>
<td>3</td>
<td>3174</td>
<td>14</td>
</tr>
<tr>
<td>NCP</td>
<td>11</td>
<td>TCP2</td>
<td>19</td>
<td>3270</td>
<td>39</td>
</tr>
<tr>
<td>SMTP</td>
<td>17</td>
<td>TCP/IP</td>
<td>216</td>
<td>3745</td>
<td>6</td>
</tr>
<tr>
<td>SNA</td>
<td>91</td>
<td>Telnet</td>
<td>27</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Figure 1: Results from network-relevant keyword searches*
Redbooks of interest. From anywhere within the Redbooks Web site, click Redbooks Online from the left sidebar, and then select the Redbook Portal of your choice from the right sidebar.

For example, if you choose Networking, the right sidebar has links to: Redbooks, Redpieces, Redpapers, Hints and tips, Residencies, Workshops, Redbook-related downloads, and CD-ROM collections. Clicking on Redbooks lists 334; the most recently published are listed first, and the list goes back a decade to the oldest.

Choosing the zSeries and System/390 Portal reveals that the Networking Portal is still under development, and lacks a few zSeries sidebar categories: What’s New and Top 15. Top 15 is especially interesting, as it lists the 15 most popular Redbooks. Several of the most popular zSeries Redbooks also belong to the Networking category, including the following of special interest:

- **TCP/IP Tutorial and Technical Overview** (GG24-3376).
- **IP Network Design Guide** (SG24-2580).
- **e-business Cookbook for z/OS Volume III: Java Development** (SG24-5980).

**HOW TO GET A SPECIFIC REDBOOK**

In the following examples, the IBM order number for the Redbook is GG24-3376 (TCP/IP Tutorial and Technical Overview). Substitute the order number of the Redbook you want for this number in the URLs shown. Omit hyphens and the revision number (the –06 in GG24-3376-06); only the latest version is available on-line.
To go directly to a Redbook in Web page (HTML) format, the URL is:


For PDF, you can go directly to the manual using the following URL:


An abstract with table of contents, publication date, number of pages, and other information is available at:


There, you’ll also find a Buy Now button under the heading Hardcopy. Surprisingly, this doesn’t take you to the IBM Publications Centre, but to amazon.com, directly to the full listing for the Redbook. The good news is that your book may be available at a lower price; the bad news is that the new and used copies from other booksellers may not be the current revision of the Redbook.

Clicking on How to buy in the left sidebar of any Redbooks Web page shows the IBM Publications Centre listed as a link to:

http://www.ibm.com/shop/publications/order

From there, select your country and get local publication ordering information.

CD-ROM

Each Redbook’s Abstract page also lists the IBM order numbers of the CD-ROMs that include the Redbook. Click on an order number and you’ll see the complete list of the Redbooks that you’ll find on the CD-ROM. Push the Buy Now button, or order them as above through the IBM Publications Centre.

Jon E Pearkins
(Canada) © Xephon 2002
IBM has announced z/OS V1.4, offering more tools, including those which:
- Simplify configuration, renumbering support, and application compatibility with new IPv6 support.
- Enable clock synchronization between clients and servers with a new TCP/IP daemon supporting SNTP.
- Simplify configuration and improve diagnosis capability and serviceability in SNA networks with Enterprise Extender (EE) and SNA enhancements.
- Provide additional configuration and definitional flexibility with tn3270 enhancements.

For more information, visit the Web site at: http://www.ibm.com/servers/eserver/zseries/zos/downloads/

---

IBM has also announced its Workload Simulator for mainframes running z/OS and OS/390, which performs stress, performance, regression, function, and capacity planning tests. It can simulate user-specified terminals and the associated messages, helping to decrease the number of terminals and reducing terminal operator time, and supports SNA, CPI-C (LU 6.2), and enhanced TCP/IP.

For more information, visit the Web site at: http://www-3.ibm.com/software/ad/workloadsimulator/about

---

Cisco has begun shipping its SN 5428 Storage Router, which integrates both IP and Fibre Channel switching capabilities, allowing enterprise workgroups to migrate from direct attached storage to storage area networks.

For more information, visit the Web site at: http://newsroom.cisco.com/dlls/prod_051402b.html

---

Stonesoft has announced its StoneGate VPN Client 2.0, addressing the need for location-independent secure connectivity, while protecting the remote device.

For more information, visit the Web site at: http://www.stonesoft.com/document/art/2697.html

---

Compuware has announced the availability of SoftICE 4.2.6 as a standalone product, which means users who have bought or are currently purchasing DevPartner Studio Enterprise Edition (Version 6.8 or 6.6), DevPartner Studio (Version 6.7 or 6.6), or DevPartner for Visual C (Version 6.6) can now buy the SoftICE system debugger tool as an add-on standalone product.

It has also announced the availability of Version 2.7 of its DriverStudio suite of tools to speed up development, debugging, testing, tuning, and deployment of Windows device drivers.

For more information, visit the Web site at: http://www.compuware.com/products/devpartner/softice