Using TCP/IP for VSE to Design a File I/O Subsystem

BY LEO J. LANGEVIN

At WAVV '97 I spoke for several days about TCP/IP, File I/O systems, and mounting VSE drives to a Windows operating system. After that, it's hard to get back into VSE-only mode, so I decided to write this month's column on the basics of designing your own File I/O system using TCP/IP for VSE and issuing operator commands to that system via FTP.

Whew! Let's see if I can make this simple...

VSE has had a native TCP/IP implementation for a couple of years now. TCP/IP has an FTP daemon, which is a server that can process standard FTP client requests. FTP clients have the ability to issue SITE commands or make requests to the FTP daemons that are site-specific. TCP/IP for VSE allows you to write your own SITE command processor as a complement to the one that comes with the FTP daemon. You can also use this same technology to write a File I/O driver to allow FTP, HTTP, and NFS to access your company's database.

TCP/IP for VSE allows you to write your own SITE command processor as a complement to the one that comes with the FTP daemon. You can also use this same technology to write a File I/O driver to allow FTP, HTTP, and NFS to access your company's database.

What's this all about? Well, last year, I wrote an article on writing your own CGI to do native VSE web page processing (***Technical Support***, November 1996). In that article, I introduced you to a special macro, FILEIOHD. That macro allows you to write a CGI (Common Gateway Interface) that intercepts HTTP requests to OPEN, READ, and CLOSE dynamically built web pages. When the web browser points to the VSE host and asks for a CGI, the VSE web server processes the request as if it were doing a File I/O request. An OPEN request is processed by one part of the program, the CLOSE request is processed by another part of the driver, and the CGI or File I/O driver handles the multiple READ requests. My point: The CGI is really nothing more than a pared down File I/O driver.

SO WHAT IS A FILE I/O DRIVER?

TCP/IP for VSE has an open architecture. It provides several interfaces that make it easy for other vendors to write add-on software. Several companies have different types of file systems whose access methods are not part of the normal VSE architecture. For example, BIM-EDIT (B I Moyle Associates, Inc.) has its own File I/O structure, CA-PCS (Computer Associates International) has its own File I/O structure, and so forth. Some of these companies provide APIs (Application Program Interfaces) that allow a programmer to use common routines to perform special functions. For example, BIM-EDIT provides some callable modules that allow a programmer to perform the usual types of requests such as OPEN, CLOSE, READ, WRITE, and so forth.

By writing a small module in the same manner as the CGI, you can provide a direct interface with TCP/IP for VSE. By coding routines to handle OPENDIR, CLOSEDIR, READDIR, OPENFILE, CLOSEFILE, READFILE, WRITEFILE, DELETE, RENAME, MKDIR, RMDIR, NOTE, POINT, SITE, and other options, you have an enormous potential for data access. As with CGIs, you can use some or all of these routines. The extensive documentation for the FILEIOHD macro as well as a working sample can be found at www.tcpip4vse.com/fioeiohd.html. Please use lower case, since this is on a UNIX server.

SO WHAT CAN I DO WITH THIS TYPE OF PROGRAM?

One company had a homegrown report archival database. The IS department wanted a way to FTP data in and out of the database easily. By writing a File I/O driver, their database became another mount point in the TCP/IP for VSE system. They could FTP data in and out. What they hadn't planned on was that they could use NFS to mount their entire database to a local PC and process archived reports as though they were local files. They could now also use HTTP to load database reports into web pages.

Representatives from B I Moyle worked closely with another company that wanted to FTP data in and out of BIM-EDIT. By using BIM-EDIT APIs, the File I/O driver was quickly written, and BIM-EDIT data can now be used by FTP, HTTP and NFS.

HOW DOES TCP/IP FOR VSE KNOW ABOUT THE DRIVER?

When you want TCP/IP for VSE to access a file system, you need to define it. For example, to define the PRD2 library to TCP/IP for VSE, a BIM-EDIT library, and an in-house database File I/O driver, you would use the following definitions:

```
DEFINE FILE,PUBLIC='LIBS/PRD2',DLBL=PRD2,TYPE=LIBRARY
DEFINE FILE,PUBLIC='LIBS/BIM',DLBL=BIMLIB,TYPE=BIM-EDIT
DEFINE FILE,PUBLIC='ARCHIVE',DLBL=FILE1,TYPE=DIR,DRIVER=DRIVER1
```

In the first two definitions, TCP/IP for VSE has some predefined file types, such as LIBRARY, ICCF, BIM-EDIT, VTOC, and so

forth. If you use one of these names, then TCP/IP for VSE will automatically use a predefined driver name, such as IPNFLIBR for LIBR, IPNFBIME for BIM-EDIT, IPNFPOWR for POWER, and so forth. In the third example, a user-written driver does not have any default, so a driver name needs to be provided. In this case, we use the name DRIVER1. Also, in order to indicate that this driver will use a directory structure, we will use a type of DIR.

SO WHAT DOES ALL OF THIS HAVE TO DO WITH SITE?

A SITE command is a standard FTP request. If you issue it from a DOS prompt, you sometimes need to precede it with a QUOTE request. When using graphical clients, this is usually unnecessary. For example, to tell the VSE FTP daemon that the PUT request will be based on a logical record length of 80 bytes, you would use:

```
SITE LRECL 80          <-- GUI
QUOTE SITE LRECL 80    <-- DOS
```

A SITE command is host-specific. This means that LRECL will work on VSE, but it may not work on some of the UNIX, Windows, or other non-VSE platforms. The FTP daemon has an internal table of valid commands. If you issue a SITE command that is not in the table, then an error message is displayed. This was fine for awhile, but the problem with trying to have an open system is that there is always going to be requests for more flexibility. Then it happened: A customer needed a method of issuing a SITE command to BIM-EDIT, so we extended the SITE logic. If the command is not in the internal table, then TCP/IP for VSE will check to see what driver controls the current subdirectory structure. If the driver has a SITE command section, then the system will pass control to it. The File I/O driver can then set register 15 to zero (good), four (bad parameter) or eight (invalid command). It can also return a buffered response.

AN OVERVIEW OF FTP MESSAGES

If you want a special message to be returned to the client, you will need to use a standard message number. In the FTP world, all messages are preceded by a four-position prefix, which is a three-digit number followed by either a blank or a dash. A blank after a number indicates that there was only one message line. A dash after a number indicates that there will be multiple lines of information. If you use multiple lines, then only the first and last line of the message will have anything in the first four positions, and all of the lines between will have nothing in the prefix. For example, the following is the result of the response of a "SITE D T" request that was given in a POWER subdirectory:

```
200-1R46I  TIME IS 10:21:15, DATE IS 11/20/97
  1R46I  013 PAGES FIXED, 022 CURRENT TASKS
200-Command Okay
```

In our implementation, we would check to see whether zero bytes of message were re-turned. If so, then one of the following messages is returned to the client (based on a re-turn code of zero, four, or eight, respectively):

```
200 Command Okay
501 Syntax error in parameters or arguments
202 Command or Parameter not implemented, superfluous
```

However, if you return a buffered response, then the daemon will always terminate it with the message "200-Command Okay." Therefore, you will need to put a "200-" prefix in the first line and four blanks as the prefix for all, if any, of the lines that follow, as was shown in the POWER example.

Of course, this means that to issue a POWER command, you must be in the POWER subdirectory; to issue a BIM-EDIT command, you must be in the BIM-EDIT subdirectory, etc. You could even go so far as to write a small File I/O driver that would not do any real I/O, had a null directory, and had only a SITE command section that would process SVC 30 requests, VTAM commands, and so on. Maintaining the VSE system from FTP would be a bit strange, but it can be done.

SO WHAT DRIVERS ARE ALREADY AVAILABLE?

Connectivity Systems provides drivers for standard VSE file access, such as LIBR, POWER, Sequential files, ESDS, KSDS, RRDS, VSAM catalog access, VTOC processing, ICCF, and BDAM. Right now, the only third-party vendor that has worked on creating a driver for their product is B I Moyle. If you have a file system written by another company, you may want to request that they develop a driver for TCP/IP for VSE. The documentation and example is straight-forward enough so that anyone with some assembler knowledge can quickly write their own program. The limit is only what one cannot imagine. *ts*

NaSPA member Leo J. Langevin has been involved with the VSE community since its inception. He is currently employed by Connectivity Systems and is the lead developer of NFS for VSE. You can reach him at leo@tcpip4vse.com.