Volume 12, Number 6
June 1991
ISSN 0270-7284

SNA Perspective

The single source, objective monthly newsletter covering IBM's Systems Network Architecture

tn3270: Another Interoperability Option

Companies installing local area networks (LANs) have traditionally faced a dilemma about which LAN technology to choose. Ethernet has been popular for groups networking personal computers with departmental computers. If access to an IBM mainframe is necessary, then token ring has been the network of choice. Generally, this has been an either/or decision.

If an Ethernet local area network is already installed, several options are available to allow connections to the mainframe: an SNA gateway with an SDLC, X.25, or token ring connection to the mainframe, installation of IBM's Advanced Interactive Executive (AIX) operating system, or replacement of the mainframe with IBM's departmental processor, the Advanced System/400 (AS/400).

This article discusses one more alternative to the selection process, tn3270, which is supplied by several vendors, including OpenConnect Systems Inc., FTP Software Inc., and IBM. It will first describe what tn3270 is, explain the architectural differences between the familiar 3270 environment and the tn3270 environment and, finally, discuss the technology employed by tn3270. Understanding each of these areas will help you evaluate tn3270 products that are available in the marketplace and make decisions about whether this technology will fit your corporate communications and application development strategies.

(continued on page 2)

A Lexicon of LU Naming

SNA was designed to be name-oriented, shielding the network user from addressing issues. Several architectural options and facilities are available to maintain the end-user transparency. These include Alias Name Translation Facility, ACBNAME=, uninterpreted names, LU 6.2 local names, USERVAR generic LU names, and transaction program name. This article describes how end users and LUs are named within an SNA environment and some of these techniques available for doing so.

(continued on page 12)

In This Issue:

(Continued from page 1)

What is tn3270?

tn3270 is a 3270 emulation for PCs on networks that employ the Internet suite of protocols to communicate with IBM mainframes that support TCP/IP. The Internet protocol suite has been the primary choice for UNIX networking, but is increasingly appearing in IBM LANs as well, both as part of IBM's UNIX product, AIX, as well as an option for Virtual Machine (VM) operating system environments (see Figure 1). The Internet protocol suite is applicable to both LAN and wide area networking (WAN) environments.

The characteristics of different terminals are defined, in the Internet protocol suite, by means of network virtual terminals (NVTs). tn3270 is one type of NVT capability and is supported by a protocol called Telnet, which defines how a terminal makes connections to a host across a network. Telnet uses Transmission Control Protocol (TCP) as its transport protocol and Internet Protocol (IP) as its network protocol. Although comprised of several protocols in addition to Telnet, TCP, and IP, the Internet protocol suite is usually referred to in the industry as TCP/IP, as it will be throughout this article.

The Beginnings of IBM's Mainframe TCP/IP Support

IBM announced support for Ethernet and TCP/IP on the IBM 9370 Information System in 1987. This announcement unveiled a LAN controller/adapter conforming to the Institute of Electrical and Electronic Engineers (IEEE) 802.3 standard (generally referred to as Ethernet), along with a TCP/IP software subsystem which runs under IBM's VM operating system for the 9370 (Program No. 5798-FAL). The software subsystem supports TCP/IP, thus making it possible for PCs running tn3270 to communicate with the IBM 9370. Since then, IBM has introduced several mainframe TCP/IP packages.

Architectural Differences between SNA 3270 and tn3270

There are architectural differences between the 3270 that operates in the SNA environment and tn3270, as is shown in Figure 2. The left figure represents the normal SNA environment where the 3270 datastream uses SNA elements—LU 1, 2, 3 and node type 2.0 on the terminal end of the connection and node type 4 and VTAM on the host end of the connection. On the right is a tn3270 environment for a VM environment using TCP/IP networking.

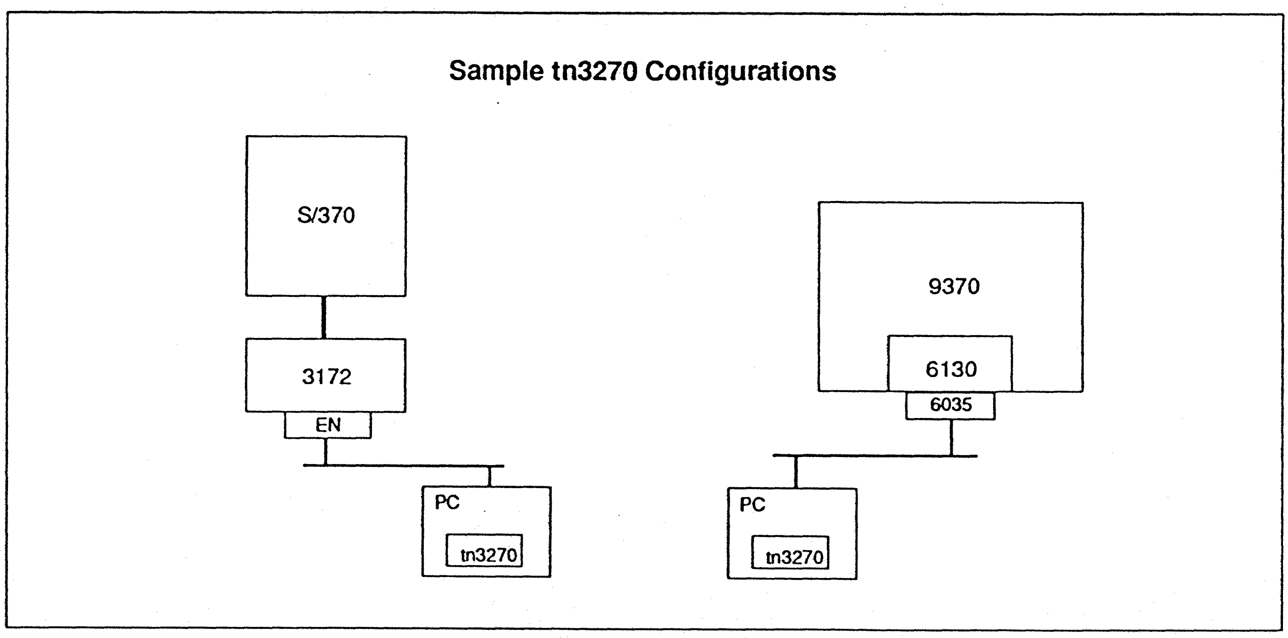

Figure 1

LU versus IP Addresses

The most significant architectural difference shown in Figure 2 is that tn3270 does not employ a logical unit (LU) and, therefore, does not use an LU address. Instead, tn3270 uses an IP address. Because of this, tn3270 provides more flexibility than SNA 3270, in that tn3270 may connect with any mainframe for which it knows the IP address. The SNA 3270 protocol is limited to connecting to a gateway that has an available LU for use or directly to a mainframe that has been configured for the terminal. The fact that tn3270 uses Internet addressing does not, however, limit the number of terminals that can be accommodated on the network. In the case of SNA, the maximum number of LUs that are available for terminals, per node type 2.0, is 254. In the case of TCP/IP, the maximum is 255, which is the size of an Internet domain.

Primary/Secondary versus Client/Server

A second architectural difference lies in the relationship of 3270 with its application. In SNA, the application is the primary LU and 3270 the secondary LU. Hence, the application initiates the contact with the terminal. For tn3270, the Telnet residing in the host performs the role of a server and tn3270 has the role of a client. In this case, tn3270 initiates the contact with the application.

Terminals Only

Another architectureal difference lies in the types of end-user devices that may be present at the terminal end of the network. For SNA, both terminals (LU 2) and printers (LU 1 and LU 3) can be configured for an LU. For tn3270, printers are not supported, which is definitely a disadvantage.

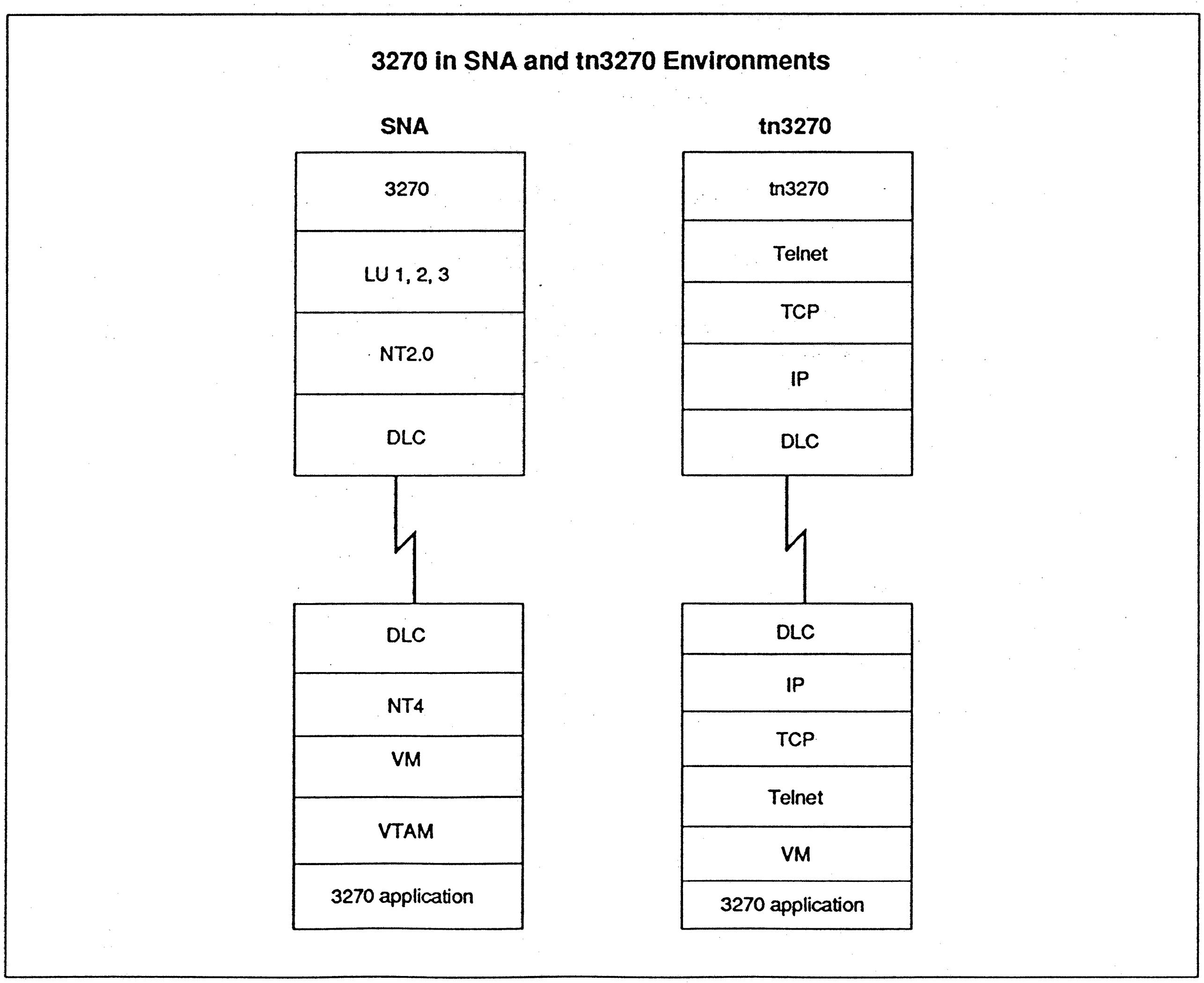

Figure 2

(In the Internet protocol suite, there is another protocol, Line Printer Daemon Protocol, for printing on devices attached to the host. However, there is no protocol for local printers at the terminal end of the network.)

TCP/IP Technology

tn3270 accesses TCP/IP/Telnet protocols through an application program interface (API). The various vendors of tn3270 either provide this API internally to their product or use a well-known API, the most common of which is the Berkeley Software Distribution (BSD) Sockets API. The BSD Sockets API allows an application to establish or listen for a connection, send and/or receive data on that connection, and release the connection when the application's task is finished. Generally, the availability of the specifications of the API will depend upon the type of vendor supplying tn3270. Vendors who primarily sell TCP/IP connectivity will include the specification of the API and vendors who sell tn3270 as a major product will not.

SNA over TCP/IP

tn3270 is not the only technology to use TCP/IP as the network transport mechanism. Many SNA gateway vendors provide or will soon provide TCP/IP as just a data link option. This technology is called SNA over TCP/IP. The basic idea is to encapsulate SNA frames within a TCP/IP frame at the points where the data traverses an Ethernet local area network. The gateway may map each link-level address to a single IP address, or may multiplex several or all link-level addresses to a single IP address. This mapping is generally controlled by a static configuration, but may soon be done dynamically.

This strategy preserves a company's investment in its existing SNA network, while allowing it to incorporate TCP/IP technology in its overall communications strategy. ■

The Internet protocols themselves are described in detail in published documents distributed to the Internet community. These documents are called "Request For Comments" documents, or RFCs. The RFCs are given numbers by the Internet Activities Board (IAB) and made available by the Network Information Center (NIC) located at SRI International in Menlo Park. One RFC describes the structure of the IAB (RFC1160); others give the latest status of various Internet protocols (The most current is RFC1200).

Telnet Technology

Because tn3270 is an Internet network virtual terminal emulation, it must adhere to certain characteristics of a virtual terminal as defined by the Internet. One of the most basic characteristics of NVTs is that they negotiate their capabilities prior to any data transfer. This is necessary to allow application programs to deal intelligently with the many kinds of hardware terminals they are likely to encounter.

The negotiation process is composed of a series of indications, delivered as byte sequences in Telnet frames. The exchanges that embody the indications take the form of frames consisting of a series of commands, each of which is preceded by a special code, "interpret as command" (IAC), which has the hexadecimal value 0xFF.

One type of indication, called either DO or DONT, is used to either enable (DO) or disable (DONT) a capability during the conversation. A second type of indication, called either WILL or WONT, is that the capability is either enabled (WILL) or disabled (WONT) by the sender. A partner may not use a capability until a WILL indication is received, or disable a capability until a WONT indication is received. Also, if a partner denies a capability, the requestor may not issue another request for the same capability "until something changes," to quote *RFC854: TELNET Protocol Specification*. This prevents infinite request-acknowledgement cycles. Either the client or the server may send an indication. A WILL/WONT indication is always treated as an acknowledgement of a previous DO/DONT, and vice versa. If an indication arrives to enable a capability that is already enabled, or to disable a capability that is already disabled, then no acknowledgement is sent.

Some capabilities involve a subnegotiation to allow more detailed capabilities to be negotiated. A subnegotiation process is described in more detail in *RFC855: TELNET Option Specifications* and *RFC1143: The Q Method of Implementing TELNET Option Negotiation.*

Telnet Technology in tn3270

tn3270 only needs to use a very small subset of the many characteristics that NVTs may have:

- Terminal Type
- End of Tecord
- Binary Transmission

These three capabilities are indicated by Telnet commands defined in RFC854. They are described in more detail in *RFC930: TELNET Terminal Type Option, RFC885: TELNET End of Record Option,* and *RFC856: TELNET Binary Transmission.*

Terminal Type

tn3270 uses the "Terminal Type" Telnet capability to determine the type of 3270 terminal that the implementation supports. This negotiation occurs prior to the negotiation for the other capabilities.

If the Telnet server does not support the terminal type indicated, it can perform additional terminal type negotiations. Clients may indicate the same terminal type during each negotiation, or different terminal types every time. Whether this is done and/or how the next terminal type is chosen are implementation issues.

End Of Record

tn3270 uses the "End of Record" Telnet capability to allow records larger than the TCP/IP frame size to be sent and received. This is necessary because tn3270 exchanges 3270 data streams, which do not necessarily contain length indicators within the data. This capability allows tn3270 to indicate the end of each logical record by an End of Record command, *IAC EOR* (FF EF).

Binary Transmission

tn3270 uses the "Binary Transmission" Telnet capability to allow binary data, rather than ASCII data, to be sent and received. The records exchanged by tn3270 will contain EBCDIC characters and other binary data. Because of this, both the server and client must use Binary Transmission.

As a side effect of allowing binary data, the Telnet sender may need to modify the data stream to allow for the inclusion of 0xFF as a data byte. This is necessary to avoid confusion with the special Telnet character *IAC*, which is also 0xFF. Telnet accomplishes this by inserting an additional 0xFF byte in front of each 0xFF data byte, an operation called *byte stuffing*. The Telnet receiver removes this extra 0xFF byte before delivering the data to the client.

Client-Server Interactions

We'll now look at how Telnet capabilities are actually negotiated. An example of the main interactions between the Telnet server and client is given in Figure 3.

Terminal Type Negotiation

Terminal Type negotiation is performed first and consists of two phases. The first phase involves the server issuing an *IAC DO TERMINAL_TYPE* request (FF FD 18), to which tn3270 replies *IAC WILL TERMINAL_TYPE* (FF FB 18). The server then issues a subnegotiation request *IAC SB TERMINAL_TYPE SEND IAC SE* (FF FA 18 01 FF F0). tn3270 replies with *IAC SB TERMINAL_TYPE IS <ASCII string> IAC SE* (FF FA 18 00 aa ... aa FF F0). The ASCII string "aa ... aa" designates the IBM terminal model number.

Shown following are the IBM terminal types designated in *RFC1060: Assigned Numbers.* A double asterisk (**) indicates that the terminal type is supported by at least one tn3270 product.

IBM-3275-2
IBM-3276-2
IBM-3276-3
IBM-3276-4
IBM-3277-2 **
IBM-3278-2 **
IBM-3278-3 **
IBM-3278-4 **
IBM-3278-5 **
IBM-3279-2 **
IBM-3279-3 **

Some of these terminal types may contain the suffix -E, indicating that 3270 extended attributes are supported. (See 3270 Data Stream Technology below for more details on the extended attributes.) This negotiation may continue until the server recognizes a Terminal Type which it can support. Once Terminal Type negotiation is completed, the other capabilities are negotiated.

End of Record Negotiation

Next, both server and client negotiate End of Record capability. This is done by exchanging *IAC DO END_OF_RECORD* (FF FD 19) and *IAC WILL END_OF_RECORD* (FF FB 19) commands. This is often done in one Telnet message, along with Binary Transmission negotiation, as seen in Figure 3.

Binary Transmission Negotiation

Then both server and client negotiate Binary Transmission capability. This is done by exchanging *IAC DO BINARY_TRANSMISSION* (FF FD 00) and *IAC WILL BINARY_TRANSMISSION* (FF FB 00) commands.

Finally, 3270 data is exchanged between server and client, with each record terminated with *IAC EOR* (FF EF).

In some cases, the server will attempt a second request for Binary Transmission negotiation. This second request may be ignored, according to the rules outlined in RFC1143. However, the request may be followed by 3270 data, because the two sides have already agreed to Binary Transmission.

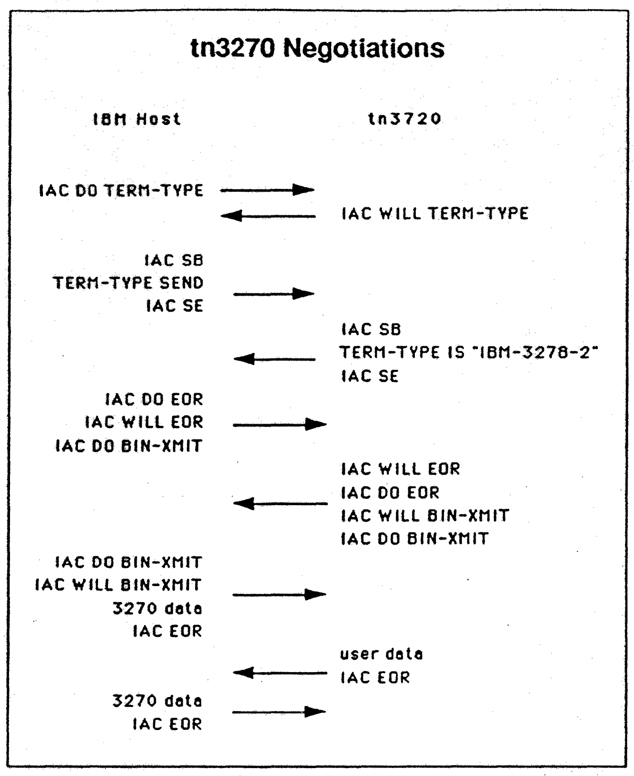

Figure 3

3270 Data Stream Technology

Most SNA technology is sent to tn3270 from the IBM host, although Attention Identification (AID) keys are sent by tn3270. Not all current products necessarily support every one of these SNA functions; current and/or future products may support additional SNA features.

The 3270 data stream is composed of four types of entities:

- Commands—define the function to be performed
- Orders—provide control information within a command
- Attributes—control the characteristics of a field or character
- Structured fields—used to send and receive additional control functions and data

SNA Commands

Commands perform general functions on the 3270 terminal. A list of SNA 3270 commands observed in the data streams of various commercial tn3270 products is shown in Table 1.

Some commands have a write control character (WCC) associated with them which is used for such functions as sounding an audible alarm and unlocking the keyboard. tn3270 processes this character in exactly the same way that a control unit with attached terminals would.

SNA Orders

Table 2 shows a list of SNA 3270 orders observed in the data streams of various commercial tn3270 products.

Other 3270 orders, such as Erase Unprotected To Address (X'12') and Modify Field (X'2C'), are probably also supported.

SNA Attributes

3270 attributes are indicated in one of two ways:

- Start Field order
- Start Field Extended order

In most circumstances, the Start Field order is used to define a field's attributes. Start Field Extended orders are sent to implementations that report support for extended attributes.

The Start Field order, Start Field Extended order, and the Attribute Pairs are illustrated in Figure 4.

SNA 3270 Commands

SNA Command	*Code (hex)*	*Notes*
Erase/Write	05	
Attention Identification	—	See the section Attention Identification Keys below
Write	01	
Write Structured Field	11	See the section SNA Structured Fields below
Erase/Write Alternate	0D	Seen when terminals allow extended attributes

Table 1

SNA 3270 Orders

SNA Order	*Code (hex)*	*Notes*
Set Buffer Address	11	Observed in data streams both received and sent by tn3270
Start Field	1D	Seen when terminal type indicates that extended attributes are not supported
Start Field Extended	29	Seen when terminal type indicates that extended attributes are supported
Repeat To Address	3C	Observed only in data streams received by tn3270.
Insert Cursor	13	Observed only in data streams received by tn3270.

Table 2

Start Field order, Start Field Extended order, and Attribute Pairs

The Start Field order is as follows:

Start Field X'1D'	Field Attribute Byte

The Start Field Extended order is as follows:

Start Field Extended, X'29'	No. of Attribute Pairs	Attribute Pair #1	...

The Attribute Pairs have the following format:

Attribute Type	Attribute Value

Figure 4

Field Attributes—All field attributes received should be accepted, but implementations of tn3270 do not always reflect all attributes on the terminal display or PC monitor. Most implementations can display attributes such as highlight, but most do not directly support blinking and reverse video, although some products alter color to indicate reverse video. Field attributes such as protected, unprotected, numeric and alphanumeric operate identically to IBM display stations, as shown in Table 3.

Extended Attributes—Start Field Extended orders contain the types of Attribute Type/Value pairs shown in Table 4.

Other extended attribute pairs such as (Reset) All Character Attributes (X'00'), Background Color (X'45'), and Field Validation (X'C1') should also be supported. Some attributes, such as Character Set (X'43'), Transparency (X'46') and Field

Field Attributes

Attribute	*Actions Allowed*
Unprotected, Alphanumeric	User can type any character into this field
Unprotected, Numeric	User can type only numerical characters into this field. Alphabetical and special characters cause the terminal or monitor to beep
Protected	User cannot type any characters into this field or the terminal or monitor will beep
Protected, Autoskipped	User cannot type any characters into this field or the terminal or monitor will beep

Table 3

Attribute Type/Value Pairs

Attribute Type	*Code (hex)*	*Notes*
3270 Field Attribute	C0	Values are the same as the Start Field order
Highlighting	41	Value normally specified as "Default" highlighting
Foreground Color	42	

Table 4

Outlining (X'C2'), may not be possible because of limited capabilities of the terminal or monitor supported by the implementation.

SNA Structured Fields

Structured fields are sent only to implementations that support extended attributes. The only two structured fields observed are:

- Attention Identification keys
- Read Partition Query request and reply

Attention Identification Keys—tn3270 supports the same AID keys as are available on the keyboards of IBM display stations such as the IBM 3278 Display Station. These AID keys include Enter, Clear, the Program Function (PF) keys, and the Program Access (PA) keys. When these keys are typed, a structured field is sent by tn3270 to the host. However, one product did not support the Clear AID key, so checking which keys are supported is important.

Read Partition Query—This request consists of a single query, which instructs the implementation to return the 3270 "functions" supported (for example, color). This request is generally carried in the same frame that performs the second request for Binary Transmission option (See Figure 5).

The Length field is zero, which indicates that the structured field is the last in a series. Also, the Partition Identifier value, X'FF', will be preceded by another X'FF' in the data stream. (See the section Binary Transmission, earlier, for a discussion of this.)

All tn3270 implementations reviewed by *SNA Perspective* report support for color (QCODE = X'86') and highlighting (QCODE = X'87'). Some reported support for usable area (QCODE = X'81') and character sets (QCODE = X'85'). One product reported support for distributed data management (DDM), but with an undocumented QCODE, X'93' instead of the documented QCODE of X'95'.

Framing

Because tn3270 processes records delimited by the End Of Record command *IAC EOR*, two extraordinary situations can occur when processing a Telnet frame (see Figure 6).

- More than one logical record may be contained in a Telnet frame
- Multiple Telnet frames may be necessary to contain one logical record

Selected List of tn3270 Suppliers

TCP/IP:
FTP Software, Inc.
P.O. Box 150
Kendall Square Branch
Boston, MA 02142

IBM Corporation
Department E15
P.O. Box 12195
Research Triangle Park, NC 27709

3270 Emulation:
OpenConnect Systems, Inc.
(owned by Mitek Systems Corp.)
2033 Chennault Drive
Carrollton, TX 75006

Some vendors of UNIX TCP/IP include tn3270 with their software. ■

Read Partition Query Request, Query Reply, and Query Reply Subfields

The format of the Read Partition Query *Request* is as follows:

Write Structured Field	Length (2 bytes)	ID, Read Partition	Partition Identifier	Type of Operation (Query)
11	00 00	01	FF	02

The Read Partition Query *Reply* consists of several subfields identifying the functions supported, as follows:

AID Key (X'88 = Structured Field)	Query Reply #1	Query Reply #2

All Query Reply subfields have the following format:

Length (2 bytes)	ID, Query Reply	QCODE	Additional fields as needed ...
00 NN	81	(see below)	...

Figure 5

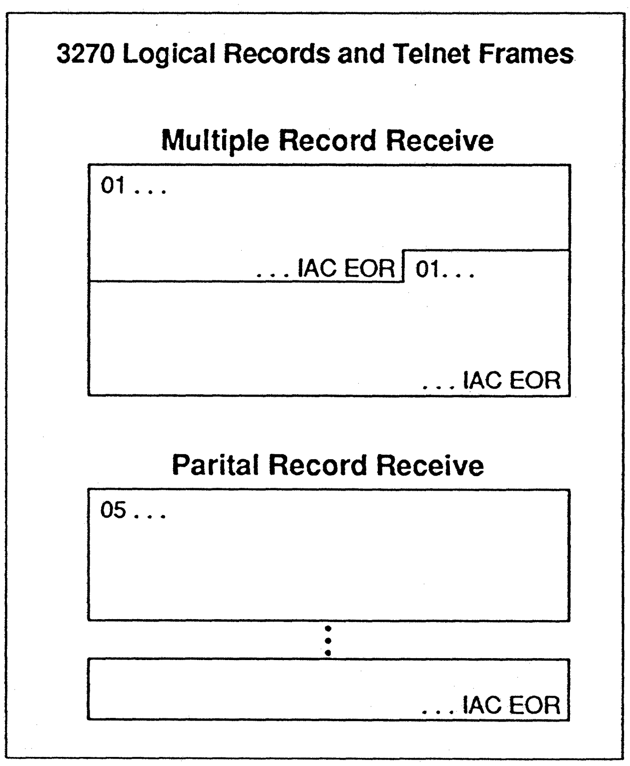

Figure 6

Multiple Record Receive

In this case, each Telnet frame holds two or more logical records, each terminated by *IAC EOR*. This can occur in cases involving multiple SNA Write commands where the command contains a partial screen update.

Partial Record Receive

In this case, a logical record spans more than one Telnet frame. The record is terminated by *IAC EOR*, which occurs in the last Telnet record. This can occur in cases involving an SNA Erase/Write command which contains a full 3270 screen's worth of data. The API gives no indication that a partial record has arrived; it is the responsibility of the tn3270 implementation to determine when the complete logical record has been received. However, implementations may process one Telnet frame containing part of the record before getting the next Telnet frame.

Multiple Sessions

Some implementations of tn3270 allow multiple 3270 sessions to occur in parallel, with one of the sessions being active at a time. This is generally accomplished by actually maintaining several independent tn3270 sessions and implementing a keystroke or sequence of keystrokes for switching between them. This functionality is similar to that exhibited by IBM's 3270/PC product. One advantage of this capability is the possibility of connecting to a different mainframe with each of the sessions, giving simultaneous access to different mainframe applications.

File Transfer

Even though the Internet protocol suite includes protocols for performing file transfers, most notably File Transfer Protocol (FTP) and Trivial File Transfer Protocol (TFTP), some tn3270 products contain file transfer capability within the terminal emulation. These file transfer capabilities use the file transfer mechanisms built into mainframe applications. IBM provides these capabilities in three of its mainframe products:

- Customer Information Control System/Virtual Storage (CICS/VS)
- Virtual Machine/Conversational Monitor System (VM/CMS)
- Time Shared Option (TSO)

All three products use the IBM-supplied program IND$FILE to control the file transfer process. Thus, tn3270 products which support file transfer use the capabilities of this program.

Conclusions

The choice of tn3270 versus an SNA gateway is based primarily on the corporate strategy for PC connectivity. If the strategy revolves around a central mainframe and each PC is a replacement for an IBM display station, then the SNA gateway approach makes the most sense. If the strategy revolves around small LAN on which the mainframe is another processing node, then tn3270 would be the best choice.

The advantages of tn3270 are:

- Configuration of networks is simpler, both at the mainframe and at the terminal
- It is easier to access multiple mainframes simultaneously
- tn3270 may be available as part of the TCP/IP package, so no additional purchase may be required (though more features may be available in products from vendors who provide just 3270 products)

The disadvantages of tn3270 are:

- Mainframe applications may have to be rewritten to use the TCP/IP programming interface
- tn3270 does not use SNA technology, which makes it different from standard 3270
- There is no support for printers at the terminal end of the network

The non-issues in considering tn3270 are:

- The number of terminals that can be accommodated on the network
- The 3270 attributes available to mainframe applications ■

References

RFC854
TELNET Protocol Specification, Postel & Reynolds, May 1983

RFC930
TELNET Terminal Type Option, Solomon & Wimmers, January 1985

RFC885
TELNET End of Record Option, Postel, December 1983

RFC856
TELNET Binary Transmission, Postel & Reynolds, May 1983

RFC1060
Assigned Numbers, Reynolds & Postel, March 1990

RFC855
TELNET Option Specifications, Reynolds & Postel, May 1983

RFC1143
The Q Method of Implementing TELNET Option Negotiation, Bernstein, February 1990

GA23-0059
Data Stream Programmer's Reference, IBM Corp., March 1990

GX20-1878
3274 Control Unit Reference Summary, IBM Corp., December 1988

Note: RFCs are made available by the Network Information Center located at SRI International, 333 Ravenswood Avenue, Menlo Park, CA 94025.

(Continued from page 1)

The data communications environment provided for end users, by either the traditional subarea SNA or the new peer-oriented SNA, is analogous to the environment offered by telephone systems to their subscribers. Both SNA and telephone systems ensure that their users (i.e., end users or subscribers) can freely interact with each other using uniform connection (i.e., session) establishment, data exchange, and call disconnection (i.e., termination) procedures that are independent of either the type, physical characteristics, or geographic locations of the users involved.

A telephone system subscriber can call any other subscriber on that system or on another accessible system by simply dialing a telephone number. The telephone number of the subscriber being called, prefixed by area code and country code when necessary, is the only prerequisite information required to place a call. A caller can dial a telephone number without having to know or sometimes not knowing (when, for example, calling a subscriber who is using a mobile cellular telephone) where the subscriber being called is physically located or what type of telephone handset the subscriber is using. The techniques for placing, using, and disconnecting a telephone call are standard across industrialized nations and are independent of the geographic locations of the subscribers and the types of handsets they might be using.

SNA also strives to provide equivalent interconnection flexibility through comparable, uniform interaction procedures for its end users. The architecture goes to great lengths to ensure that an end user can initiate the establishment of a session through which to interact with another end user via a single standard session initiation process that is independent of the geographic locations, types, or even characteristics of the end users involved. A terminal user in an SNA environment can routinely log on to an accessible application program by entering an appropriate session initiation request, such as LOGON APPLID(TSO), without ever having to know on which host the application is executing or the route to that host. The application could even be moved to a different host between sessions without the terminal user ever being aware of it.

SNA, however, differs radically from telephone systems in one fundamental and crucial aspect of connection establishment—destination user identification. Telephone systems, at least at present, rely on an address-based (i.e., telephone number) identification scheme rather than a subscriber name-based method. In marked contrast, SNA has always been, and will continue to be, very much a user-name–oriented architecture rather than a user-address–oriented architecture.

Many people are under the impression that SNA is an address-oriented architecture. This is reinforced by several aspects of SNA. For example, it is true that the basic constituent building blocks of SNA environments—logical units (LUs), physical units (PUs), and system services control points (SSCPs)—are referred to as *network addressable units* (NAUs). Further, most of the mandatory and memorable operands required by VTAM and NCP definitions (e.g., SUBAREA=, LOCADDR=, HOSTSA=, MAXSUBA=) apply to addressing. Finally, like all major, large-scale networking schemes, subarea-based SNA (i.e., the bedrock 1978 SNA-4 architectural specification) relies on an underlying network addressing scheme to identify all NAUs and data links and to convey message units between NAUs. However, all network address assignments are performed, and all transactions involving network addresses are conducted, behind the scenes by control programs such as VTAM, NCP, and 3x74 microcode in such a way that end users, particularly humans, can be totally oblivious to network addresses.

Session Partner Identification

The ability for SNA end users to identify and interact with each other using meaningful, intuitive, and mnemonic names rather than arbitrary and usually lengthy addresses was a primary architectural goal of SNA. It has also proved to be one of the intrinsic strengths of SNA that has contributed to its unparalleled success. SNA itself, as well as all major implementations of SNA, permit end users to identify each other by an unadorned, symbolic name of up to eight characters (i.e., bytes) in length, regardless of the diversity or complexity of the SNA environment.

Unlike some of IBM's contemporary electronic mail systems or even the widely used TCP/IP-based Internet network, SNA end users are not even expected to qualify the addressee's name with a location identifier such as the node, host, or network through which the subject end user may be found. Thus an SNA end user invoking a session initiation or termination process can simply identify a potential session partner with a name such as CICS, TER06C02, or DIVMNGR, instead of using qualified names like TSO@HOST3, CICS.USNET, or TERM3@NODE09. Name qualification at the end user level is not even an option that could theoretically be exploited by experienced end users.

This is still the case in SNA Network Interconnection (SNI) environments where multiple, autonomous SNA networks, each with its own independently managed end user naming schemes, are loosely coupled to form a freely interoperable, integrated internetwork. It is true that, since the advent of SNI and Type 2.1 nodes, network (identifier) qualified names or fully qualified network names are routinely used by the SNA session initiation, session establishment, and session termination protocols. However, appending the appropriate network ID to fully qualify a name is again performed behind the scenes by a control program fulfilling the role of an LU, SSCP, or Type 2.1 control point (CP), rather than by an end user.

Directory Services

Because SNA, or at least subarea-based SNA-4, relies on network addresses, symbolic end-user names must be converted at some point into appropriate SNA network addresses. This name resolution, called Directory Services, is performed in subarea-based SNA environments by SSCPs during session initiation processing. Ironically, the much maligned S/370 host-centered, hierarchical nature of traditional SNA stemmed from the need to provide Directory Services function in order to be name oriented. In the early 1970s, when SNA was being formulated, S/370s were the only general-purpose processors with sufficient processing, memory, and disk storage capacity to provide a real-time Directory Service for even a reasonably large network.

Thus, SSCP-housing Type 5 nodes were implemented on S/370 hosts and the networks for which they were providing Directory Services were constructed around them.

The name orientation of SNA encompasses not just end users but also the SNA system or network operator interface. Thus, operators managing an SNA environment can do so by using resource names rather than cumbersome and easily mistyped network addresses.

Type 2.1 nodes and APPN, which is based on Type 2.1 nodes, do not use persistent network addresses like SNA-4. Instead they use a dynamic, reusable session index known as a local form session identifier (LFSID) to identify specific sessions. In the absence of global network addresses, Type 2.1 node and APPN session initiation and session establishment protocols are designed to be entirely name-oriented rather than address-oriented. (See January 1990 SNA Perspective: "Breaking the Chains of Hierarchical Networking: Integrating Node Type 2.1.") Thus end-user names and associated LU names assume even greater significance in new SNA. The Directory Service function that determines the node containing a target end user in Type 2.1 node environments is performed by the local CP serving the calling end user, while in APPN networks it is performed in a distributed manner by the local CP and the CPs in relevant APPN network nodes.

End User and LU Naming

Prior to the introduction of LU Type 6.2 in 1982 and the now-legendary LU 6.2 protocol boundary, SNA did not have a consistent methodology to clearly differentiate between the exact composition or functions of SNA end users and those of the LUs serving those end users. In most instances, particularly in the case of application program end users, it was difficult to clearly identify the end user-to-LU boundary. SNA also used to claim that the definition and specification of the characteristics and composition of SNA end users—other than just identifying them as falling into the generic categories of application programs, terminal users, and I/O devices—was outside the scope of SNA. This precluded SNA from being able to explicitly define naming conventions, name formats, and identification schemes for end users.

SNA adroitly overcame this limitation by allowing LU names to be surrogate end-user names. Thus the supposed end-user names in current SNA-4–compliant environments that enable end users to to identify and establish sessions to interact with each other are in reality SNA LU names rather than true end-user names. Because this has been the case for the last seventeen years, most SNA users, SNA network administrators, and network implementors take it for granted. The gradual but inevitable migration toward APPN networking and LU 6.2-based applications over the next few years will, however, alter this scenario dramatically. Because LU 6.2s can and invariably will support multiple end users, they typically require the target end user to be explicitly identified.

The use of LU names as surrogate end user names, though an expedient architectural and implementational work-around, was another key contributing factor to the blurred distinction between an end user and the LU serving it. Thus, typical names used in SNA environments (e.g., TSO, CICS, IMSTERM03, or A03T0812) actually referred to an LU rather than an end user. This was not a major issue since, in most SNA implementations, there was a one-to-one mapping between an end user and an LU. Although his one-to-one mapping between end users and LUs was permissible, it was not explicitly dealt with by the architecture.

SNA architecture, prior to LU 6.2, essentially relegated the end user-to-LU mapping issue to being an implementation-specific option. This meant that it was also possible for an LU to support multiple end users. Early SNA RJE systems like the 3770s were examples of SNA implementations that supported multiple I/O device end users per LU. Such LU configurations, however, face the problem that the surrogate LU naming technique is no longer adequate to uniquely identify the destination end user. SNA has therefore provided two end-user partitioning techniques that could be used with non-

6.2 LUs that supported multiple end users. These two techniques are the use of Function Management Header Type 1 (FMH-1) and the innovative use of Session Instance Identifiers in the User Data field of the BIND request used to establish a given session. Both techniques provide a prefixing scheme that can be used by the destination LU to select the intended target end user. It is also possible to achieve end-user partitioning at the application level using a scheme that relies on embedding end-user identifiers with the end-user message units. Fortunately, end-user partitioning will cease to be a concern as LU 6.2, with its concept of conversations that occur between end users rather than sessions that occur between LUs, provides an elegant solution for future multiple end-user support.

LU Network Names

Each LU within an SNA environment must have a symbolic name assigned to it. This name, which is typically expected to represent the role of that LU and is not qualified in any way, is referred to as the network name of that LU. Although the architecture does not explicitly state a maximum length for an LU name, the fixed length of certain fields within key SNA requests such as the BIND request (which is used to establish a session) dictates that LU names cannot exceed eight EBCDIC characters (i.e., eight bytes) in length. Thus, one- to eight- character LU names are supported by all major IBM SNA implementations, including VTAM and NCP.

Architecturally, and in most implementations, an LU name must begin with an uppercase alphabetic character, @, #, or $, and can only contain these or numeric characters in character positions 2 to 8. (When defining LU names in ACF/VTAM Version 3 Release 3 onward, it is theoretically possible to use both uppercase and lowercase alphabetical characters without any restrictions.) IBM provides a variety of common sense guidelines for LU naming—including a "T" in the name of LUs serving terminal end users and an "A" in the name of LUs in channel attached control units. LU naming guidelines can be found in the IBM manuals *SNA Products Installation Guide ACF/VTAM Release 2* GG24-1509 and *Network Program Products: Samples* SSC 30-3352.

The network names of LUs do not necessarily have to be unique across an entire SNA environment. They only have to be unique within the domains that the LUs may have to deal with when establishing interdomain LU-LU sessions to satisfy various end-user interaction requirements.

Each LU within a given domain must have an unique LU network name. Thus, if an SNA environment consists of just a single domain, each LU in that environment has to be assigned a unique LU network name.

If an SNA environment consists of multiple domains, the uniqueness of an LU network name depends on the span of the LU-LU sessions that an LU may require to participate in. The network name assigned to an LU has to be unique within each domain that contains other LUs wanting interdomain LU-LU sessions. The network name of an LU does not, however, have to be unique in any intermediary domains traversed by such interdomain sessions.

The same LU network name may be assigned to LUs in separate domains provided that LUs with the same name do not participate in interdomain LU-LU sessions that impinge on domains containing another LU with the same network name. This does not preclude LUs with duplicate names participating in interdomain LU-LU sessions; it just partitions the domains with which such LUs can freely interact. LU network names in a multidomain environment are therefore very different from the SNA network address(es) assigned to each LU. Network addresses always have to be unique throughout multidomain environments irrespective of the span of interest of the LU.

In SNI environments, LU network names need only be unique within each individual, autonomous SNA network. A primary goal of SNI was to ensure that each autonomous network had free reign to assign LU names without having to worry about possible duplicate names in other networks. If individual networks are multidomain environments, as most SNI networks are, the LU naming rules for multidomain environments described above will apply within each network. However, as with multidomain environments, duplicate names cannot be tolerated during internetwork session establishment. This limitation can be overcome by using the Alias Name Translation Facility provided by NetView (or NCCF). LU name aliasing allows any given LU to have two active LU names: the *bona fide* name by which it is known in its own network and the alias name by which it is known in other networks.

In SNI and Type 2.1 Node/APPN environments, when SSCPs or CPs are resolving LU names to determine their location, they automatically append to the front of the LU name a network identifier of up to eight characters that corresponds to the network in which the subject LU is located to the front of the LU name. The network identifier and the network name are then separated by a period. An LU name qualified by a network identifier is referred to as a fully qualified network name, which can be up to 17 bytes in length because the period used to separate the two components of the name is always present.

Network names for LUs supporting S/370 host application programs are defined as labels prefixing the APPL definition statements that are included within a VTAM application program major node. An alternate name that is only valid during intradomain session initiations or terminations can be assigned to an application program LU using the ACBNAME= operand, which can be included in the APPL definition statement. This allows exact copies of an application program using the same internal name to identify its supporting LU to be used unmodified in multiple domains. (This would, for example, be the case with IBM application programs such as IMS, TSO, or CICS.) Another LU wishing to interact with such an application program can now access the copy in its own domain by using the alternate name specified in the ACBNAME (which will be the same in all domains) or access a copy in another domain by using the LU name assigned via the APPL statement label in the target domain.

Network names for LUs in Type 2, Type 2.1, and Type 1 nodes that typically support terminal user or I/O device end users are defined by LU definition statements to VTAM (in the case of channel attached nodes) and to NCP (in the case of link attached nodes). As with application program LUs, network names are assigned to the LU in the form of labels prefixing the LU definition statement. LU statements, however, do not provide a facility comparable to ACBNAME= in which an intradomain-only name can be assigned to LUs.

Uninterpreted Names

SNA offers an optional capability for an end user or an LU serving an end user to identify another LU serving a target end user by a local alternate name instead of that LU's *bona fide* network name. Such a local, alternate name for a target LU is referred to as the uninterpreted name of that LU. One or more uninterpreted names may be associated with a given LU. Uninterpreted names are only meaningful and valid in session initiation and session termination requests such as character-coded logons, field-formatted INIT-SELFs, INIT-OTHERs, or TERM-SELF.

An SSCP receiving a request from an LU in its domain that refers to another LU by an uninterpreted name immediately uses a local Interpret Table to interpret that name into the network name corresponding to the LU in question. The network name derived from the Interpret Table will be used in all subsequent interactions. Using the Interpret Table facility, uninterpreted names can be used in VTAM environments to refer just to S/370 host-resident application programs.

Architecturally, LU 6.2s also provide a built-in local uninterpreted name translation capability. Thus, an LU 6.2 end user can refer to a remote LU by an uninterpreted name known to its local LU rather than by the remote LU's *bona fide* network name. The LU, as opposed to an SSCP or CP, is responsible in this instance for interpreting the alternate name into the appropriate network name. (Unfortunately, the LU 6.2 refers to such uninterpreted names for remote LUs as local names.)

Generic LU Names

In order to support XRF (see November 1990 *SNA Perspective* "XRF: High Availability à la SNA"), ACF/VTAM Version 3, Release 1.1 provides an implementation-specific feature—the USERVAR facility in VTAM Interpret Tables—to facilitate the seamless functioning of XRF-based applications. This facility enables users to log on to a generic rather than a specific application, thus effectively shielding users from the twin-application nature of an XRF configuration. Users always log on using the same application name regardless of which copy of the application is active. For example, in an XRF-proofed IMS/VS system, users will always be told to log on by entering LOGON IMS, even though the two copies of IMS are defined to VTAM with the names IMSA and IMSB. A variable USERVAR(iable) associated with the name IMS will be set to indicate whether IMSA or IMSB is the currently active version. A system operator can reset this variable using the new MODIFY NET,USERVAR VTAM command as follows:

```
MODIFY    net,USERVAR,ID=IMSVAR,
                    OPTION=UPDATE,VALUE=IMSB
```

where IMSVAR is the name of the (USER)variable associated with the IMS logon sequence. The Interpret Table USERVAR capability which is invaluable in XRF configurations, is inevitably assumed to be an XRF feature. This is not the case. It can be and is used quite independently of XRF to gainfully implement generic logon schemes that could be of particular interest in installations with multiple, crossdomain S/3x0 hosts since it facilitates the transparent movement of applications between hosts, load balancing, and application backup.

LU 6.2 and APPN

In terms of LU naming, LU 6.2s do not differ significantly from other LUs. Where they do differ significantly is in terms of end-user identification. It is no longer assumed that one LU 6.2 will only host one end user. Thus, the LU 6.2 protocol boundary introduces a new operand, the transaction program name (TPN), through which the remote end user can be explicitly identified by a symbolic name of up to 64 characters. Exactly how this TPN is specified is, unfortunately, implementation dependent. For example, the VTAM APPCCMD API expects the TPN name to be embedded within an FMH-5 data structure constructed in a buffer and does not provide an explicit parameter for it. Nonetheless, end users will invariably have a means by which to specify a remote TPN.

Since APPN is LU 6.2-based, it also uses the TPN concept. This gets confusing since APPN refers to LUs as LOCATIONS, which was intended to allow users to think about where the nodes at which the applications they required were located—at LOCATION NEWYORK, LONDON, etc. This was an unnecessary refinement because APPN, true to its SNA roots, permits location-independent session establishment. The easiest way to deal with this is to think of LOCATIONS as LUs, as stated in APPN manuals, and to remember that the individual application program end users served by each LOCATION (i.e., LU) will be identified at the application level by a TPN.

Conclusions

Among SNA's goals were interconnection flexibility regardless of network size and complexity, which relies on a robust underlying network addressing scheme, and transparent and symbolic nomenclature for end users, which led to name orientation rather than address orientation.

Therefore, SNA was designed to be name-oriented, from both the end user and network operator points of view, and address-oriented internally, shielding the network user from the addressing issues. Name-to-address resolution is performed by a control program within the network. Even with APPN, this resolution is accomplished behind the scenes, though in a distributed fashion through several control programs.

In subarea SNA, LU names became surrogate end-user names since, in most implementations, one LU served one end user. When this was not the case, SNA relegated the problem of end user-to-LU mapping to an implementation-specific option. Several techniques were available for end-user partitioning, including FMH-1, Session Instance Identifiers, and embedded end-user identifiers.

LU 6.2 and APPN bring this issue to the surface, since most LUs will support multiple end users. Also, LU 6.2 uses the concept of conversations between end users rather than sessions between LUs. Each LU has a network name. There are certain requirements for uniqueness of LU network names; however, they need not be completely unique across the entire network, as the address must be.

Several architectural options and facilities are available to maintain the end-user transparency and symbolic name use for LU names. These include Alias Name Translation Facility, ACBNAME=, uninterpreted names, LU 6.2 local names, USERVAR generic LU names, and transaction program name. ■

SNA Announcements

McDATA Announces New Controller Line

In May, McDATA Corporation of Broomfield, Colorado, announced a new family of network controllers. The LinkMaster 7100 network controllers feature Ethernet and token ring integration, multiple host access, non-IBM system access, and an Enterprise Systems Connection (ESCON) interface.

McDATA has consolidated multiple modular options into three basic 7100 models—10, 20, and 60—the first two of which have local and remote capability. A comparison of IBM 3174 and McDATA LinkMaster 7100 models is shown in Table 5. The 7100 family appears to obsolete the company's 4174 family, except for its low-end models.

The controllers support coax, ASCII, and PC users. The 7100 family can be configured to connect to token-ring or Ethernet networks. However, 7100

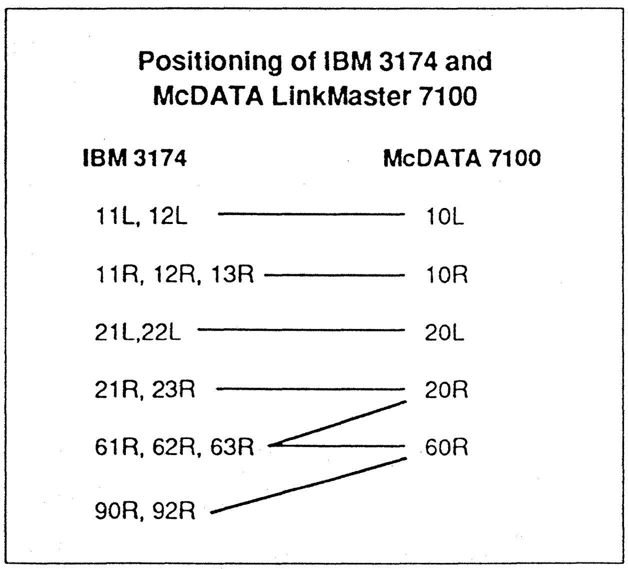

Table 5

local controllers cannot connect to Ethernet LANs; this capability is available in McDATA's 6100E. Unlike the IBM 3174, the 7100 models can be field upgraded from remote to local capability. Both SNA and non-SNA protocols are supported on the same channel, which can eliminate redundant controllers for consoles.

The 7100 controllers can be used in three configurations, as shown in Figure 7: traditional (connecting directly attached devices locally or remotely to one or more hosts), gateway (supporting connection to one or more hosts from a LAN), or downstream node (supporting devices through a LAN or bridged LAN to a gateway 7100). The latter two must be used in combination—a downstream 7100 communicates to one or more gateway 7100s and a gateway 7100 supports one or more downstream 7100s. Controllers in gateway configurations can simultaneously support traditional direct connections and LAN-attached devices as well as downstream 7100s.

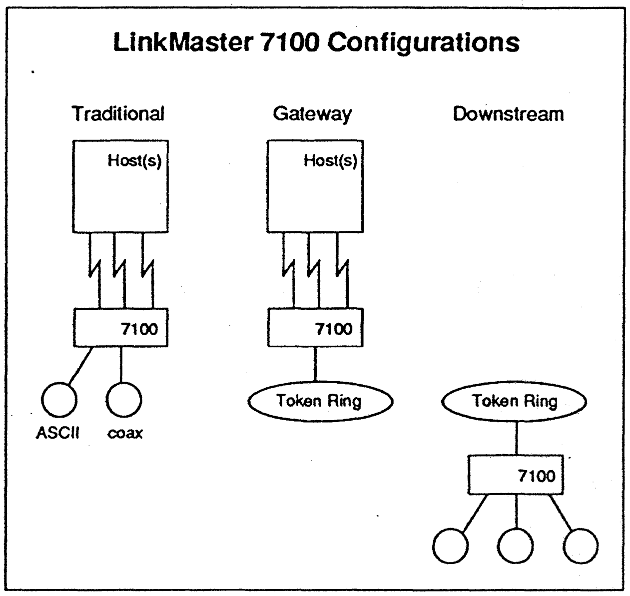

Figure 1

From a token-ring downstream 7100, a user can access up to four hosts now and up to eight hosts by late 1991. From a Ethernet downstream 7100, users can access both IBM and VAX hosts, communicating with the latter via Digital Equipment's LAT protocols. *SNA Perspective* expects that the company will add TCP/IP support in a future release.

Several of the 7100 capabilities are summarized in Table 6. Prices range from $5,635 for a 16-port, single remote host configuration to $29,935 for a 128-port single channel configuration. Additional host and LAN connections increase the prices. McDATA's 7100 prices are comparable to IBM's controllers when both are configured for a smaller number of ports, and up to thirty percent lower for the highest number of ports.

7100 Device Connectivity

	Token Ring	Ethernet	Coax Devices	ASCII Devices/ LAN Users	Hosts (as gateway)	Hosts (from downstream)
10L	Yes	No	128	34	4 (2 local)	n/a
20L	Yes	No	64	10	4 (1 local)	n/a
10R	Yes	Yes	128	34	4 remote	8
20R	Yes	Yes	64	10	4 remote	8
60R	Yes	Yes	32	3	2 remote	8

Table 6

McDATA is the first vendor other than IBM to announce support for ESCON hosts, which builds on McDATA's strength in IBM channel technology. The 7100 can be upgraded from bus and tag cables to ESCON fiber; IBM controllers do not support such upgrades. The 7100 model 10L, which can support two local hosts, can mix ESCON fiber channel support with bus and tag. McDATA's ESCON will be delivered in the first half of 1992.

LU 6.2 and node Type 2.1 support is provided with the 7100 family. This capability will support for McDATA's PC-based Central Site Customization (in late 1991) and Central Site Change Management (in 1992). These will allow users to crossload customization data and software from a PC anywhere on the network.

McDATA, a privately held company, has been struggling in the past year to recover from both the slowdown in the traditional 3270 terminal market and the loss of two significant U.S. OEM customers, Courier and Memorex, each of which were bought by companies which made IBM controllers. This resulted in the loss of about a third of the company's revenue between 1989 ($80 million) and 1990 ($55 million), though McDATA turned a profitable quarter in the last quarter of 1990. The company maintains strong ties with its non-U.S. partners, and has been broadening its product base from its heavy reliance on 3270 controllers. ■