Digital Equipment Corporation Digital Network Architecture (DNA)

In this report:

Synopsis

Editor's Note
Digital Equipment Corporation bases its communications capabilities among its own computers and with those of other vendors on the Digital Network Architecture (DNA). The company introduced DNA in 1975, and the architecture has evolved to its current Phase V implementation.

Report Highlights
Digital Equipment Corporation's Digital Network Architecture (DNA) defines the standards used by computer hardware and operating systems for interfaces, protocols, and communications functions within a Digital network. DNA is fashioned similarly to the seven-layer Open Systems Interconnection (OSI) reference model. DECnet is the means by which DNA hardware and software are implemented into a network.

The hierarchical, layered architecture of DNA accommodates the substitution and addition of new technologies in future phases while maintaining compatibility with previous phases. This arrangement protects a customer's application-level software investment as Digital adds products and capabilities from one phase of DNA to another.

This report traces the history of the five phases of DNA, stressing the importance of the most recent release, Phase V, which supports multivendor networking and makes allowances for the distribution of large computer networks. Table 1 shows the evolution of DNA to DNA/OSI Phase V.

The report also includes detailed descriptions of the key protocols that implement DNA: Digital Data Communications Message Protocol (DDCMP), Network Services Protocol (NSP), and Data Access Protocol (DAP). DDCMP supports the DNA Data Link Layer and controls the operation of the physical link between nodes. NSP performs the DNA Session Control and Network Services layers, and DAP, which resides in the Network Application Layer of DNA, uses the logical link services of the End-to-End Communications layer.

Analysis

DNA is Digital Equipment Corporation's overall networking architecture: its master plan for communications among Digital and multivendor computers. As an architecture, DNA outlines a broad range of hardware, software, and protocol implementations designed to connect multivendor computers and peripheral devices together. Introduced in 1975, DNA is currently in its fifth phase of implementation, called Phase V/OSI.

The OSI designation in Phase V reinforces Digital's commitment to migrate to the ISO/OSI standards as they develop. At the Physical and Data Link levels of OSI, Phase V implementations cover Digital Data Communications Message Protocol (DDCMP) and High-Level Data-Link Control (HDLC). DDCMP, designed in 1974 for DECnet, is a byte-oriented Data Link protocol, operating over asynchronous and synchronous links. HDLC support, at the Physical layer, covers balanced- or normal-mode operations, option negotiation, extended frame sequence numbering, error detection (32 CRC), and a subset for X.25 Level 2.

The Network layer covers ISO services and protocols, including ISO Connectionless service, ISO Connection-Oriented Service (over X.25), IS-IS Routing protocol, and ISO ES-IS Routing protocol. The Transport layer contains the DECnet Transport (NSP) and OSI Transport protocols. NSP, first defined in Phase 1, is in Phase V for compatibility with Phase IV. The upper layers, 5, 6, and 7, follow two tracks: applications specific to Digital follow the DNA Session Control layer path, while industry standardized applications follow the OSI Session, Presentation, and Application protocol tracks. See Table 1 for a summary of Phase V implementations, OSI layers and products, and DNA Phase IV layers.

DNA Phases

Since the introduction of DECnet *Phase I* in 1976, each new phase of DNA has included new features that extend its networking capabilities. Figure 1 illustrates each of the phases of DNA and describes the associated capabilities. The functions and protocols supported by each phase of DNA are implemented by an equivalent phase of DECnet layered software products. Specific DECnet versions run on each of Digital's operating systems.

Phase II DECnet products offered point-to-point communications between dissimilar Digital computers (e.g., between 16-bit PDP-11s and 32-bit VAXs). In Phase II, certain systems could transfer files to or from a remote node and could access files, peripherals, and other resources at remote sites as if they were parts of local systems. In addition, a remote command submission capability allowed one system to direct another system to execute a program, completely transparent to the user. In Phase II, however, only adjacent nodes could communicate directly with one another.

Phase III established direct logical connections between two adjacent or nonadjacent nodes in any network. Phase III DECnet products supported dynamic adaptive message routing, which passed information to nonadjacent nodes over the most cost-effective route; network command terminal capability, which enabled a terminal attached to one processor in the network to logically connect to another node with the same operating system; multipoint communications facility, which allowed a designated master processor to control multiple tributary systems on a single communications line; and network management functions, which permitted central or distributed control.

Phase III nodes operated as either "full function" or "end node only." Either variety could initiate messages to, or receive messages from, any node in the network. A full-function node had routing capabilities and forwarded a message from one node to another. An end node did not have routing capabilities.

Phase III's capability for least-cost routing allowed the user to assign line costs for each line in the network. The same line could have more than one path cost, depending on which participating node was sending the message. Each full-function node calculated and kept a table of the path costs to each of the other nodes in the network. If a node or a line in the least-cost path went down, the network automatically and transparently routed the traffic over the path with the next lowest cost. If a failure occurred during transmission of sequenced

Table 1. Evolution of Digital Network Architecture to DNA/OSI Phase V

OSI Layers	DNA Phase IV Layers	OSI Products	DNA/OSI Phase V Functions		
7 Application	User Nertwork Management	• OSI File Transfer (FTAM) • Electronic Mail (X.400)	• Networked Office Systems • Videotex • Electronic Mail • Computer Conferencing • Remote Data Base • File Transfer • Virtual Terminal • Network Management • SNA Interconnection • DECnet System Services • Others DECnet Session Control	• OSI FTAM • CCITT X.400 • Others	M A N A G E M E N T
6 Presentation	Network Application	OSI Applications Kernel (OSAK)		OSI Presentation	
5 Session	DECnet Session Control			OSI Session	
4 Transport	DECnet Transport (NSP)	VAX OSI Transport (VOTS)	Common Transport Interface DECnet Transport (NSP)	OSI Transport	
3 Network	DECnet Routing (Adaptive Routing)		• ISO Connectionless Service • ISO Connection-Oriented Service (over X.25) • ISO ES-IS Routing Protocol • IS-IS Routing Protocol		
2 Data Link	Data Link	• X.25	• DDCMP • X.25 • OSI-Standard Ethernet • HDLC		
1 Physical	Physical Link	• OSI-Standard Ethernet			

messages, the network performed automatic recovery procedures so that the destination processor or user received all of the messages intact.

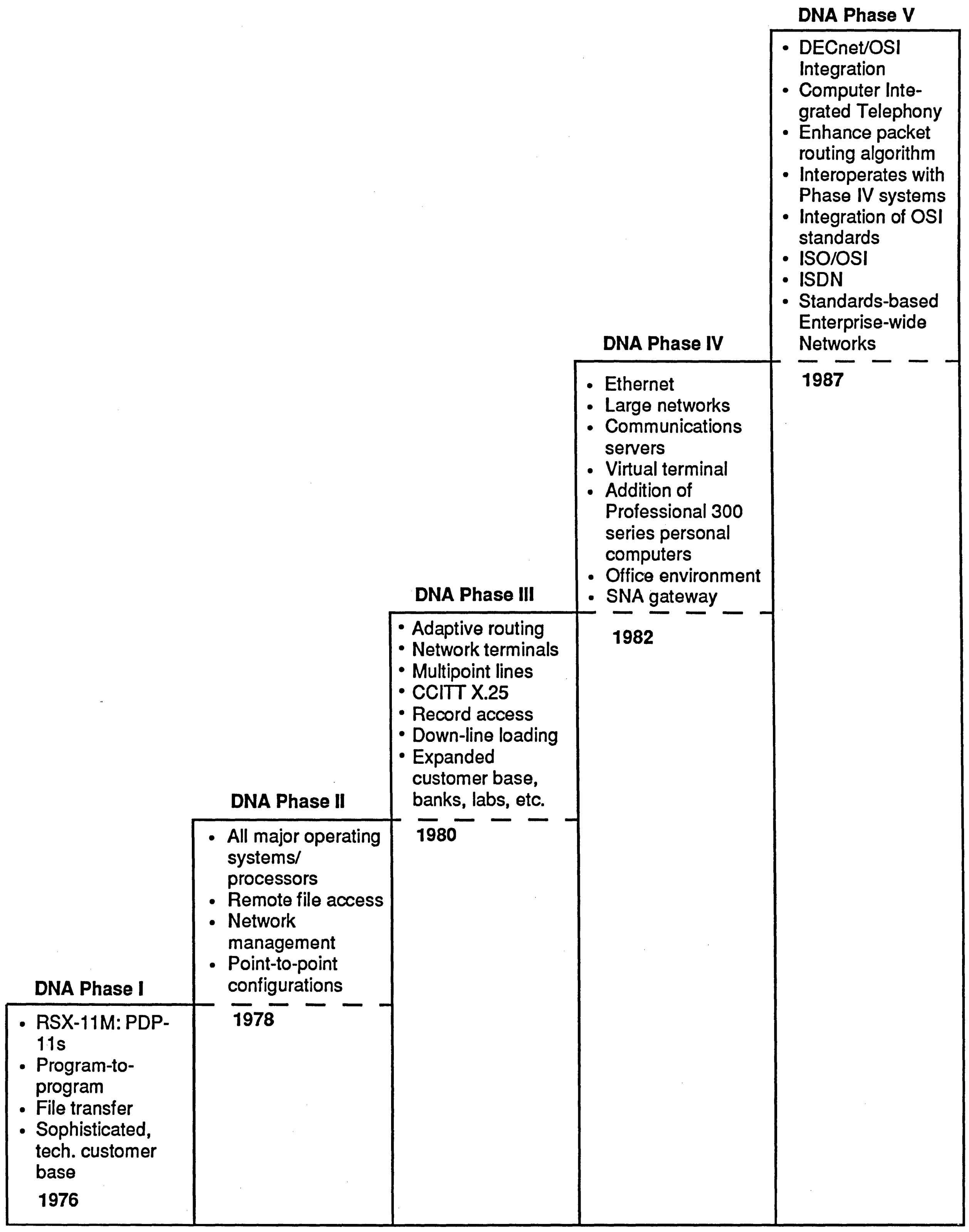
Phase III's network command terminal function (called a "virtual terminal" function by some vendors) allowed a terminal connected physically to one Phase III node to establish a logical connection with another (adjacent or nonadjacent) Phase III node and to operate interactively with that node as if the two were physically connected. Both had to use similar operating systems.

A polling technique operated within Phase III's multipoint capability. One station (a computer) controlled network traffic by polling the slave stations for messages being sent. The system routed each message through the control station. All Phase III capabilities, except for downline system loading, functioned on multipoint lines.

Phase III's network management programs monitored and controlled network operations by tuning parameters, logging events, and testing network components. The programs automatically gathered and displayed statistical information on network activity such as the number of data blocks received with errors, and the number of times the status of a link changed. A loopback testing feature allowed the network manager to test individual nodes, lines, modems, and communications interfaces. Phase III network management functions operated locally or from remote sites.

In May 1982, Digital announced *Phase IV* of DNA, which incorporated capabilities that established Digital as a supplier of products for multivendor network environments. In addition to backward compatibility with Phase III, Phase IV features included data link independence to increase data link options for customers; dedicated server products that off-load communications overhead from general-purpose nodes; increased network routing support, allowing over 64,000

Figure 1.
The Phases of DNA and Associated Capabilities



nodes to be configured into the network; enhanced network management capabilities; gateway support for communications with nodes using X.25 protocol over a packet switched data network; and gateway support for communications with IBM systems in a Systems Network Architecture (SNA) network.

Phase V, compatible with existing phases, debuted in 1987 and is still evolving. It will serve as Digital's plan for the future. Phase V supports multivendor networking and the distribution of large computer networks. Phase V products provide a migration path into OSI and serve as Digital's platform for building global networks.

DNA Characteristics

DNA is an open-ended architecture designed to absorb new and increasingly efficient communications technology. The architecture adapts to diverse situations and supports a broad range of user applications and functions. Because Digital adheres to this network architecture in the design of all communications products, users realize several important advantages.

To keep up with advances in communications technology, future phases of DNA may model additional layers, additional modules within layers, or alternative modules for certain layers. Because of the wide range of DNA facilities, organizations can implement networks tailored to meet their specific application needs.

The network architecture also specifies common communication mechanisms and user interfaces required by different types of computer systems during communications. The networking standard facilitates data exchange and resource sharing among operating systems, communications devices, and computer hardware.

Network management, error recording, and maintenance are simplified with standard procedures for error detection, recording, isolation, recovery, and repair. The architecture provides the needed standardization to manage the network consistently across different systems and to manage unattended remote nodes from a local node, even if they are running on a different operating system than the local node.

A DNA network consists of two or more processors, each loaded with a DECnet software product compatible with its operating system. Each DNA interprocessor operation uses a layered architecture and a common set of DECnet protocols. The range of functions that can be performed between two nodes is limited to the functions they share.

Layered Architecture

DNA includes the specifications that govern the interrelationship of software components making up DECnet. DNA defines both protocols and interfaces. Interfaces are definitions of specific functional boundaries between DECnet software components within a single node. Protocols specify the relationship between equivalent DNA layers in separate DECnet nodes. Refer to Figure 2 for a comparison between DNA's eight protocol layers and the ISO Network Layers and for information about DNA's functions and capabilities.

A message sent from one application to another passes through each layer twice—once in departing the local processor and again after entering the remote processor. A message originates at the *User layer*, where the heart of the message is generated by an application program or as the result of direct input from a user. This data can be a command or an extensive file, destined for a remote processor and application. The most common services accessed by users at this level include resource sharing, file transfers, remote file access, database management, and network management.

The *Network Management layer* supplies routines that define the functions used by operators and programs to plan, control, and maintain the operation of the network. Unlike other layers, this layer interfaces with all the DNA layers below it to gather data on network performance and to change network operational parameters. The Network Information and Control Exchange (NICE) protocol actually performs the services of the Network Management layer. NICE defines the means for the exchange of network, node, and configuration data and handles requests from modules residing in the layer.

The user data then passes to the *Network Application layer*, which appends control information about the message's function. Modules in this layer include remote file access modules, remote file transfer utility, remote system loader, and a network command terminal function. The remote file access and file transfer functions performed in the Network Application layer occur through the Data

Figure 2.
Digital Network Architecture (DNA) and ISO Network Architecture Layers

ISO Layers	DNA Layers	DNA Functions	DNA Capabilities
Application	User	Contains user supplied functions and programs.	Network Applications
	Network Management	Uses Network Information and Control Exchange (NICE) protocol for tasks such as downline loading, upline dumping and testing.	Peer-to-peer Network Management
Presentation	Network Application	Uses Data Access Protocol (DAP) for remote file access and transfer. Uses loopback Mirror Protocol for loopback testing.	Generic Network Functions, File Access, and Terminal Access
Session	Session Control	Uses Session Control Protocol for sending and receiving logical link data and disconnecting logical links.	Network/Node Addressing/Naming, Access Control, Name-to-Address Translation.
Transport	End to End Communication	Uses Network Services Protocol (NSP) to handle all system-independent aspects of creating logical links, such as data flow control.	Task-to-Task Communication, Logical Link handling, End-to-End Error control, Flow control, and Segmentation.
Network	Routing	Uses Transport Protocol to route user data in packets to its destination and control congestion on the lines.	Dynamic Message Routing, Packetization Congestion Control
Data Link	Data Link	Uses DDCMP, X.25, or Ethernet protocol. Makes the connection between adjacent nodes and ensures accuracy and sequencing of messages.	LAN/WAN Interconnect, data integrity.
Physical Link	Physical Link	Manages the physical transmission of data over a channel. Functions include monitoring channel signals, clocking on a channel, and handling interrupts from the hardware.	Communications facilities interface physical transmission of data.

Access Protocol (DAP), indigenous to each DECnet module. This layer also defines a universal input/output (I/O) language within DNA to simplify resource sharing among heterogeneous systems.

The *Session Control layer* and the *End-to-End Communications layer* (previously known as the Network Services layer) work together to form the logical link that allows a program or user in one node to communicate with a program or user in another node. Their combined functions resemble those of the telecommunications access method in a large mainframe. In Phase III and Phase IV products, the Session Control layer defines the local aspects of logical link management (name-to-address translation, process addressing, and access control), and the End-to-End Communications layer handles the creation and management of system-independent aspects of communication, such as connection management, logical links, data flow control, end-to-end error control, and segmentation/reassembly of user messages. As in other layers, additional control information is appended to the message as part of these functions. The Network Service Protocol (NSP) performs the functions of these layers.

The *Routing layer* handles message addressing. It extracts the address and routing information necessary to move the message through the network to the destination node from a predefined network parameter table. Under Phases III and IV, store-and-forward message switching takes place on a dynamically adaptive, least-cost routing basis. Nodes support different degrees of routing, depending on the type and position of each node within the network. Modules in this layer also handle network congestion control.

The *Data Link layer* ensures the message's correct transmission to an adjacent node, either the message's destination node or an intermediate routing node. The Data Link layer encapsulates the message into a frame for transmission across the physical network. DNA supports three different protocols at the Data Link layer: Digital Data Communications Message Protocol (DDCMP), for Digital-to-Digital communications over public or leased transmission lines; Ethernet, for local high-speed transmission over coaxial cable; and X.25, for transmission over public packet-switching networks. Certain network management functions, such as downline loading, upline dumping, and testing connections, use a fourth protocol: Maintenance Operation Protocol.

The *Physical Link layer* defines the manner in which device drivers and communications hardware (modems and lines, for example) should move data over a transmission line. The functions of the modules in this layer include monitoring channel signals, clocking on the channel, handling interrupts from the hardware, and informing the Data Link layer when a transmission has been completed.

Each of these layers resides in every DECnet node but can function differently with various operating systems. The layered architecture permits simple modification or expansion of a specific layer, without requiring changes to all the others. Each layer contains one or more software modules. Users can implement only those modules that provide the functions desired for a specific node.

All the DNA protocols are common among all DECnet nodes. Though each node may not contain the same software modules, these common protocols allow system-independent networking.

The protocols implement the layered architecture. As data passes from the User layer to the Data Link layer, protocol modules in each layer add some control information to the data received from the next higher module. Each DNA module protocol performs only the processing associated with the functions of the layer in which it resides. Since it does not mix the information between layers, the network is sufficiently flexible to add and delete protocols or to modify existing ones.

Digital Data Communications Message Protocol (DDCMP)

DDCMP supports the DNA Data Link layer and controls the operation of the physical link between nodes while maintaining the integrity and correct sequence of the transmitted data.

DDCMP is a byte-oriented link protocol that supports half- or full-duplex transmissions over point-to-point or multipoint lines in a DNA network. Its structure is similar to IBM's byte-oriented Binary Synchronous Communications (BSC) protocol.

With all control information appended, DDCMP groups data into blocks that comprise complete messages. The user determines the maximum message length, which, once determined, remains fixed between any two nodes. The maximum message size depends on several criteria including the reliability and quality of the transmission facility, response time required, type of processing performed (batch, interactive), and the size of allocated buffer space.

DDCMP assigns each message a number to ensure proper sequencing at the receiving node. Typically, the receiving node acknowledges the correct reception of a series of data messages by returning the sequence number of the last received message as a response. Users can define up to 225 messages to be sent before a response is required. Sending more messages before acknowledgment can reduce line turnaround overhead (if received correctly). DDCMP, however, uses retransmission to recover from errors. The DDCMP error recovery mechanism uses time-outs and control messages to resynchronize and trigger retransmission.

DDCMP ensures data integrity and correct reception through the use of a cyclic redundancy check (CRC), a two-byte binary polynomial generated by an algorithm common to all network nodes and appended to the ends of all messages. DDCMP typically uses one CRC field after the header/control information and another at the end of the data field.

DDCMP differs in several ways from bit-oriented protocols, such as SDLC and HDLC, offered by other major vendors. The byte-oriented DDCMP relies on specific message fields for the placement of control information and is, therefore, insensitive by design to patterns in the data that can accidentally resemble control signals. Bit-oriented protocols are characterized by variable message lengths, which can transmit data that may contain bit patterns resembling control information. To remedy this problem, bit-oriented protocols typically use a zero-insertion technique to break up such bit patterns.

Because of fixed block lengths and acknowledgments in DDCMP messages, these messages generate more overhead than bit-oriented protocols, particularly in short messages. DDCMP includes operational techniques, however, that reduce much of this inherent overhead and ensure optimal line use.

Procedures that permit negative acknowledgment of a sequenced message to imply positive receipt of a previous message reduce the number of message-by-message acknowledgments required. Message acknowledgments can also be included in a return data message.

Unlike Binary Synchronous Communications (BSC) protocol, DDCMP contains a special maintenance message format for diagnostic testing, dumping, and bootstrapping.

In addition to maintenance message and data message formats, DCCMP also uses control messages, which are usually short, unnumbered messages used during link start-up or reinitialization, as well as for the acknowledgment of individual messages. The first byte (eight-bit field) of any DDCMP message tells the receiving device whether the incoming message contains control, data, or maintenance information. DDCMP precedes this control header with characters that establish synchronization between both ends of the communications link.

CCITT X.25

In Phase IV, Digital also incorporated Levels 1, 2, and 3 of the CCITT X.25 recommendation for the connection of communication equipment to a public or private packet-switching network. Digital has implemented Level 1 of X.25, which defines mechanical, electrical, functional, and procedural characteristics in the Physical Link layer; Level 2, which defines the format for a frame of data analogous to the Ethernet frame at the Data Link layer; and Level 3, which defines a packet format for the establishment of virtual circuits between two end users at the Data Link layer. Messages sent along a virtual circuit are numbered sequentially (as are DDCMP messages) to ensure transmission without error or duplication.

Network Services Protocol (NSP)

The Network Services Protocol (NSP) performs the functions of the Session Control and Network Services layers of the DNA architecture. NSP establishes, maintains, and terminates communications links between users and/or application programs. NSP performs the "dialog" or "handshake" process by which the NSP module in one node converses with a corresponding NSP module in another node to establish a logical link between the users or programs that want to communicate. The conversation consists of an exchange of parameters and control messages that establish the link; it provides for maintenance of the link and eventual disconnection. NSP also determines the storage area required, if applicable, for the data to be transmitted and message sequencing information, including segment size.

Once a link has been established, data messages can flow freely between the sending application (whether user or program) and the destination application. Messages between applications usually contain application data. The sending NSP takes the entire data block to be sent and, unless the block is small enough to be a single message, breaks the message into segments for sequential transmission; the receiving NSP reassembles the segments before delivering them to the end user or program. In the case of direct terminal input, the user can control this function.

Before sending any data, the transmitting NSP examines the control parameters it has received from the other NSP, and establishes how it should acknowledge any sequenced message. This process occurs through an ACK/NAK procedure both before and during data transmission. If it lacks sufficient buffer space, the receiver can reject any subsequent message segments by returning a NAK (negative acknowledgment) to the transmitter.

The NSP-to-NSP logical link uses a separate, dedicated subchannel for the transmission of link service and interrupt messages. Messages on this subchannel are numbered and acknowledged separately from data channel messages. Link service messages exchange information used to control the flow of data. Interrupt messages carry high-priority information, such as alarm conditions. At any time while the link is established, the receiver can request interrupt messages, allow or stop data flow, or request data.

The logical modules that have been connected by the NSPs initiate disconnection of the logical link. Either party can abort or terminate conversation. Once this has been done, and the NSPs have been properly notified, they will disconnect the link.

Data Access Protocol (DAP)

The Data Access Protocol (DAP), residing in the Network Application layer of the DNA architecture, uses the logical link services of the End-to-End Communications layer. DAP defines a set of messages that control the execution of remote file access and outline procedures to accomplish specific file applications. To create a file, an accessing program and a remote File Access Listener (FAL) must exchange a subset of DAP messages in a prescribed sequence.

DAP supports file transfer between heterogeneous file systems and supports sequential, relative, and indexed file organizations. It retrieves a file from an input device such as a disk, card reader, or terminal and stores a file on an output device such as a tape drive, line printer, or terminal. It supports sequential, random, and indexed sequential (ISAM) record access. DAP lists directories of remote files and supports the deleting and renaming of these files. It allows multiple datastreams to travel over a logical link, recovers from transient errors, and reports fatal errors to the user. DAP also submits and executes remote command files. It permits wild card file specification for command file execution, sequential file retrieval, file deletion, and file renaming. It also permits an optional file checksum to ensure file integrity.

In a typical DAP dialog, the first message exchange consists of configuration information—the source operating system and file system, target nodes, and buffer size. *File Attributes* messages then supply information about the file to be accessed. Following these messages, the *Access Request* message opens a particular file. If data is transferred to the accessing program, a datastream is set up at this point. For both sequential and random access file transfer, one control message sets up the datastream. DAP has a file transfer mode that eliminates the need for DAP control messages once the flow of file data begins. When the file transfer is complete, *Access Complete* messages terminate the datastream.

DAP minimizes protocol overhead. Whenever possible, defaults are specified for fields. Relatively small records can be blocked together and sent as one large message.

A user program does not handle DAP directly. All DECnet implementations include system software that can send and receive DAP messages on behalf of user programs. DAP-speaking DECnet modules use a unique set of messages to accomplish remote file access and transfer. ■