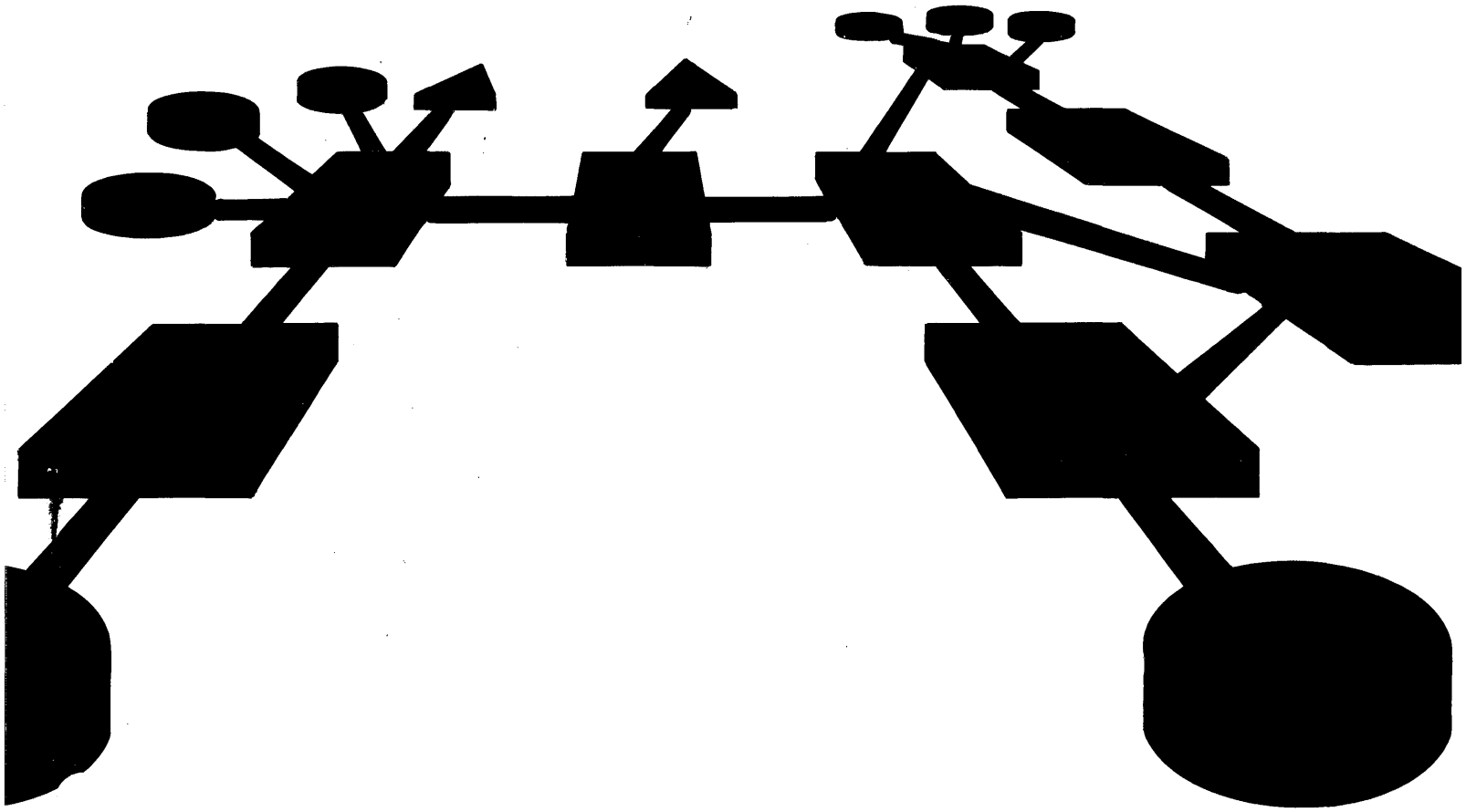
Order No. AA-K179A-TK

DECnet
DIGITAL Network Architecture

General Description



DECnet
DIGITAL Network Architecture (Phase III)

General Description

Order No. AA-K179A-TK

October 1980

This document describes the design of the DIGITAL Network Architecture that serves as a model for Phase III DECnet implementations. It includes descriptions of functions, protocol messages, and operation.

To order additional copies of this document, contact your local Digital Equipment Corporation Sales Office.

digital equipment corporation · maynard, massachusetts

First Printing, October 1980

The postage-prepaid READER'S COMMENTS form on the last page of this document requests the user's critical evaluation to assist us in preparing future documentation.

The following are trademarks of Digital Equipment Corporation:

DIGITAL	DECsystem-10	MASSBUS
DEC	DECtape	OMNIBUS
PDP	DIBOL	OS/8
DECUS	EDUSYSTEM	PHA
UNIBUS	FLIP CHIP	RSTS
COMPUTER LABS	FOCAL	RSX
COMTEX	INDAC	TYPESET-8
DDT	LAB-8	TYPESET-11
DECCOMM	DECSYSTEM-20	TMS-11
ASSIST-11	RTS-8	ITPS-10
VAX	VMS	SBI
DECnet	IAS	PDT
DATATRIEVE	TRAX	

Contents

Chapter 4 The Network Services Layer

Chapter 5 The Session Control Layer

Chapter 6 The Network Application Layer

Chapter 7 Network Management

Glossary

Figures

Tables

Preface

This document is an overview of the DIGITAL Network Architecture (DNA). DNA is a model of structure and function upon which DECnet implementations are based. DECnet is a family of communications software and hardware products that enable DIGITAL operating systems and computers to function in a DECnet network.

A DECnet network is a group of DIGITAL computer systems with associated operating systems, DECnet software, and communication hardware that are connected to each other by physical channels or lines. Each computer in a network containing a DECnet implementation is called a *node*. The *network* therefore consists of connected nodes and lines. DNA defines standard protocols, interfaces and functions that enable DECnet network nodes to share data and access each others' resources, programs, and functions.

This document describes Phase III DNA. Phase III DECnet implementations include DECnet–11M, DECnet–11M–PLUS, and DECnet–11S. Within the next two years, DECnet/VAX, DECnet/E, DECnet–10, DECnet–20, and DECnet–RT will also support Phase III. The previous version of DECnet, Phase II, is compatible with Phase III. However, Phase II nodes can provide Phase II functions only.

DNA supports a broad range of applications and a variety of network topologies. (A network *topology* is a particular configuration of nodes and lines.) User documentation and marketing brochures describe in detail various types of applications and specific programming and network management information.

This document summarizes the design and structure of DNA, and serves as an introduction to the DNA functional specifications. It is intended for readers with a knowledge of communications technology who desire an understanding of the overall DNA structure. A glossary at the end of the document defines many DNA terms. Additionally, many DNA terms are italicized and explained in the text at their first occurrence.

The DNA functional specifications, containing the architectural details of DNA, are as follows:

DNA Data Access Protocol (DAP) Functional Specification, Version 5.6.0, Order No. AA–K177A–TK

DNA Digital Data Communications Message Protocol (DDCMP) Functional Specification, Version 4.1.0, Order No. AA–K175A–TK

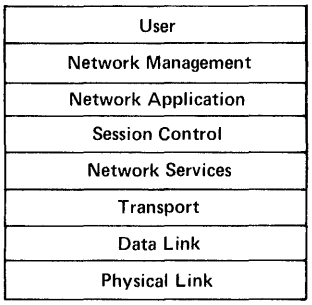
DNA Maintenance Operations Protocol (MOP) Functional Specification, Version 2.1.0, Order No. AA–K178A–TK

DNA Network Management Functional Specification, Version 2.0.0, Order No. AA-K181A-TK

DNA Network Services (NSP) Functional Specification, Version 3.2.0, Order No. AA-K176A-TK

DNA Session Control Functional Specification, Version 1.0.0, Order No. AA-K182A-TK

DNA Transport Functional Specification, Version 1.3.0, Order No. AA-K180A-TK



Chapter 1
Introduction

A network architecture specifies common communication mechanisms and user interfaces that computer systems of different types must adhere to when passing data between systems. A network architecture may also be designed so that implementations meet other goals, such as cost effectiveness and flexibility.

A network architecture is necessary for several reasons:

- Communications technology is changing continually. It is necessary to have a structure for networking that can adjust to these changes as effectively as possible.
- A variety of operating systems, communications devices, and computer hardware exists. A common standard of networking to which each operating system or communication hardware facility can adhere is necessary.
- A network, to be most useful and cost effective, should provide a broad range of user applications and functions. This requires very complex software. A network architecture forces such software to have a clean, well-structured design.
- Network management, error recording, and maintenance are easier when implementations provide standard procedures for error detection, recording, isolation, recovery, and repair. An architecture provides this standardization.
- Networks need to be adaptible to different communication situations. A hierarchical, modular architecture enables the substitution of modules of equivalent function to suit the specific needs of a particular network. For example, a data link protocol that ensures data integrity over a leased physical circuit might be replaced by a protocol that allows two systems to communicate over a public data network.
- A network architecture enables system-independent functions to be designed specifically enough to be implemented in hardware. This can improve system performance. The DMC11 communications device, for example, contains the equivalent of the DDCMP data link modules that ensure data integrity.

Basically, DIGITAL Network Architecture is the specification of a layered hierarchy in which each layer contains modules that perform defined functions. The architecture specifies the functions of these modules and relationships between the modules.

Modules in a layer typically use the services of modules in the layer immediately below. Some layers may contain more than one type of module, but in this case the modules fall into a more general category of function. For example, user-written programs all operate in the top layer, or User layer. The bottom layer is the most basic layer. It manages the physical transmission of information over a channel.

Each module specified by DNA operates as a *black box*. That is, the operation within the black box is transparent to other layers and to equivalent modules in other nodes. The architecture does not define specific code for implementing modules. It only defines specific operations that the modules must perform. These definitions may take the form of algorithms written in a higher level programming language.

The architecture specifies two kinds of relationships between modules:

- **Interfaces.** Interfaces are the relationships between different modules that are usually in the same node. Typically, a module in one layer interfaces with a module in the layer immediately below to receive a service. The architecture specifies the functions supplied by these interfaces as calls to subroutines. These specifications are functional: they need not be implemented as subroutine calls.
- **Protocols.** Protocols are the relationships between equivalent modules that are usually in different nodes. These protocols consist of messages with specific formats and the rules for exchanging the messages.

A state table is often specified to show the transitions occurring in a black box when protocol message exchanges or "calls" from higher layers take place.

Figure 1–1 shows the basic architectural structure of DNA.

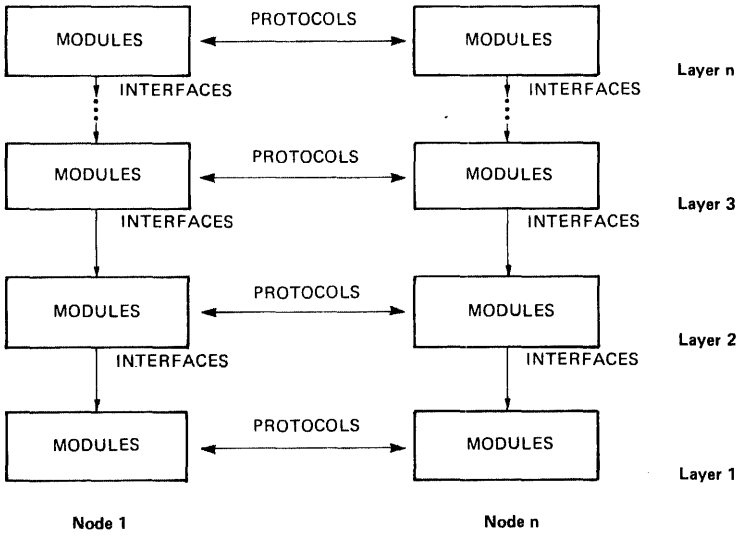


Figure 1–1: Basic DNA Structure

The architecture specifies common error reporting, operational *parameters*, and *counters* that certain layers must maintain. This standardization ensures that maintenance, error logging, and network management can take place consistently and, to some extent, across systems. For example, an operator could manage an unattended remote system from his local node. He could expect to observe or, in some cases, change values of parameters equivalent in function and name to those at his own node, even if the remote node was running under a different operating system.

DNA is an open-ended architecture. Future phases of DNA may model additional layers, additional modules within existing layers, or alternative models for certain layers.

1.1 Design Goals

DNA achieves the following design goals:

Create a common user interface. The application interface to the network is common across the varied implementations. The common interface hides the internal characteristics and topology of the network.

Support a broad spectrum of applications. DECnet networks can:

- Share resources.
- Distribute computation.
- Communicate with remote systems efficiently.

Resource sharing activities include remote file access, intercomputer file transfer, distributed data base queries, task-to-task communication, and remote use of peripheral devices. Data moved across the network can be streams of related sequential data and short independent messages.

Distributed computation means cooperating programs executing in different computers in a network. Examples are real-time process control (such as a manufacturing system), specialized data base multiprocessor systems (such as order-processing or payroll systems), and distributed processing systems (such as a banking, airline reservation, or combined order-processing/inventory control system).

Remote communication facilities include remote interactive terminals and batch entry/exit stations.

Support a wide range of communication facilities. Such facilities include a variety of asynchronous and synchronous, full- and half-duplex communications devices, some with multipoint (multidrop) and/or multiple controller/multiplex capabilities. (Consult the glossary for definitions.) In addition, DECnet networks support a variety of communication channels, such as leased lines or satellite links.

Be cost effective. A DECnet network application costs about the same as a custom-designed network that achieves the same performance for the same application.

Support a wide range of topologies. DECnet networks support many configurations of nodes and lines. These range from point-to-point or star topologies to hierarchical structures to topologies in which each node has equal control over network operation.

Be highly available. DECnet networks can be configured to maintain operation even if a subset of lines or nodes fail. Because maintenance functions are highly distributed, DECnet networks can recover from operator error unless the error is extreme.

Be extensible. DNA allows for incorporation of future technology changes in hardware and/or software. Moreover, DNA makes possible the movement of functions from software to hardware. DNA also permits subsets of modules.

Be easily implementable. DNA is independent of the internal characteristics of DIGITAL computers and their operating systems. It can be implemented on a wide variety of systems.

Be highly distributed. The major functions of DNA are not centralized in one node in a network.

Implement network control and maintenance functions at a user level. This permits easier system management. For example, terminal commands can set, change or display lower level parameters or counters.

Allow for security. The DNA design allows for security at several levels. User access control and the exchange of passwords at other levels is implemented in most DECnet products.

1.2 Layers, Modules, and Interfaces

DNA defines the following layers, described in order from the highest to the lowest:

User layer. The User layer contains most user-supplied functions. It also contains the Network Control Program, a Network Management module that gives system managers access to lower layers. Chapter 7 describes the function of the Network Control Program, the only DNA-defined User layer module.

Network Management layer. Modules in the Network Management layer provide user control of and access to operational parameters and counters in lower layers. Network Management also performs down-line loading, up-line dumping, and test functions. In addition, Network Management performs event logging functions. This layer is the only one that has direct access to each lower layer. Chapter 7 describes Network Management.

Network Application layer. The Network Application layer provides generic services to the User layer. Services include remote file access, remote file

transfer, and resource managing programs. This layer contains both user- and DIGITAL-supplied modules. Modules execute simultaneously and independently in this layer. Chapter 6 describes the Network Application layer.

Session Control layer. The Session Control layer defines the system-dependent aspects of logical link communication. A *logical link* is a virtual circuit (as opposed to a physical one) on which information flows in two directions. Session Control functions include name to address translation, process addressing, and, in some systems, process activation and access control. Chapter 5 describes the Session Control layer.

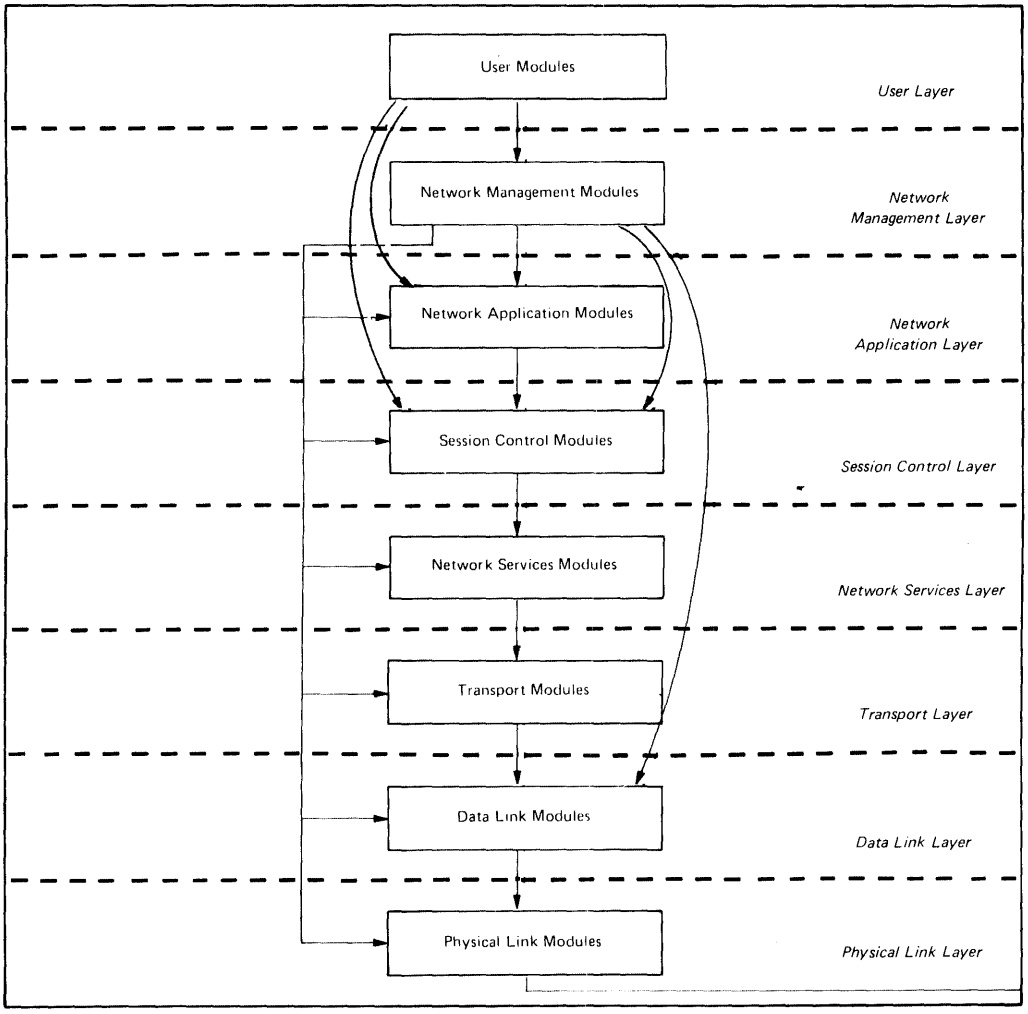
Network Services layer. The Network Services layer is responsible for the system-independent aspects of creating and managing logical links for network users. Network Services modules perform data flow control, end-to-end error control, and the segmentation and reassembly of user messages. Chapter 4 describes the Network Services layer.

Transport layer. Modules in the Transport layer route user data, contained in *packets*, to its destination. Transport modules also provide congestion control and packet lifetime control. Chapter 3 describes the Transport layer.

Data Link layer. Modules in the Data Link layer create a communication path between adjacent nodes. Data Link modules ensure the integrity and correct sequencing of blocks of data transferred across the path. Chapter 2 describes the Data Link layer.

Physical Link layer. Modules in the Physical Link layer manage the physical transmission of data over a channel. The functions of 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 is completed. Implementations of this layer encompass parts of device drivers for each communications device as well as the communication hardware itself. The hardware includes devices, modems, and lines. In this layer, industry standard electrical signal specifications such as *EIA RS-232C* or *CCITT V.24* operate rather than protocols. There is no chapter devoted to this layer since only its Network Management interface, counters, and events are DNA-defined.

Figure 1–2 shows the relationships among the DNA layers in a single node. The three upper layers each interface directly with Session Control for logical link services. Each layer interfaces with the layer directly below to use its services. The User layer interfaces directly with the Network Application layer as well. In addition, Network Management modules interface with each lower layer directly for access and control purposes. Finally, Network Management interfaces directly with the Data Link layer for service functions that do not require logical links.



Black arrows show direct access for control and examination of parameters, counters, etc. Red arrows show interfaces between layers for normal user operations such as file access, down-line load, up-line dump, end-to-end looping, and logical link usage.

Figure 1-2: DNA Layers and Interfaces

Figure 1–3 shows a typical DECnet node containing multiple modules in some layers.

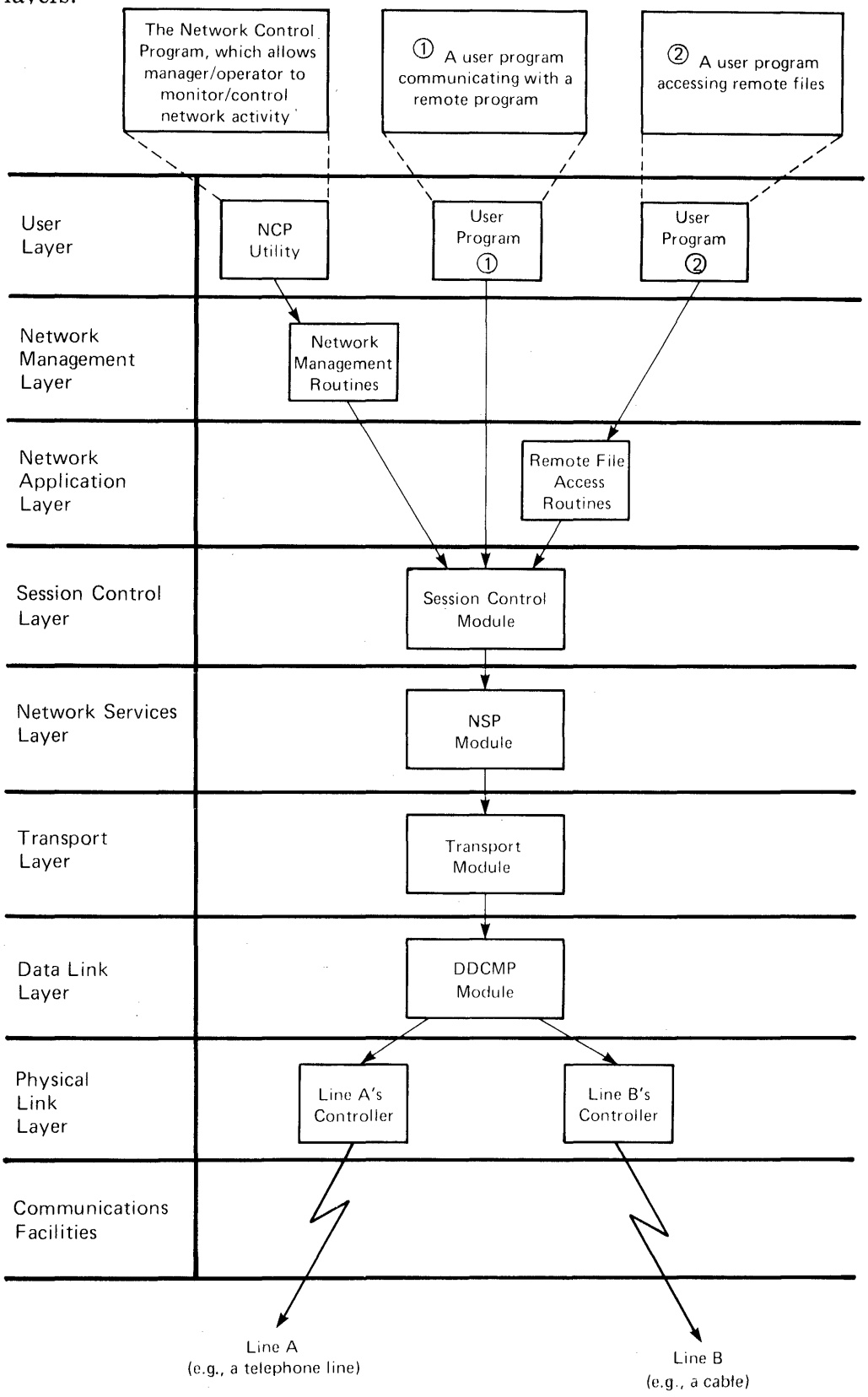


Figure 1–3: DNA Modules Resident in a Typical DECnet Node

1.3 User Services

As shown in Figures 1–2 and 1–3, there are different ways in which modules use the services of other modules in a system. DNA allows the following services:

- **User-to-user.** A user level process in one node communicates via a logical link with a user level process in another node. For example, a MACRO call from the user level might, using a direct interface to the Session Control module that links to a remote node, send data to a user level process in another node.

- **Remote application.** A user-level process uses a Network Application module which, in turn, uses a logical link to perform a function at a remote node. For example, a user-level command causes a Network Application module to transfer a file to a remote node. The Network Application module uses the DAP protocol to communicate with the remote Network Application module.

- **Network Management.** A user-level command or process uses a Network Management module to perform a Network Management service at a remote node. Network Management performs some services using a logical link provided by Session Control and other services by interfacing directly to the Data Link layer. An example of a Network Management operation is a command that displays a remote node's counters at a local terminal.

1.4 Protocols

Modules with equivalent functions in the same layer, but in different nodes, communicate using protocols. A protocol is both a set of messages and the rules for exchanging the messages. The architecture defines the message formats and rules very specifically.

Protocols are necessary for every DNA function that requires communication between nodes. For example, the Data Link layer protocol assigns numbers to transmitted messages and checks the numbers of received messages. In this way, the protocol ensures that transmitted data is received in the correct order.

The DNA-defined protocols, listed according to layer, are:

Network Management Layer

- **Network Information and Control Exchange (NICE) Protocol.** This is used for triggering, down-line loading, up-line dumping, testing, reading parameters and counters, setting parameters, and zeroing counters.
- **Event Logger Protocol.** This is used for recording significant occurrences in lower layers. An *event* could result from a line coming up, a counter reaching a threshold, a node becoming unreachable, and so on.

Network Application Layer

- **The Data Access Protocol (DAP).** This is used for remote file access and transfer.
- **The Loopback Mirror Protocol.** This is used for Network Management logical link loopback tests.

Session Control Layer

- **Session Control Protocol.** This is used for functions such as sending and receiving logical link data; and disconnecting and aborting logical links.

Network Services Layer

- **Network Services Protocol (NSP).** This handles all the system-independent aspects of managing logical links.

Transport Layer

- **Transport Protocol.** This handles routing and congestion control.

Data Link Layer

- **Digital Data Communications Message Protocol (DDCMP).** This ensures integrity and correct sequencing of messages between adjacent nodes.

Some implementations may use the Network Services Protocol (NSP) that creates and manages logical links for network communication within a single node, as well as for internode communication. For example, Network Management modules may use a logical link even when performing local functions.

Figure 1–4 shows protocol communication between two nodes.

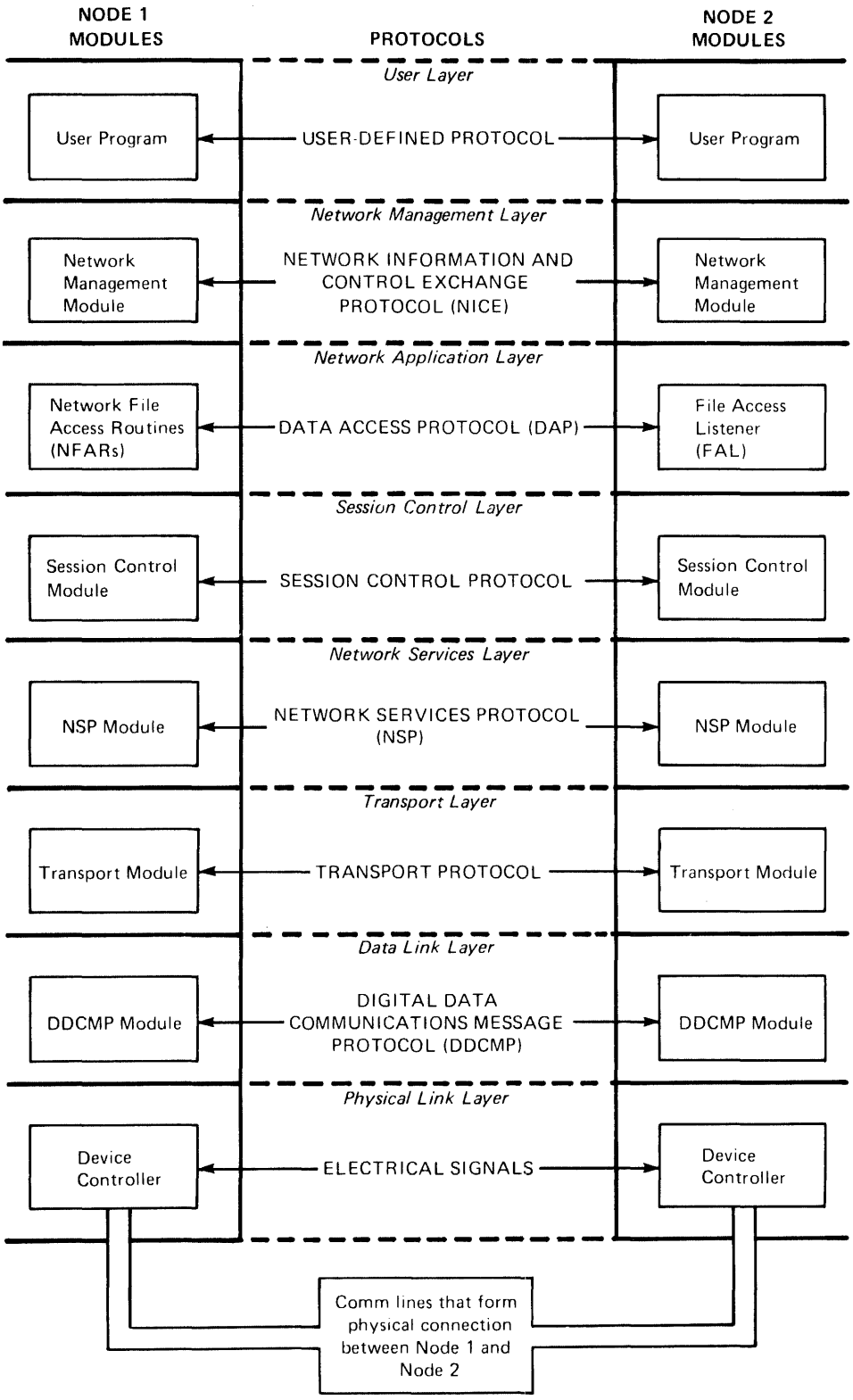


Figure 1–4: Protocol Communication between Two Nodes

1.5 Data Flow

The primary purpose of a network is to pass data from a *source* in one node to a *destination* in another. Because DNA is layered, it is important to understand how data flows through these layers and between nodes. Data travelling from one node in a network to another passes from a source process in the user layer down through each layer of the DNA hierarchy of the source node before being transmitted across a line. If the destination node is not adjacent, the data must then travel up to the Transport layer of the adjacent node, where it is *routed* (or *switched*), sent back down through the two lower layers, and transmitted across the next line in the path. The data keeps travelling in this manner until it reaches its destination node. At this node, the data passes up the hierarchy of layers to the destination process.

Figure 1–5 shows how information is built as data passes through the DNA layers at one node. In this example, Network Management is not involved.

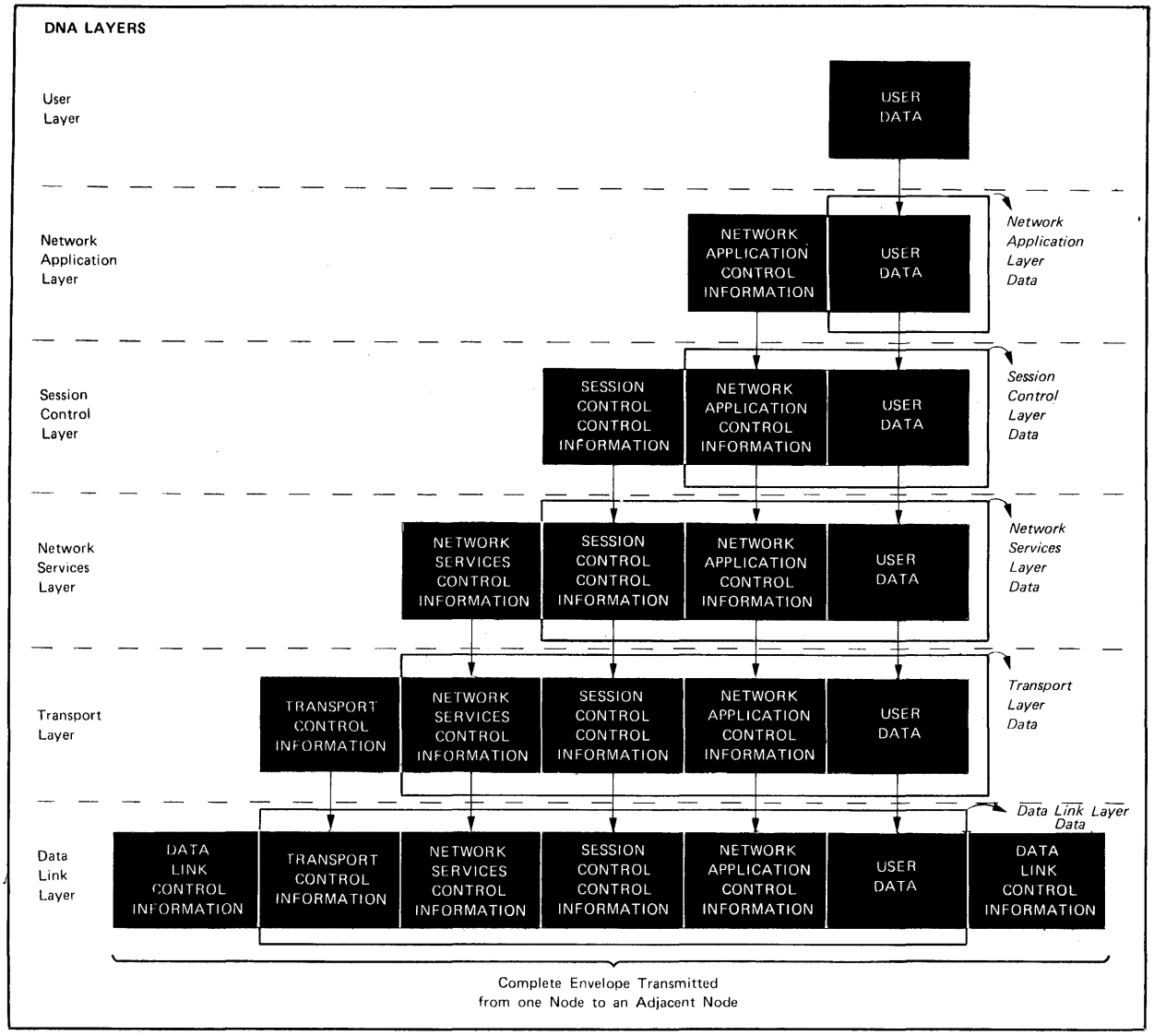


Figure 1–5: Building of Information as Data Passes Through the DNA Layers

In the following scenario a user attempts to form a logical link with another user. The requesting user passes initial connection data to the destination user. The numbered steps correspond with numbers on Figure 1–6 which follows this explanation.

Data Flow at the Source Node

❶ The source user requests a connection to the destination user and passes connect data.

❷ The Session Control module receives the data, maps the destination node name to a numerical address, if necessary, and places the data in the next transmit buffer, adding control information to the message. The message then passes to the Network Services layer.

❸ The Network Services module adds its control information (including a logical link identification), and passes the message (now called a *datagram*) to the Transport layer.

❹ The Transport module adds a header consisting of the destination and source node addresses and selects an outgoing channel for the message based on routing information. Transport then passes the message (now called a *packet*) to the Data Link layer, specifying the outgoing channel address and, if a multipoint link, the station address of the receiving node on the channel.

❺ The Data Link module adds its protocol header consisting of framing and synchronization information, a transmission block sequence number, and a trailer consisting of a cyclic redundancy check (CRC). The packet is now *enveloped* for transmission.

❻ The Physical Link module transmits the enveloped message over the data channel.

Data Flow Across the Network to the Destination Node

❼ The enveloped data message arrives at the next node. The Physical Link module receives the message and passes it to the Data Link layer.

❽ The Data Link module checks the packet for bit errors in transmission. In addition the message number is checked for proper sequence. If there were any errors, the Data Link protocol performs correction procedures. With DDCMP, the procedure is retransmission. The Data Link header and trailer are removed from the message, which is then passed up to the Transport layer.

❾ The Transport layer checks the destination address in the header. If the address is not this node, Transport selects the next outgoing channel from its routing table, and passes the message back to the Data Link layer. The Transport layer has routed the message on to the next line in its path. The message proceeds as in steps ❺ and ❻ above.

❿ The message proceeds through the network, switching at routing nodes, until it arrives at the Transport layer with the same address as the destination address in the message.

Data Flow at the Destination Node

⓫ The packet passes to the Transport layer of the destination node as described in ❼ and ❽ above. The destination Transport module removes the Transport header, and passes the datagram to Network Services.

⓬ The Network Services module examines the Network Services header on the datagram. If it has the resources to form a new logical link, Network Services passes the connect data, without the Network Services control information, to Session Control.

⓭ The Session Control module performs any necessary access control functions and passes the message to the appropriate process in the user layer after removing Session Control header information.

⓮ The destination process interprets the data according to whatever higher level protocol is being used.

Figure 1–6 illustrates the data flow just described.

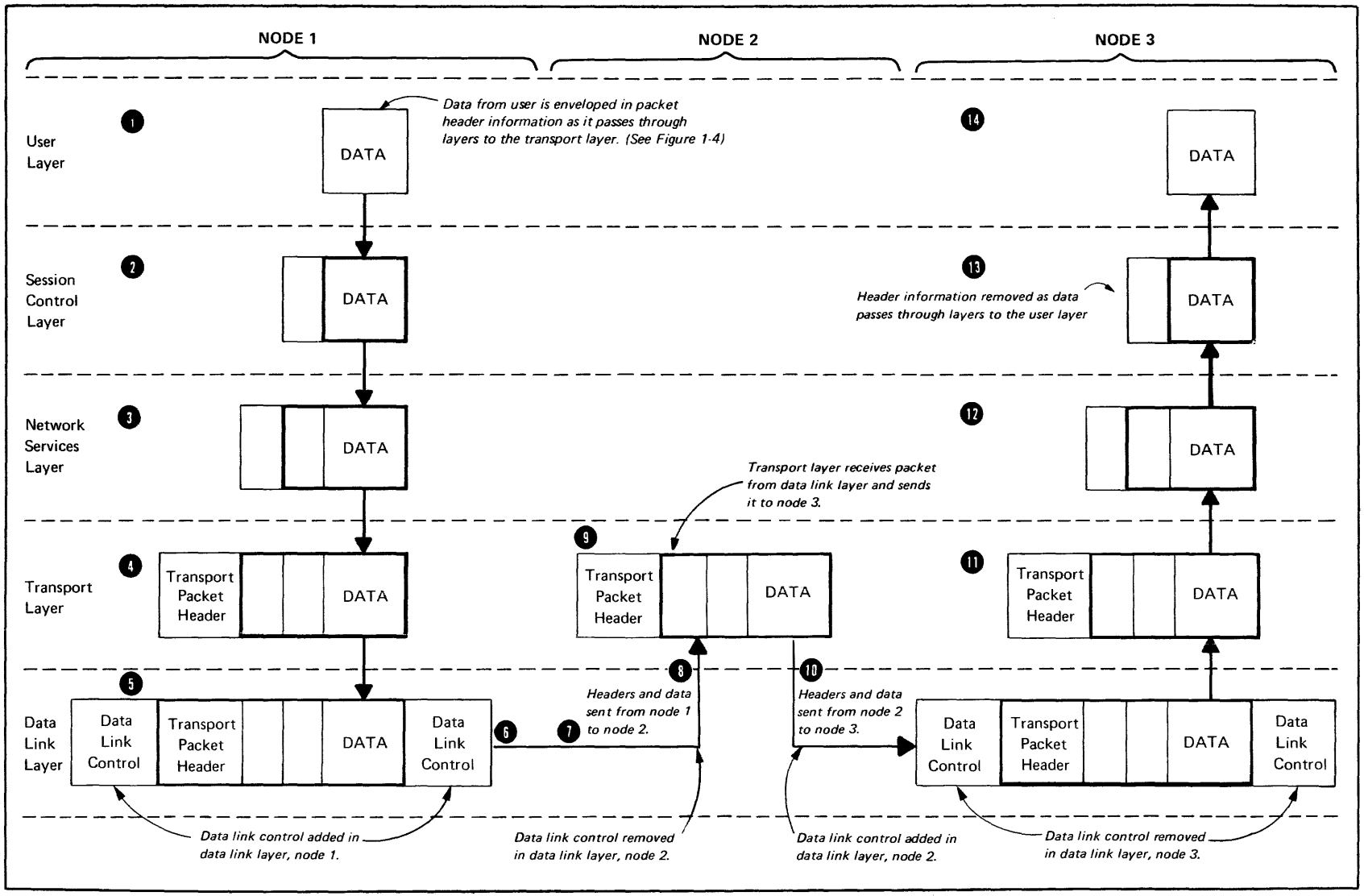
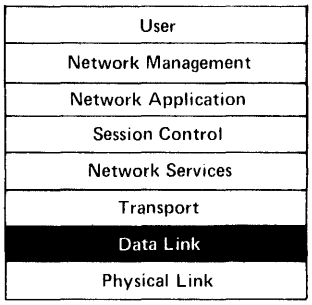


Figure 1-6: Data Flow



Chapter 2
The Data Link Layer

The Data Link layer, residing immediately above the Physical Link layer, is responsible for creating a communications path between adjacent nodes. The Data Link layer frames messages for transmission on the channel connecting the nodes, checks the integrity of received messages, manages the use of channel resources, and ensures the integrity and proper sequence of transmitted data.

Currently there are two DNA protocols concerned with the Data Link layer:

- **Digital Data Communications Message Protocol (DDCMP).** DDCMP provides correct sequencing of data and error control to ensure data integrity. Sections 2.1–2.3 describe DDCMP.
- **Maintenance Operations Protocol (MOP).** MOP usually operates within DDCMP. It provides functions concerned with maintaining a remote and possibly unattended system. Sections 2.4–2.6 describe MOP.

2.1 DDCMP Functional Description

DDCMP is a *byte-oriented* protocol. There are three general types of data link protocols: byte-oriented, character-oriented, and bit-oriented. A byte-oriented protocol provides a count of the number of bytes that will be sent in the data portion of each data message sent. In contrast, a character-oriented protocol uses special ASCII characters to indicate the beginning of a message and the end of a block of text, while a bit-oriented protocol uses flags to frame data, sent in undefined lengths.

DDCMP was designed in 1974 specifically for DNA. DDCMP is functionally similar to HDLC (High-level Data Link Control), the data link protocol adopted in 1975 by the International Standards Organization, although HDLC is a bit-oriented protocol. Another type of data link protocol, binary synchronous, is character-oriented.

DDCMP is a general-purpose protocol: It operates on a variety of communication systems from the very small to the very large. DDCMP makes maximum use of channel bandwidth and handles transparent data efficiently. (*Data transparency* is the capability of receiving, without misinterpretation, data containing bit patterns that resemble protocol control characters. Character-oriented protocols cannot handle transparent data as efficiently as byte- or bit-oriented protocols.) Other major goals of the DDCMP design include error recording, to prevent channel failure on degraded lines, and the provision of a basic mode for bootstrapping and testing functions.

DDCMP transmits data grouped into physical blocks known as data messages. DDCMP provides a mechanism for exchanging error-free messages. A general description of how this mechanism works follows:

DDCMP assigns a number to each data message beginning with number one (after each initialization) and incremented by one (modulo 256) for each subsequent data message. DDCMP places a 16-bit cyclic redundancy check (CRC–16) error detection polynomial at the end of each data message transmitted.

The receiving DDCMP module checks for errors and, if there are none, returns the message number with a positive acknowledgment of message receipt.

The receiving DDCMP need not acknowledge each message sent; acknowledgment of data message *n* implies acknowledgment of all data messages sent up to and including data message *n*.

If an error is detected by the receiving DDCMP, it uses time-outs and control messages to resynchronize and trigger retransmission.

The principal DDCMP features are as follows:

- Obtains data from the Physical Link layer in blocks consisting of 8-bit bytes.
- Sequences data by message numbers.
- *Pipelines* up to 255 messages. That is, it sends messages without waiting for acknowledgment of each successive message.
- Operates independently of channel bit width (serial or parallel) and transmission characteristics (asynchronous or synchronous).
- Operates with a wide variety of communication hardware and modems.
- Detects errors by means of CRC–16 message trailers.
- Retransmits to correct errors.
- Achieves optimum performance with techniques such as pipelining, *piggybacking* (that is, sending an acknowledgment within a returned data message), and implying a positive acknowledgment of previous messages by a negative acknowledgment of current message.
- Operates in both half-duplex and full-duplex modes.

- Supports point-to-point and multipoint communications.
- Synchronizes transmission and reception on byte and message level.
- Frames (envelopes) data messages.
- Provides a *maintenance mode* for diagnostic testing and bootstrapping functions.
- Provides data transparency.
- Notifies the other end of the link when restarting or initializing.
- Maintains error counters.
- Records the occurrence of events for automatic error reporting to the user.

2.2 DDCMP Messages

There are three types of DDCMP messages:

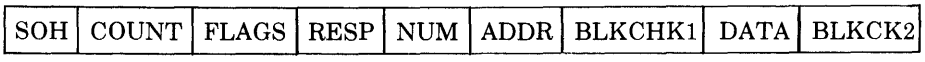
- Data (Section 2.2.1)
- Control (Section 2.2.2)
- Maintenance (Section 2.2.3)

Data messages send user data over a physical link. Control messages return acknowledgments and other control information. Maintenance messages consist of the basic DDCMP envelope but contain information for down-line loading, up-line dumping, link testing, or triggering a remote, adjacent system.

2.2.1 Data Messages

DDCMP formats all messages received from the Transport layer to be sent across the physical link into a data message format (Figure 2–1). The data message format ensures proper handling and error checking of both the header information and the data being sent. In the message format figures in this chapter the numbers below each message field indicate its length in bits.

Data Message Format



SOH = the numbered data message identifier
COUNT = the byte count field
FLAGS = the link flags
RESP = the response number
NUM = the transmit number
ADDR = the station address field
BLKCK1 = the block check on the numbered message header
DATA = the numbered message data field
BLKCK2 = the block check on the data field

Figure 2–1: DDCMP Data Message Format

2.2.2 Control Messages

DDCMP has five control messages, which carry channel control information, transmission status, and initialization notification between DDCMP modules. A brief description of each message follows. Figure 2–2 shows the corresponding formats.

Acknowledge Message (ACK). This message acknowledges the receipt of correctly numbered data messages that have "passed" the CRC–16 check. The ACK message is used when acknowledgments are required, and when no numbered messages are to be sent in the reverse direction. The ACK message conveys the same information as the RESP field in numbered data messages.

Negative Acknowledge Message (NAK). The NAK message passes error information from the DDCMP "data-receiving" module to the DDCMP "data-sending" module. The NAKTYPE field indicates the cause of the error. The NAK message serves two purposes.

- It acknowledges receipt of all previously transmitted messages with a number less than the current message number received (modulo 256).
- It notifies the sender of error conditions related to the current message.

Reply to Message Number (REP). The REP message requests received message status from the data receiver. The data sender sends a REP message under the following conditions:

- The data sender has sent a data message, and
- The data sender has not received an acknowledgment of that data message, and
- A time-out has occurred: the time allocated for an acknowledgment has expired.

Start Message (STRT). The STRT message establishes initial contact and synchronization on a DDCMP link. The DDCMP module sends this message during link start-up or reinitialization.

Start Acknowledge Message (STACK). The STACK message is the response to a STRT message. It tells the receiving DDCMP that the transmitting node has completed initialization.

Acknowledge Message (ACK) Format

ENQ	ACKTYPE	ACKSUB	FLAGS	RESP	FILL	ADDR	BLKCK3
8	8	6	2	8	8	8	16

Negative Acknowledge Message (NAK) Format

ENQ	NAKTYPE	REASON	FLAGS	RESP	FILL	ADDR	BLKCK3

Reply to Message Number (REP) Format

ENQ	REPTYPE	REPSUB	FLAGS	FILL	NUM	ADDR	BLKCK3

Start Message (STRT) Format

ENQ	STRTTYPE	STRTSUB	FLAGS	FILL	FILL	ADDR	BLKCK3

Start Acknowledge Message (STACK) Format

ENQ	STCKTYPE	STCKSUB	FLAGS	FILL	FILL	ADDR	BLKCK3

ENQ = the control message identifier
ACKTYPE = the ACK message type with a value of 1
NAKTYPE = the NAK message type with a value of 2
REPTYPE = the REP message type with a value of 3
STRTTYPE = the STRT message type with a value of 6
STCKTYPE = the STACK message type with a value of 7
ACKSUB = the ACK subtype with a value of 0
REASON = the NAK error reason
REPSUB = the REP subtype with a value of 0
STRTSUB = the STRT subtype with a value of 0
STCKSUB = the STACK subtype with a value of 0
FLAGS = the link flags
RESP = the response number used to acknowledge received messages that checked out to be correct
FILL = a fill byte with a value of 0
ADDR = the tributary address field
BLKCK3 = the control message block check

Figure 2-2: DDCMP Control Message Formats

2.2.3 Maintenance Messages

The Maintenance Message, used in DDCMP maintenance mode, has the format shown in Figure 2–3. This message is a DDCMP envelope for data controlling down-line loading (deposit), up-line dumping (examine), link testing, and restarting (triggering) an unattended computer system. The protocol that DNA models for performing these functions is the Maintenance Operation Protocol (MOP). MOP messages are sent within the DDCMP Maintenance Message.

Maintenance Message Format

DLE	COUNT	FLAGS	FILL	FILL	ADDR	BLKCK1	DATA	BLKCK2
8	14	2	8	8	8	16	8n	16

DLE = the maintenance message identifier
COUNT = the byte count field
FLAGS = the link flags
FILL = a fill byte with a value of 0
ADDR = the tributary address field
BLKCK1 = the header block check on fields DLE through ADDR
DATA = the data field (Section 3.5)
BLKCK2 = the block check on the DATA field

Figure 2–3: DDCMP Maintenance Message Format

2.3 DDCMP Operation

The DDCMP module has three functional components:

- Framing
- Link management
- Message exchange

Framing. The framing component locates the beginning and end of a message received from a transmitting DDCMP module. Framing involves synchronizing data by locating a certain bit, byte or message, and then operating at the same rate as the bit, byte or message.

The modems and interfaces at the Physical Link level synchronize bits.

The DDCMP framing component synchronizes bytes by locating a certain 8-bit window in the bit stream. On asynchronous links, DDCMP uses start-stop transmission techniques to synchronize bytes. On synchronous links, DDCMP searches for a SYN character. Byte synchronization is inherent in 8-bit multiple parallel links.

DDCMP synchronizes messages by searching for one of the three special starting bytes (SOH, ENQ, or DLE) after achieving byte synchronization. To maintain message synchronization, DDCMP:

- Counts out fixed length headers.
- When required, counts out variable length data based on the count field of the header.

Link management. The link management component controls transmission and reception on links connected to two or more transmitters and/or receivers in a given direction. Link Management controls the direction of data flow on half-duplex links and the selection of tributary stations on multipoint links, using link flags. In addition, link management uses selective addressing to control the receipt of data on multipoint links.

Message exchange. The message exchange component transfers the data correctly and in sequence over the link. Message exchange operates at the message level (after framing is accomplished), exchanging data and control messages.

2.3.1 Typical Message Exchange

DDCMP is a *positive-acknowledgment-with-retransmission* protocol. This means that for each data message correctly received and passed to the next level DDCMP returns a positive acknowledgment. Such acknowledgment is either an Acknowledge Message (ACK) or a piggy-backed acknowledgment in the response (RESP) field of a data message.

If DDCMP receives a message out of sequence or with an error detected by the CRC, DDCMP does not pass the data to the user. Typically, DDCMP does not acknowledge this message. Eventually a time-out occurs, and the "data-sending" DDCMP retransmits the message. Alternatively DDCMP may send a NAK to the "data-sending" DDCMP module.

The DDCMP operation for transmission of a data message works as follows:

1. The transmitter increments the message number (modulo 256), putting it (n) in the data message. The message is transmitted within the required framing envelope. A timer is started.
2. The receiver frames and receives the message, checks the received CRC value against a computed CRC value, and compares the message number with that expected. If the message checks out, the receiver returns a positive acknowledgment (ACK) with that number, passes the message to the next higher layer, and increments the next expected number to $n + 1$ (modulo 256). If the message does not check out, the receiver ignores the message (or sends a NAK).
3. The transmitter then follows one of three procedures:
   - If the transmitter receives a positive acknowledgment, it checks the number received to see if it is for an outstanding message. If so, the transmitting DDCMP notifies the user of successful transmission of that

message as well as of any previous, lower-numbered outstanding messages. If all outstanding messages have been acknowledged, the timer is stopped. If one or more messages remain outstanding, the timer is restarted.

- If the transmitter receives nothing, the timer expires. The transmitter sends a REP message to initiate error recovery.
- If the transmitter receives a negative acknowledgment, it retransmits the message.

The transmitter may send several data messages before requiring acknowledgment of the first one. Acknowledgment of the highest numbered message implies acknowledgment of lower-numbered messages. A negative acknowledgment implies a positive acknowledgment of any previously transmitted lower-numbered messages.

Figure 2–4 shows a message exchange involving positive acknowledgment and pipelining. Figure 2–5 shows error recovery from a NAK.

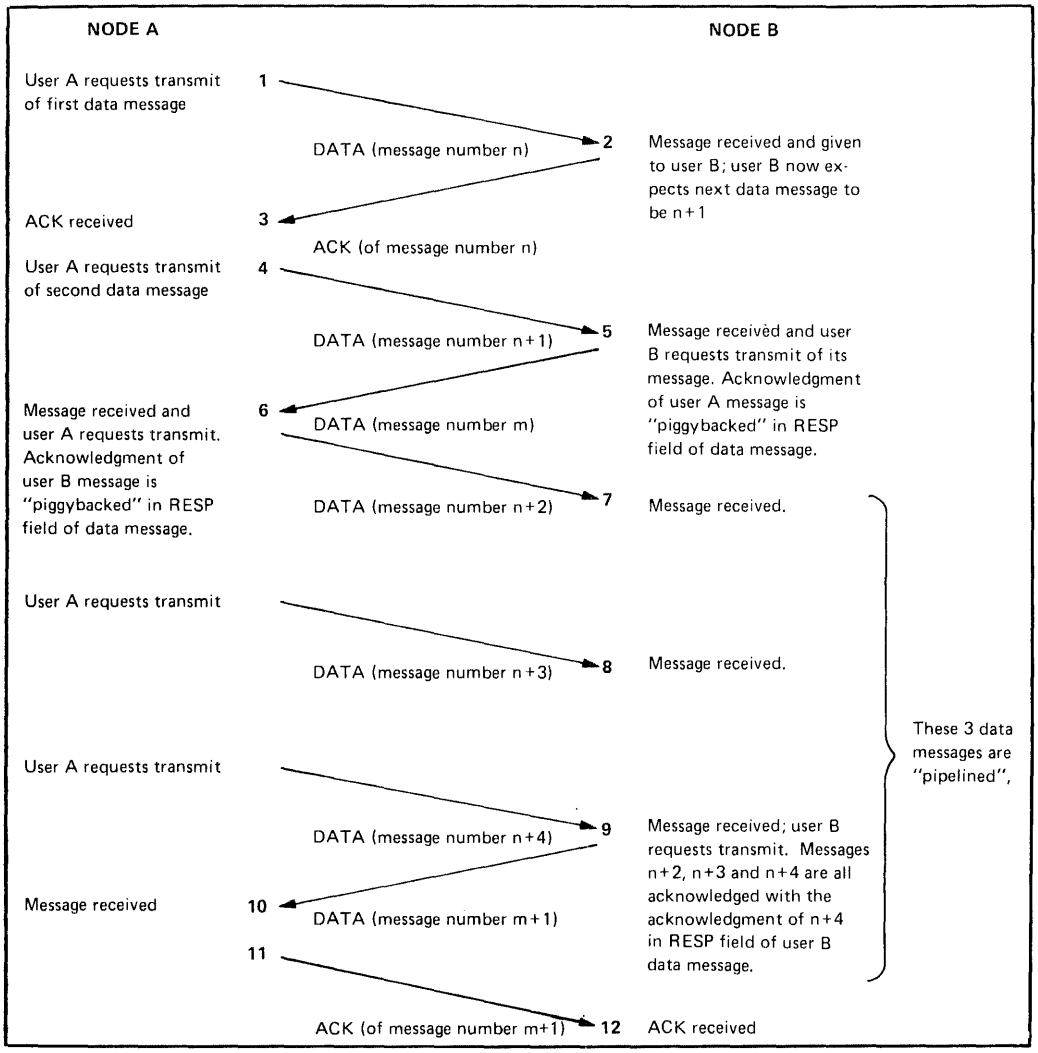


Figure 2–4: Typical DDCMP Message Exchange, Showing Positive Acknowledgment, Piggybacking, and Pipelining

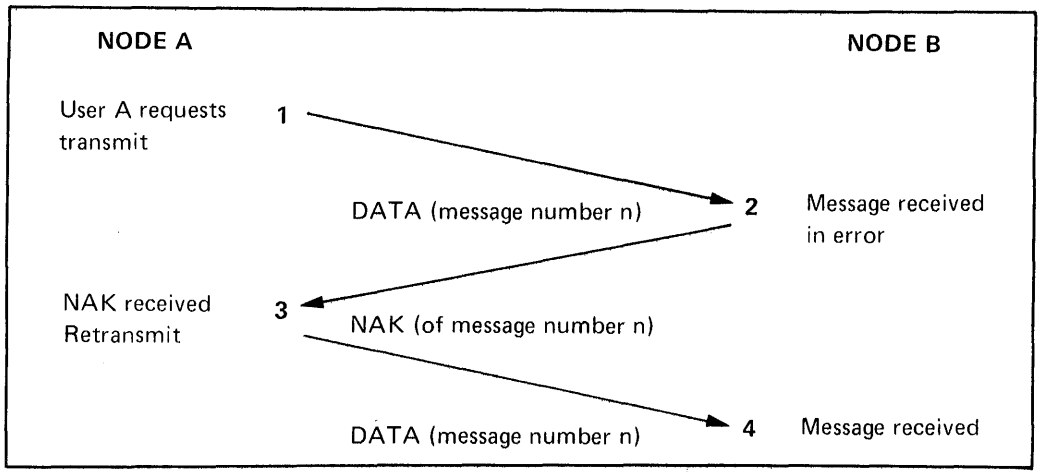


Figure 2-5: DDCMP Message Exchange, Showing Error Recovery

2.3.2 Maintenance Mode

Maintenance mode uses the DDCMP framing and link management components, but not the message exchange component. Sequencing or acknowledgment, if required, must be handled within the data fields of the maintenance messages.

2.4 MOP Functional Description

MOP messages are transmitted within the DDCMP maintenance mode envelope. MOP performs the following functions:

- Down-line loading the memory of a computer system
- Up-line dumping memory contents, usually upon a system failure
- Loopback testing of the data link and/or its hardware components
- Restarting a remote and possibly unattended computer system

2.5 MOP Messages

Table 2-1, following, describes the MOP messages:

Table 2-1: MOP Messages

Message	Description
Memory Load with Transfer Address (Deposit Memory and Transfer)	Causes the contents of the image data to be loaded into memory at the load address, and the system to be started at the transfer address.
Memory Load without Transfer Address (Deposit Memory)	Causes the contents of the image data to be loaded into memory at the load address.
Request Memory Dump (Examine Memory)	Requests a dump of a portion of memory to be returned in a memory dump data message.

(continued on next page)

Table 2-1 (Cont.): MOP Messages

Message	Description
Enter MOP Mode	Causes a system not in the MOP mode to enter MOP mode if the password matches. Usually transfers control of the satellite to a MOP program. Used for unattended satellite systems.
Request Program	Requests a program to be sent in some unspecified number of memory load messages.
Request Memory Load	Requests the next load in a loading sequence and provides error status on the previous load.
MOP Mode Running	Indicates to a host that the system is in the MOP mode and supports the features indicated in the message.
Memory Dump Data	Returns the requested memory image in response to a Request Memory Dump message.
Parameter Load with Transfer Address	Loads system parameters and transfers control to the loaded program.
Loopback Test	Tests a link by echoing the message sent by the host.
Looped Data	Returns the test message data in response to a Loopback Test message. Returned by the passive side if the message is looped from a computer.

2.6 MOP Operation

At the request of a higher level module, MOP sends an appropriate message within the DDCMP envelope. MOP messages and functions handle all message acknowledgment, time-out, and retransmission functions.

The node being serviced (being down-line loaded, up-line dumped, triggered, or tested) is called the *satellite* node. The node providing the services is called the *host* node. (However, in Network Management, the node executing the user command to perform the service is the *executor* and the node being serviced is the *target*.) MOP messages pass alternately between the host and satellite.

It is possible for a node on a multipoint link to be in DDCMP maintenance mode executing MOP protocol while the other nodes are on-line executing DDCMP protocol.

A satellite usually processes MOP messages using a program stored in a local Read Only Memory (ROM) or a local storage device. However, some satellite systems, due to memory or storage constraints, may require a program to bootstrap the satellite. For these systems, a subset of MOP, called MOP primary mode, can bootstrap the satellite to operational mode. Normal, operational mode is called MOP secondary mode.

Some implementations of MOP use hardware facilities to transmit MOP messages. For example, a channel between a host computer and a minicomputer front-end may use hardware to provide message framing and error detection. In this case MOP does not require the DDCMP maintenance mode envelope.

2.6.1 Down-Line Loading

In the MOP secondary mode, either the satellite or the host can initiate a down-line load. The MOP message exchange consists of memory load messages from the host, and request memory load responses from the satellite.

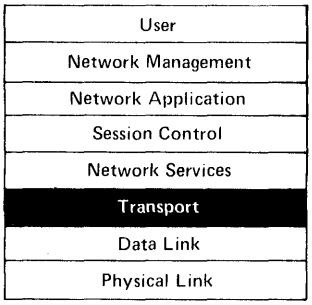
The satellite responds with a request for the next load, number 1, if no errors were detected by the cyclic redundancy check on the host message. The Request Memory Load message is both a request for the next load, and either a positive acknowledgment or a report of a load error. In some implementations, if the host sends a Memory Load message that the satellite detects to be in error, the host does nothing, and waits for a time-out and retransmission of the load.

2.6.2 Up-Line Dumping

The host system initiates up-line dumping. The host sends a Request Memory Dump message to the satellite. The satellite responds with a Memory Dump Data message. If the satellite does not respond, the host repeats the request. If the satellite receives a message that is found to be in error, the satellite either lets the host time out, or returns a MOP Mode Running message. This message causes retransmission of the host request prior to timer expiration.

2.6.3 Link Testing

MOP tests the data link by looping a test message from a point in the physical connection. By moving the loopback point and isolating components, the user can diagnose problems. The active side (the module controlling the test) sends a Loopback Test message out on the link, and waits for its return. The passive side returns a Looped Data message if the message is looped from a computer. If the message is looped from an unintelligent device such as a loopback plug, modem, or hard-wired driver, the passive side returns the Loopback Test message. The Looped Data message prevents infinite loops.



Chapter 3
The Transport Layer

Transport is a message delivery service. Transport accepts messages, called packets within the context of Transport, from the Network Services layer in a source node, and forwards the packets, possibly through intermediate nodes, to a destination node. Transport implements what is termed in the communications field, a Datagram Service. A Datagram Service delivers packets on a *best effort* basis. That is, Transport makes no absolute guarantees against packets being lost, duplicated, or delivered out of order. Rather, higher layers of DNA provide such guarantees (see Chapter 4, The Network Services Layer).

Transport selects routes based on network topology and operator-assigned line costs. Transport automatically adapts to changes in the network topology, for example, by finding an alternate path if a line or node fails. Transport does not adapt to traffic loading: The amount of *traffic* (electrical signals representing data) on a channel does not affect Transport's routing algorithms.

3.1 Transport Functional Description

Transport provides the following functions:

- **Determines packet paths.** A path is the sequence of connected nodes between a source node and a destination node. If more than one path exists for a packet, Transport determines the best path.
- **Forwards packets.** If a packet is addressed to the local node, Transport forwards it up to the Network Services layer. If a packet is addressed to a remote node, Transport forwards it on to the next line in the path.
- **Manages the characteristics of packet paths.** If a line or node fails on a path, Transport finds an alternate path, if one exists.
- **Periodically updates other Transport modules.** Other Transport modules are periodically updated so that all nodes in the network are aware of any routing change (such as line down or node up).

- **Returns packets addressed to unreachable nodes.** Transport returns packets addressed to unreachable nodes to Network Services (NSP), if requested to do so by NSP.
- **Delivers packets between Phase III nodes and Phase II nodes.** A Phase II node can deliver packets to an adjacent node only.
- **Manages buffers.** Transport manages the buffers at nodes that are capable of routing packets from a remote source to a remote destination.
- **Limits the number of nodes a packet can visit.** This prevents old packets from cluttering the network.
- **Limits the amount of time a packet can spend in a node that has halted.**
- **Performs node verification (exchange of passwords), if necessary.**
- **Monitors errors detected by the Data Link layer.**
- **Maintains counters and keeps track of events for network management purposes.**

Transport allows for a functional subset so that in terms of routing there can be two types of Phase III implementations:

1. **Routing.** Sometimes called full routing, this is the full complement of Transport components. A routing node can:
   - Receive packets from any other Phase III node as well as an adjacent Phase II node.
   - Send packets to any other Phase III node as well as an adjacent Phase II node.
   - Route packets from other nodes through to other nodes. This is referred to as *route-through* or *packet switching*.

2. **Nonrouting.** This is a subset of the Transport components. A nonrouting node can:
   - Send packets from itself to any other Phase III node.
   - Receive packets addressed to itself from any other Phase III node.

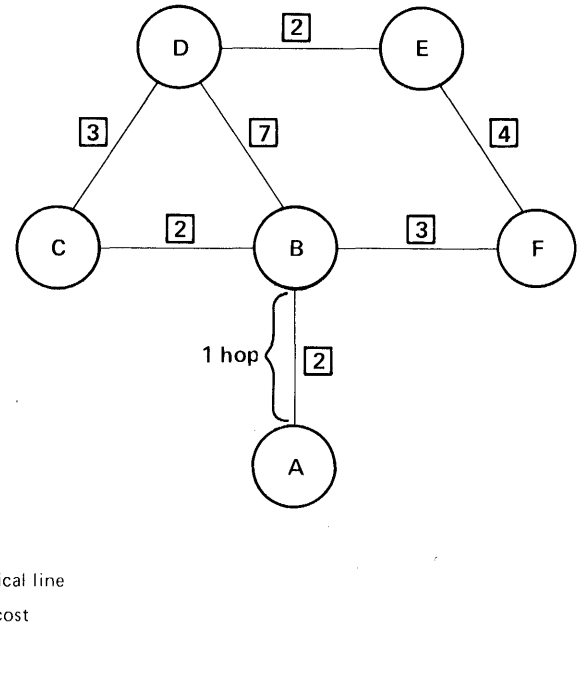
Nonrouting nodes can only be placed as end nodes in a network. An end node does not require the packet switching components. Such a node, therefore, has less routing overhead. An end node may be physically connected to a network by only one link.

The logical distance from one node to another in a network is a *hop*. The complete distance that a message travels from source to destination is the *path length*. Path length is measured in hops. The maximum number of hops

the routing algorithm will forward packets is called *maximum hops*. This number is limited to 30 by the Transport architecture, which supports networks with up to 120 routing nodes.

Transport allows the user to assign a *cost* to each line connected to a node. Cost is an arbitary integer (within the limits of the length of the cost field). Cost is used in the routing algorithm that determines the best path for a packet. Transport routes packets on the path of least cost, even if this is not the path with the fewest hops. The *DNA Transport Functional Specification* does not specify how to assign line costs, but it does suggest a cost assignment algorithm based on line bandwidth. Both cost and maximum hops are values the system manager at each node assigns using Network Management software (Chapter 7).

Figure 3–1 illustrates some of the Transport routing terms. The glossary contains precise definitions of these and other Transport terms (including network diameter, maximum path length, maximum visits, maximum path cost, and maximum cost).



Node A wants to send a packet to Node D. There are three possible paths.

PATH	PATH COST	PATH LENGTH
A to B, B to C, C to D	2 + 2 + 3 = 7*	3 hops
A to B, B to D	2 + 7 = 9	2 hops
A to B, B to F, F to E, E to D	2 + 3 + 5 + 2 = 12	4 hops

*7 is the lowest path cost; Node A therefore routes the packet to Node D via this path.

Figure 3–1: Routing Terms

Transport does not automatically adjust to traffic flow, since packets travel on the least costly paths.

Transport routes packets to numerical node addresses. Because users often prefer to refer to nodes by names, network node names must be mapped to unique network node addresses in a mapping scheme. A node is known to other nodes by one name (optional) and one address, both of which are unique in the network. However, implementations can optionally provide local users with the capability of assigning other node names that map to network-known names and/or addresses. This process occurs at the Session Control level (Chapter 5).

3.2 Transport Messages

There are two types of Transport messages: data messages and control messages. Data messages carry data from the Network Services layer. Transport adds a packet route header to NSP messages. Control messages exchange information between Transport modules in adjacent nodes to initialize Transport, maintain routing data, and monitor the states of lines.

3.2.1 Packet Route Header

Figure 3–2 shows the packet route header format. The numbers under fields in this and the following Transport message formats indicate the lengths of the fields in bits.

Packet Route Header Message Format

RTFLG	DSTNODE	SRCNODE	FORWARD
8	16	16	8

RTFLG = the set of flags used by the routing nodes, including:

- Routing evolution flag (set to either routing node or Phase II node)
- Return to sender flag (indicates whether or not packet is being returned)
- Return to sender request flag (indicates whether to discard or try to return packet).

DSTNODE = the destination node address
SRCNODE = the source node address
FORWARD = the number of nodes this packet can visit

Figure 3–2: Packet Route Header Format

3.2.2 Transport Control Messages

Transport has four control messages:

Routing message. The Routing message provides information necessary for updating the routing data base of an adjacent node. The information is the path cost and path hops to a destination.

Hello and Test message. The Hello and Test message tests the physical link to a node to determine if a node is still on. Transport sends this message periodically to adjacent nodes in the absence of other traffic. Upon receipt of this or any other valid message, Transport starts (or restarts) a timer. If the timer expires before another message is received from that node, Transport considers the line down.

Initialization message. Transport sends this message when initializing. The message contains information about the node type, required verification, maximum Data Link layer receive block size, and Transport version.

Verification message. This message is used for verification purposes if the Initialization message indicates it is required.

Figure 3–3 summarizes the Transport control message formats.

Routing Message Format

CTLFLG	SRCNODE	RTGINFO	CHECKSUM
16	16	16n	16

Hello and Test Message Format

CTLFLG	SRCNODE	TEST DATA
16	16	8–128

Initialization Message Format

CTLFLG	SRCNODE	INITFO
16	16	8n

Figure 3–3: Transport Control Message Formats

Verification Message Format

CTLFLG	SRCNODE	FCNVAL
16	16	8-64

CTLFLG = Transport control flag, with the following types (bits 1–3):

0 = Initialization message
1 = Verification message
2 = Hello and Test message
3 = Routing message

SRCNODE = identification of source node's Transport
RTGINFO = path length and path cost to all destinations
CHECKSUM = one's complement add check on routing information
TEST DATA = sequence of up to 128 bytes of data to test the line
INITFO = node type, required Transport verification message, maximum Data Link layer receive block size, Transport version.
FCNVAL = type-dependent verification information; function value.

Figure 3–3 (Cont.): Transport Control Message Formats

3.3 Transport Operation

Transport consists of the following two sublayers with associated functional components:

Transport Control Sublayer

- Routing (Section 3.3.1)
- Congestion control (Section 3.3.2)
- Packet lifetime control (Section 3.3.3)

Transport Initialization Sublayer

- Initialization (Section 3.3.4)
- Physical line monitor (Section 3.3.4)

3.3.1 Routing

This component contains five processes described below:

- Decision
- Update
- Forwarding
- Select
- Receive

Decision. The decision process selects routes to each destination in the network. It consists of a connectivity algorithm that maintains path lengths, and a traffic assignment algorithm that maintains path costs. When a routing node receives a Routing message, the routing node executes the two decision algorithms. This execution results in updating the data bases used to determine packet routes.

Update. The update process constructs and propagates Routing messages (see Section 4.2.2). The update process sends Routing messages to adjacent nodes as required by the decision process and periodically to ensure the integrity of the routing data bases.

Forwarding. The forwarding process supplies and manages the buffers necessary to support packet route-through to all destinations.

Select. The select process performs a table look-up to select the output line for the packet. If a destination is unreachable, this process either returns the packet to the sender or discards it, depending on the option flagged in the packet route header.

Receive. The receive process inspects a packet's route header, dispatching the packet to an appropriate Transport control component or to Network Services.

3.3.2 Congestion Control

Congestion control consists of a single process, transmit management. This process manages buffers by limiting the maximum number of packets on a queue for a line. If a queue for a particular line reaches a predetermined threshold, additional packets for that queue are discarded to prevent congestion. Transmit management also regulates the ratio of packets received directly from Network Services to route-through packets. This regulation prevents a locally-generated packet from degrading route-through service.

3.3.3 Packet Lifetime Control

Packet lifetime control comprises three processes:

Loop detector. The loop detector prevents excessive packet looping by discarding packets that have visited too many nodes.

Node listener. The node listener receives Hello and Test messages from adjacent Transport modules in order to monitor lines. If messages stop arriving on a line, this process declares the line down.

Node talker. The node talker sends Hello and Test messages to the node listener of adjacent nodes.

3.3.4 Initialization and Physical Line Monitor

Transport initialization is a start-up procedure for adjacent nodes which provides synchronization between adjacent Transport modules. The physical line monitor monitors errors detected by the Data Link layer. These components

are Data Link-dependent. They control the Data Link layer and mask its characteristics from the rest of the Transport components.

Figure 3-4 summarizes the Transport Control sublayer operation.

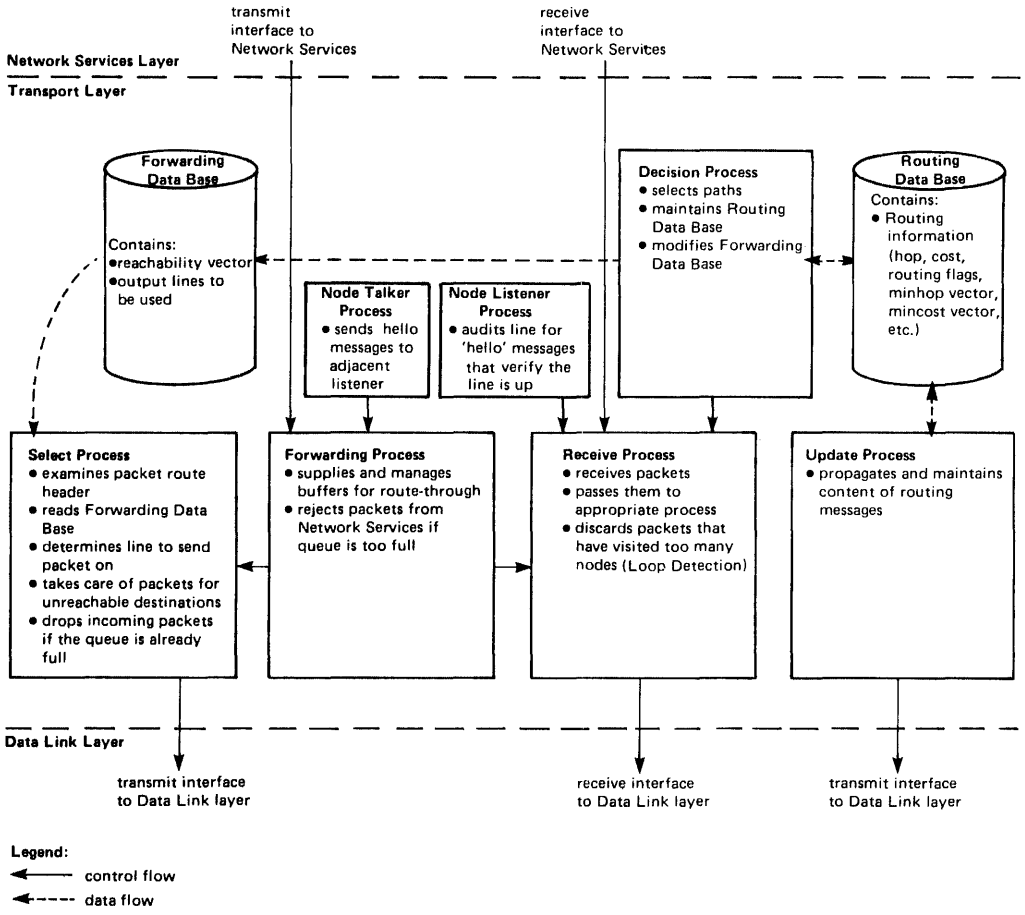
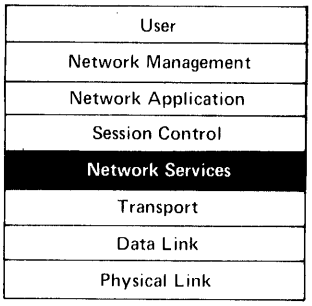


Figure 3-4: Transport Control Components and their Functions



Chapter 4
The Network Services Layer

The Network Services layer, residing immediately above the Transport layer, provides a system-independent process-to-process communication service that allows two processes to exchange data reliably and sequentially, regardless of their locations in a network. Communication is on a connection basis. A connection between two processes is called a logical link. A logical link permits two-way simultaneous transmission of normal data messages and independent two-way simultaneous transmission of interrupt messages.

4.1 NSP Functional Description

NSP performs the following functions:

- Creates and destroys logical links.
- Guarantees the delivery of data and control messages in sequence to a specified destination by means of an error control mechanism.
- Manages the movement of interrupt and normal data from transmit buffers to receive buffers, using flow control mechanisms.
- Breaks up normal data messges into segments that can be transmitted individually, and reassembles these segments in proper sequence upon reception.

4.2 NSP Messages

NSP modules provide logical link service, flow control, error control, and other functions by sending and receiving NSP messages. There are three types of NSP messages:

- Data
- Acknowledgment
- Control

Table 4–1 summarizes the functions performed by each NSP message.

Table 4-1: NSP Messages

Type	Message	Description
Data	Data Segment	Carries a portion of a Session Control message. (This has been passed to Session Control from higher DNA layers and Session Control has added its own control information, if any.)
Data (also called *Other Data*)	Interrupt	Carries urgent data, originating from higher DNA layers.
	Data Request	Carries data flow control information (also called Link Service message).
	Interrupt Request	Carries interrupt flow control information (also called Link Service message).
Acknowledgment	Data Acknowledgment	Acknowledges receipt of either a Connect Confirm message or one or more Data Segment messages.
	Other Data Acknowledgment	Acknowledges receipt of one or more Interrupt, Data Request or Interrupt Request messages.
	Connect Acknowledgment	Acknowledges receipt of a Connect Initiate message.
Control	Connect Initiate	Carries a logical link connect request from a Session Control module.
	Connect Confirm	Carries a logical link connect acceptance from a Session Control module.
	Disconnect Initiate	Carries a logical link connect rejection or disconnect request from a Session Control module.
	No Resources	Sent when a Connect Initiate message is received and there are no resources to establish a new logical link (also called Disconnect Confirm message).
	Disconnect Complete	Acknowledges the receipt of a Disconnect Initiate message (also called Disconnect Confirm message).
	No Link	Sent when a message is received for a nonexisting logical link (also called Disconnect Confirm message).
	No Operation	Does nothing (included for compatibility with NSP V3.1).

4.3 NSP Operation

NSP operation consists of four functions:

- Creation, maintenance, and destruction of logical links (Section 4.3.1)
- Segmentation and reassembly of data (Section 4.3.2)
- Error control (Section 4.3.3)
- Flow control (Section 4.3.4)

4.3.1 Logical Links

The primary functions of NSP are to create, operate, and destroy logical links at the request of the Session Control layer. A logical link may be thought of as a full-duplex logical channel between two users. The users are guaranteed that, in the absence of a network failure disconnecting them, data sent on a logical link by one user will be received by the other user in the order sent. An equivalent term for logical link that is often used is virtual circuit. A logical link enables a user process at one end of the link to send data to a user process at the other end of the link. The NSP mechanisms that set up a link, check data for errors, and manage data flow are transparent to the user processes.

There can be several logical links at any time, even between the same two NSP implementations. Any Phase III node can establish a logical link with any other Phase III node in the same network. A Phase II node can establish logical links only with adjacent nodes. Figure 4–1 illustrates typical connections.

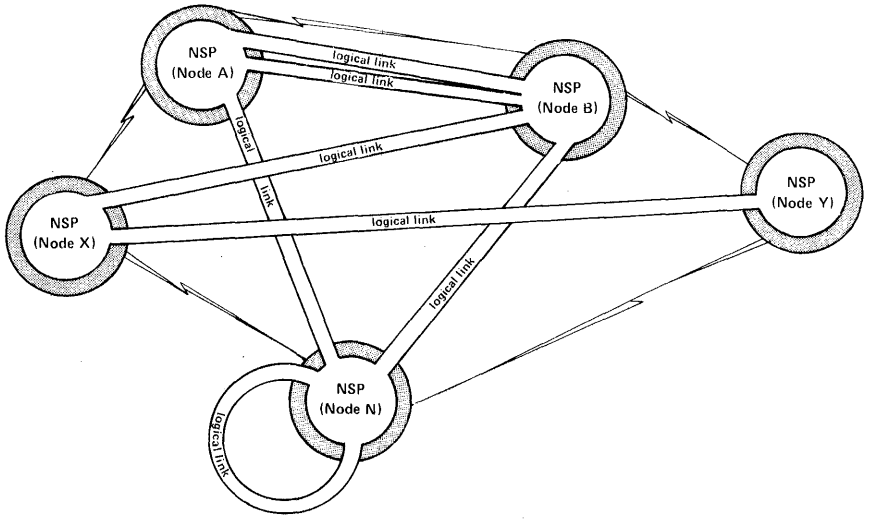


Figure 4–1: Logical Links

NSP establishes, maintains, and destroys logical links by exchanging control messages with other NSP modules or with itself. Figure 4–2 shows a typical message exchange. In this figure an NSP module first initiates a connection, then sends data, and finally, disconnects the link, based on commands from Session Control. The messages that control data flow are not shown.

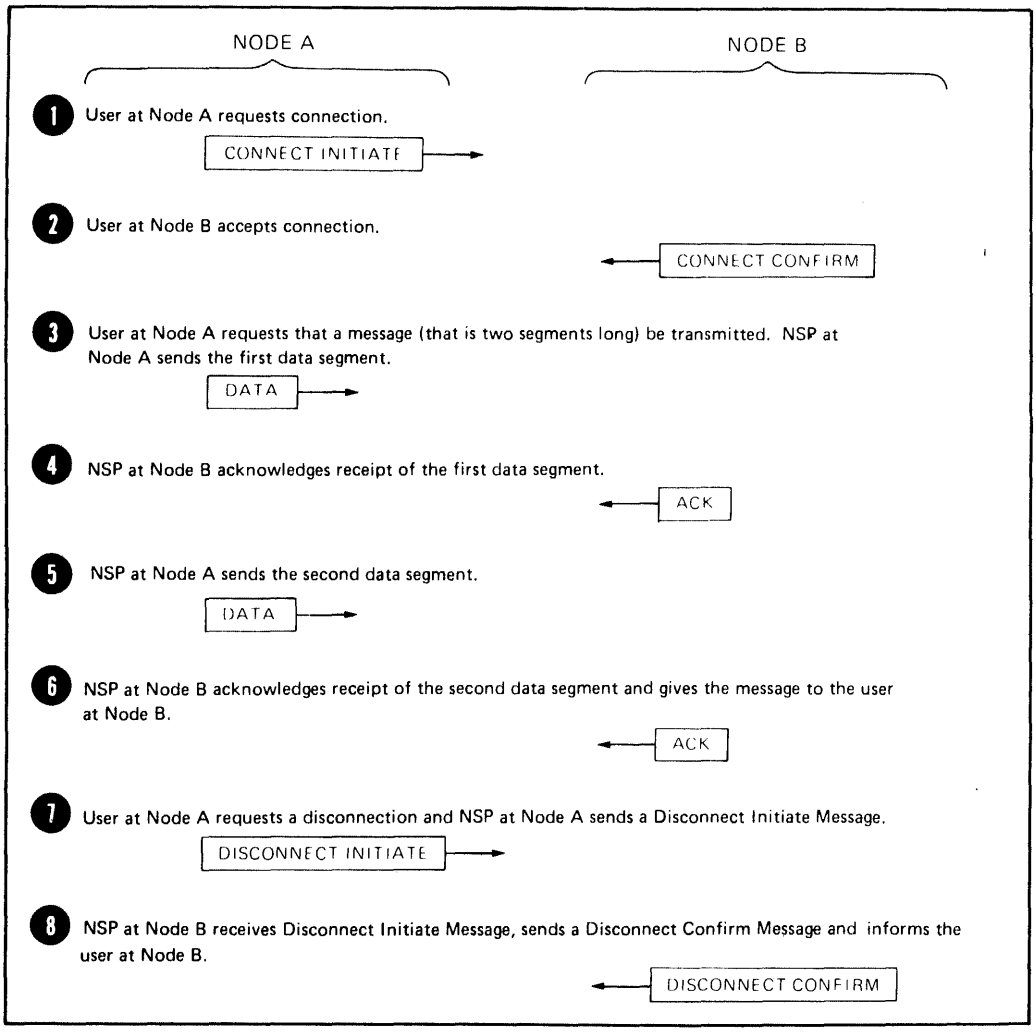


Figure 4–2: Typical Message Exchange Between Two Implementations of NSP

NSP operates concurrently in both directions on a logical link (full-duplex). The user process at either end of the link can initiate a disconnection at any time.

You can think of a logical link as made up of two separate data *subchannels*, each carrying messages in both directions:

Normal data subchannel. This subchannel carries Data Segment messages between two NSP modules.

***Other Data* subchannel.** This subchannel carries:

- Interrupt messages
- Data Request messages
- Interrupt Request messages

4.3.2 Segmentation and Reassembly of Data

Transport limits the amount of user data that NSP can send in one datagram. Taking normal data from Session Control buffers, NSP breaks it up, if necessary, into segments. Each segment is numbered and sent along with its number and other control information in a Data Segment message to the receiving NSP module. The receiving NSP module uses the sequence numbers to reassemble the data segments in correct sequence in receiving Session Control buffers.

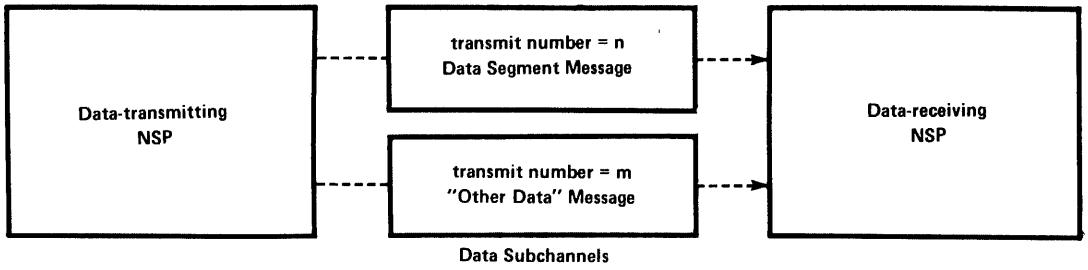
NSP segments normal data only, not interrupts: data travelling in the *Other Data* subchannel is not segmented.

4.3.3 Error Control

The NSPs at each end of a logical link positively acknowledge received data. Optionally, an NSP may negatively acknowledge received normal data that it must discard (for example, if the data is received too far out of order). If the transmitting NSP receives a negative acknowledgment or fails to receive a positive acknowledgment during a timeout interval, it retransmits the data.

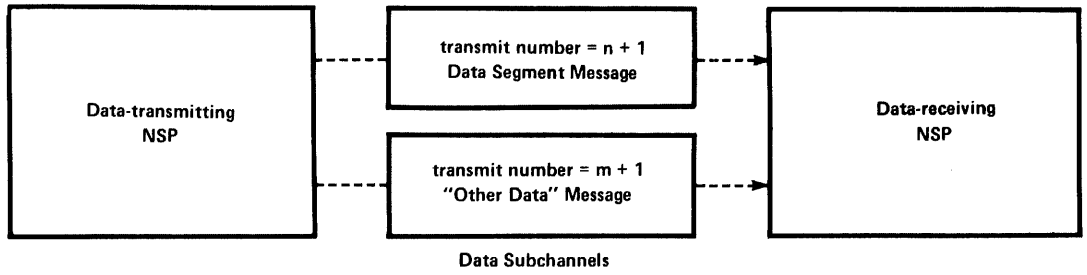
Figure 4–3 describes data segmentation and acknowledgment.

1. The data-transmitting NSP assigns a transmit number to a message, transmits the message, and starts a timer.



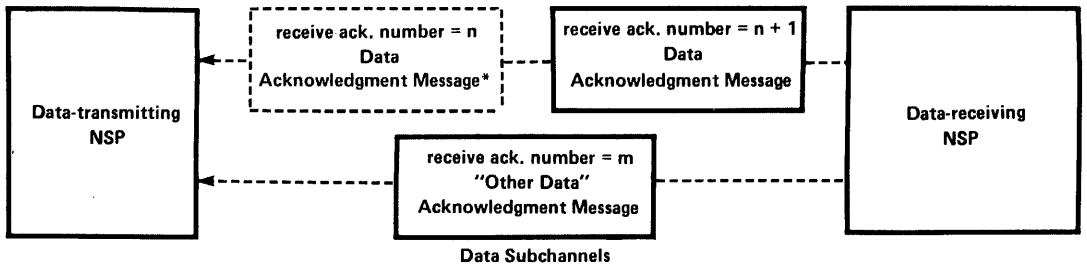
2. If the timer times out, the message is retransmitted.

3. If the timer does not time out, and the flow control mechanism allows another message to be sent, the data-transmitting NSP assigns the transmit number plus one to the next data message transmitted in that subchannel.



4. When the message with the first transmit number is received by the data-receiving NSP, it returns that number as an acknowledgment number within the first acknowledgment.

5. If the next data message transmit number received is equal to the current acknowledgment number plus one, the data-receiving NSP accepts the data message, incrementing the acknowledgment number. It then sends the new receive acknowledgment number back to the data-transmitting NSP within an acknowledgment message.



*The data-receiving NSP might not send an acknowledgment for each data message received. The receive acknowledgment number implies that all previous numbers were received.

6. However, if the data-receiving NSP receives a data message transmit number less than or equal to the current receive acknowledgment number for that subchannel, the data segment is discarded. The data-receiving NSP sends an acknowledgment back to the data-transmitting NSP. The acknowledgment contains the receive acknowledgment number.

7. If the data-receiving NSP receives a data message transmit number greater than the current receive acknowledgment number plus one for that subchannel, the data segment may be held until the preceding segments are received or it may be discarded.

Figure 4-3: Acknowledgment Operation

4.3.4 Flow Control

NSP's flow control mechanisms ensure that data is not lost for lack of buffering capability and that deadlocks do not occur. Both normal and interrupt data are flow-controlled.

The data-receiving part of NSP controls data flow. When a logical link is formed, each NSP informs the other of the way it wants to control the flow of normal data as a data receiver. The receiving NSP chooses one of three types of normal data flow control:

- **None.**
- **Segment.** The receiver sends a request count of the number of segments it can accept. See Figure 4–4.
- **Message.** The receiver sends a request count of the number of Session Control messages it can accept.

These schemes define the use of a request count that is set by the receiver and used by the transmitter to determine when data may be sent. In addition, the receiver can always tell the transmitter to either stop sending data unconditionally or to start sending data under the normal request count conditions. The receiver also controls interrupt data flow with an interrupt request count.

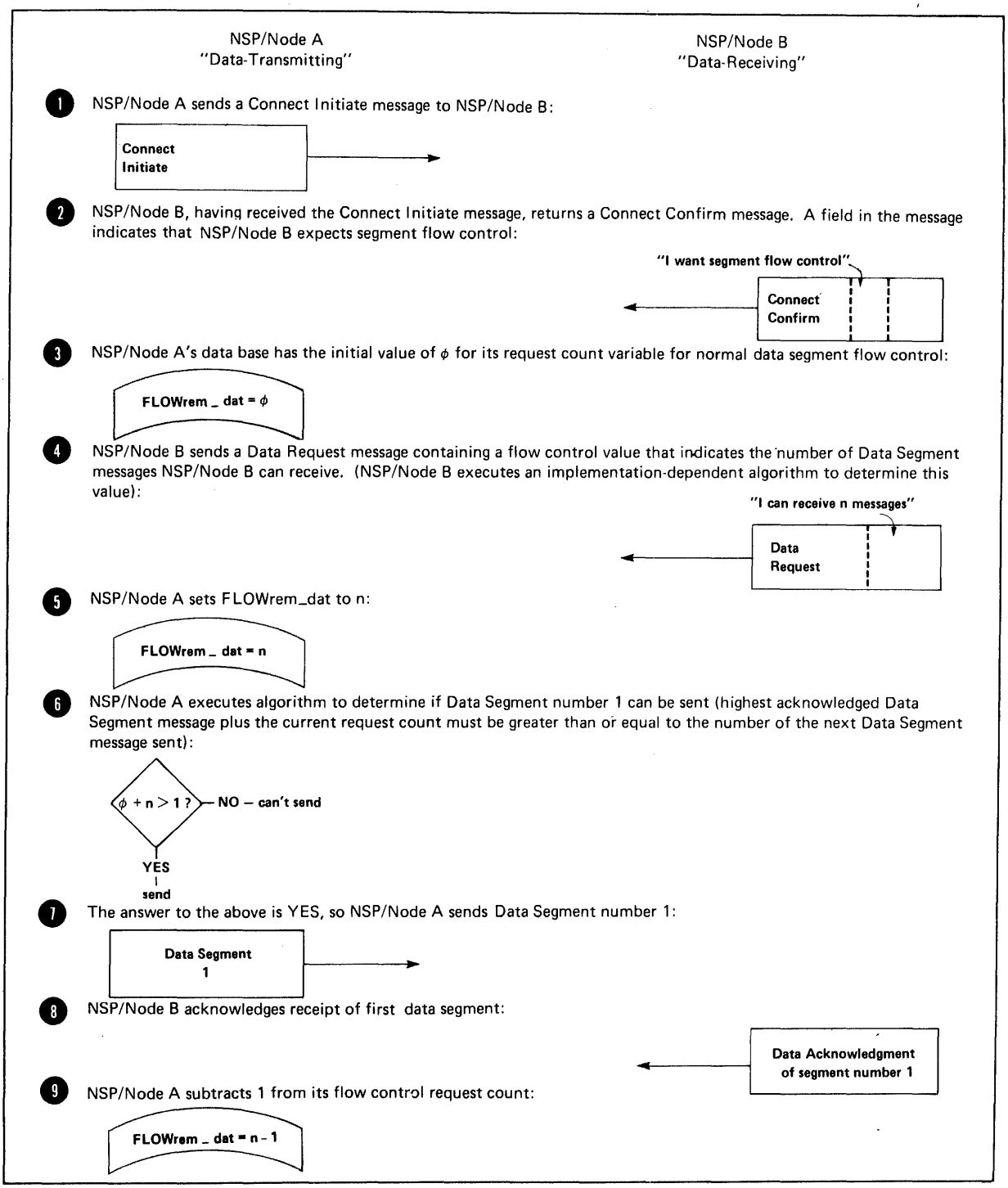
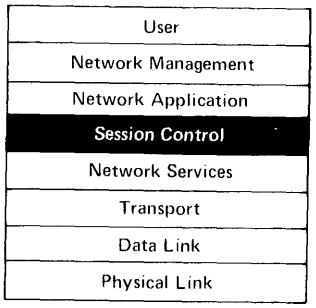


Figure 4–4: Segment Flow Control Shown in One Direction on a Logical Link



Chapter 5
The Session Control Layer

The Session Control layer, residing immediately above the Network Services layer, provides system-dependent, process-to-process communications functions. These functions bridge the gap between the Network Services layer and the logical link functions required by processes running under an operating system. Thus Session Control is the point at which DECnet is integrated with an operating system.

5.1 Session Control Functional Description

Session Control functions include:

- **Mapping node names to node addresses.** A Session Control module maintains a node name mapping table that defines the correspondence between a node name and either a node address or a channel number. (A channel number is used for loopback testing under the control of Network Management.) The table enables the Session Control module to select the destination node address or channel number for outgoing connect requests to NSP. For incoming connect requests from NSP, the Session Control module uses the table to identify the node from which the request originated.
- **Identifying end users.** A Session Control module executes a system-dependent algorithm to determine if an existing end user corresponds to the destination end user specified in an incoming connect request. It also performs additional functions related to passing a connect request to an existing end user.
- **Activating or creating processes.** A Session Control module may create a new process or activate an existing process to handle an incoming connect request.
- **Validating incoming connect requests.** A Session Control module uses access control information included in an incoming connect request to perform system-dependent validation functions.

Session Control defines two data bases that are not implementation-specific:

- A node-name mapping data base.
- A data base containing the states of Session Control and optional default connection timers.

Figure 5–1 shows a model of Session Control operating within a network node.

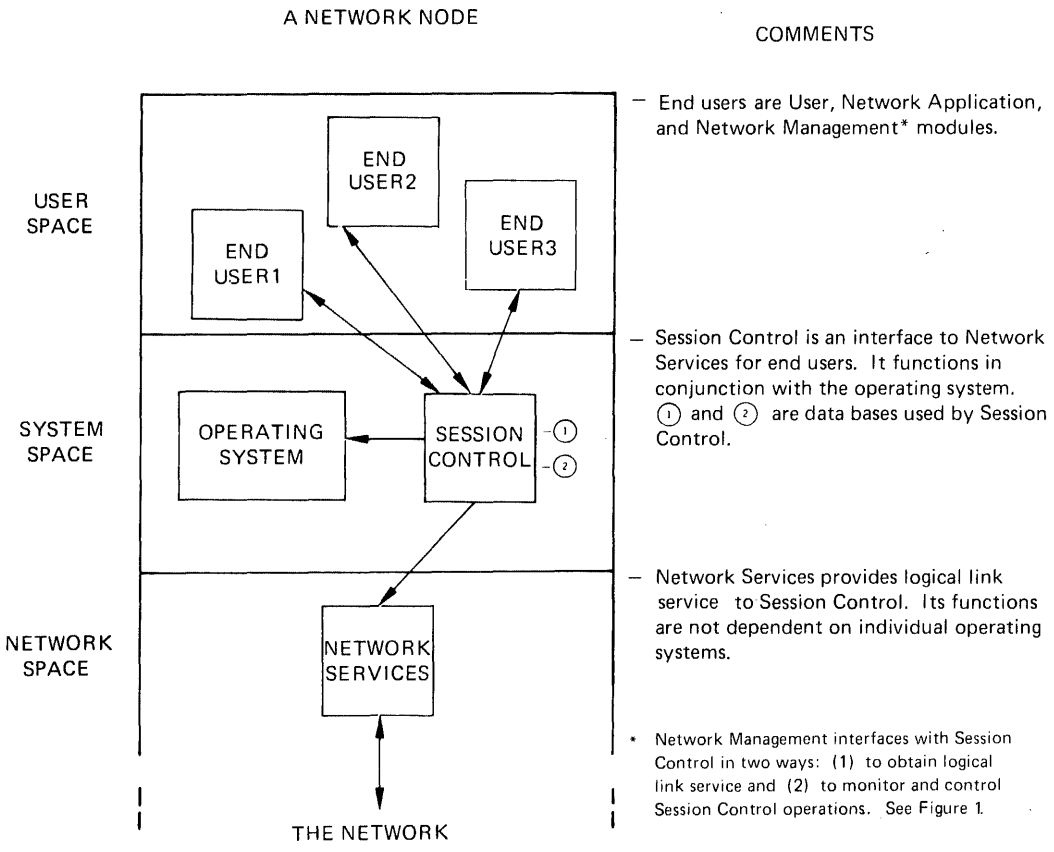


Figure 5–1: A Session Control Model

5.2 Session Control Messages

Session Control's message protocol defines messages sent on a logical link as connect data, reject data, and disconnect data.

Figure 5–2 shows the formats for the Session Control messages. The numbers below each message field indicate its maximum length in bytes.

Connect Data Message Format

DSTNAME	SRCNAME	MENUVER	RQSTRID	PASSWRD	ACCOUNT	USRDTA
19	19	1	39	39	39	16

Reject/Disconnect Data Message Format

REASON	DATA-CTL
2	n

DSTNAME = the destination end user name
SRCNAME = the source end user name
MENUVER = the field format and version format
RQSTRID = the source user identification for access verification
PASSWRD = the access verification password
ACCOUNT = the link or service account data
USRDATA = the end user connect data
REASON = a reason code
DATA-CTL = user data (length of field determined by the total length of reject or disconnect data received from NSP)

Figure 5-2: Session Control Message Formats

5.3 Session Control Operation

Session Control performs the following basic operations:

- Requests logical links on behalf of end users (Section 5.3.1).
- Receives connect requests addressed to end users (Section 5.3.2).
- Sends and receives logical link data (Section 5.3.3).
- Disconnects and aborts logical links (Section 5.3.4).
- Optionally, monitors logical links (Section 5.3.5).

5.3.1 Requesting a Connection

Upon receipt of a logical link connect request from an end user, Session Control performs four tasks:

- Identifies the destination node address or channel number for Network Services by using the node name mapping table.
- Formats connect data for Network Services.

- Issues a connect request to Network Services.
- Optionally starts an outgoing connection timer. Expiration of the timer prior to an acceptance or rejection from Network Services causes Session Control to disconnect the logical link for the source end user.

5.3.2 Receiving a Connect Request

Upon detection of an incoming connect request from Network Services, Session Control performs six tasks:

- Parses connect data to obtain such information as source and destination end user names and access control information.
- Validates any access control information.
- Identifies, creates, or activates a destination end user.
- Maps the source node's address or channel number to a node name, if there is one.
- Delivers the incoming connect request to the end user.
- Optionally starts an incoming timer when the connect request is delivered. Expiration of the timer before the end user accepts the connect request causes Session Control to issue a reject to Network Services.

5.3.3 Sending and Receiving Data

This is a system-dependent function that handles end user requests to send and receive data. Basically, the functions are passed directly to the Network Services layer.

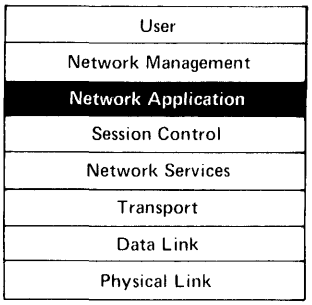
5.3.4 Disconnecting and Aborting a Logical Link

As in the case of sending and receiving data, Session Control passes end user disconnect and abort requests directly to Network Services. Similarly, notification of a logical link disconnect or abort is passed directly to the end user.

5.3.5 Monitoring a Logical Link

This is an optional system-dependent function that may be used for the following purposes:

- Detecting probable network disconnection between the nodes at either end of the logical link
- Detecting a failure, by Network Services, to deliver transmitted data in a timely manner



Chapter 6
The Network Application Layer

The Network Application layer is designed to contain a number of separate, commonly-used modules that access data and provide other often-used services to users. Currently, two Phase III DIGITAL-supplied DNA protocols are specified for this layer:

- **The Data Access Protocol (DAP).** This protocol permits remote file access and file transfer in a manner that is independent of the I/O structure of the operating system being accessed. Sections 6.1 - 6.4 describe DAP.
- **The Loopback Mirror Protocol.** This protocol consists of one message used between the Network Management loopback access routines and loopback mirror. These modules test logical links. Section 7.2.3 describes this operation.

6.1 DAP Functional Description

DAP provides the following functions and features:

- Supports heterogeneous file systems.
- Retrieves a file from an input device (a disk file, a card reader, a terminal, *etc.*).
- Stores a file on an output device (a magtape, a line printer, terminal, *etc.*).
- Transports ASCII files between nodes.
- Supports deletion and renaming of remote files.
- Lists directories of remote files.
- Recovers from errors and reports fatal errors to its user.
- Allows multiple data streams to be sent over a logical link.

- Submits and executes command files.
- Permits sequential, random and indexed (ISAM) access of records.
- Supports sequential, relative, and indexed file organizations.
- Supports wildcard file specification for sequential file retrieval, file deletion, file renaming, and command file execution.
- Permits an optional file checksum to ensure file integrity.

DAP is designed to minimize protocol overhead. For example, defaults are specified for fields wherever possible. In addition, a file transfer mode eliminates the need for DAP control messages once file data flow begins. Finally, relatively small file records can be blocked together and sent as one large message.

DAP is a set of messages and the rules governing their exchange between two cooperating processes. Section 6.2 describes the messages (Table 6–1). Section 6.3 summarizes the operation of the cooperating DAP-speaking processes, which provide DECnet remote file access. Section 6.4 describes the most common DAP-speaking DECnet facilities.

6.2 DAP Messages

DAP-speaking processes use the messages listed in Table 6–1 to accomplish remote file access and transfer.

Table 6–1: DAP Messages

Message	Function
Configuration	Exchanges system capability and configuration information between DAP-speaking processes. Sent immediately after a link is established, this message contains information about the operating system, the file system, protocol version, and buffering ability.
Attributes	Provides information on how data is structured in the file being accessed. The message contains information on file organization, data type, format, record attributes, record length, size, and device characteristics.
Access	Specifies the file name and type of access requested.
Control	Sends control information to a file system and establishes data streams.

(continued on next page)

Table 6–1: (Cont.): DAP Messages

Message	Function
Continue-Transfer	Allows recovery from errors. Used for retry, skip, and abort after an error is reported.
Acknowledge	Acknowledges access commands and control connect messages used to establish data streams.
Access Complete	Denotes termination of access.
Data	Transfers the file data over the link.
Status	Returns the status and information on error conditions.
Key Definition Attributes Extension	Specifies key definitions for indexed files.
Allocation Attributes Extension	Specifies the character of the allocation when creating or explicitly extending a file.
Summary Attributes Extension	Returns summary information about a file.
Date Time Attributes Extension	Specifies time-related information about a file.
Protection Attributes Extension	Specifies the file protection code.
Name	Sends name information when renaming a file or obtaining a directory listing.

6.3 DAP Operation

Two cooperating processes exchange DAP messages: the user process and the server process that acts on the user's behalf at the remote node. User I/O commands accessing a remote file are mapped into equivalent DAP messages and transmitted via a logical link to the server at the remote node. The server interprets the DAP commands and actually performs the I/O for the user. The server returns status and data to the user.

In a typical DAP dialogue, the first message exchange is of Configuration messages that provide information about the operating and file systems, buffer size, and so on. Attributes messages then supply information about the file. The Access Request message typically follows to open a particular file. If data is to be transferred, a data stream is then set up. For both sequential and random access file transfer, one control message sets up the data stream. After the completion of file transfer, Access Complete messages terminate the data stream.

Figure 6–1 shows a DAP message exchange for sequential file retrieval.

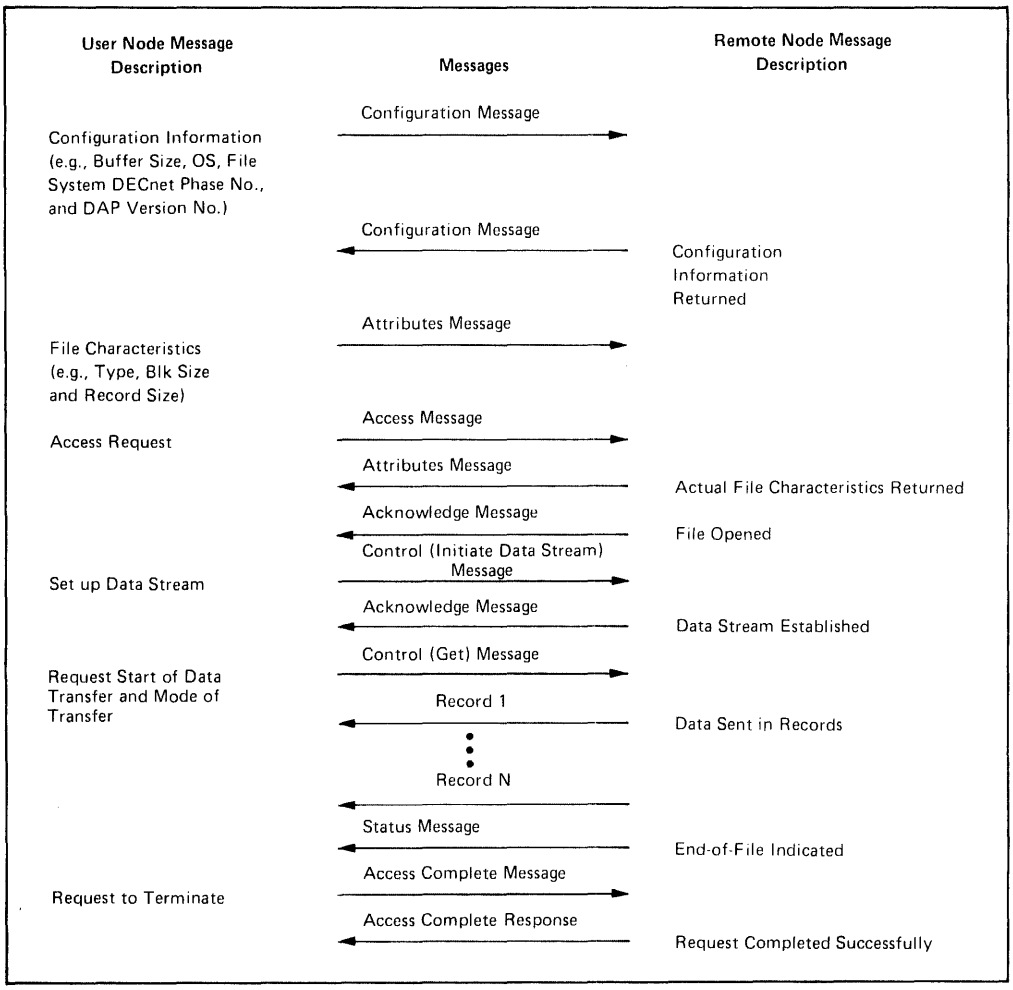


Figure 6–1: DAP Message Exchange (Sequential File Retrieval)

6.4 DECnet Remote File Access Facilities

DECnet implementations currently access remote files using the following facilities:

- **File Access Listener (FAL).** FAL receives user I/O requests at the remote node and acts in the user's behalf. This is the remote DAP-speaking server process.
- **Network File Transfer (NFT).** This interactive utility operates at the user level. It interfaces to DAP-speaking accessing process to provide DAP functions (Figure 6–2). NFT provides network-wide file transfer and manipulation using DAP and the server, FAL (Figure 6–2).
- **Record Management Services (RMS).** RMS is becoming the standard file system for many of DIGITAL's operating systems. In particular, VMS DECnet uses RMS. RMS can generate and transmit DAP messages over logical links. If an RMS file access request includes a node name in the file

specification, RMS maps the access request into equivalent DAP messages. These DAP messages are sent to a remote FAL to complete the request. To the user, remote file access is handled the same way as local file access, except that a remote node name and possibly access control information is necessary for remote file access.

- **Network File Access Routines (NFARs).** NFARs are a set of FORTRAN-callable subroutines (Figure 6–2). NFARs become part of the user process; they cooperate with FAL, using DAP, to access remote files for user applications. RSX DECnet uses NFARs to provide DAP functions.
- **VAX/VMS terminal commands.** VMS commands pertaining to file access and manipulation interface with RMS to provide network-wide file access.
- **Network Management modules.** Network Management modules use DAP services to obtain remote files for down-line loading other remote nodes and to transfer up-line dumps for storage.

Figure 6–2 shows a node-to-node file transfer.

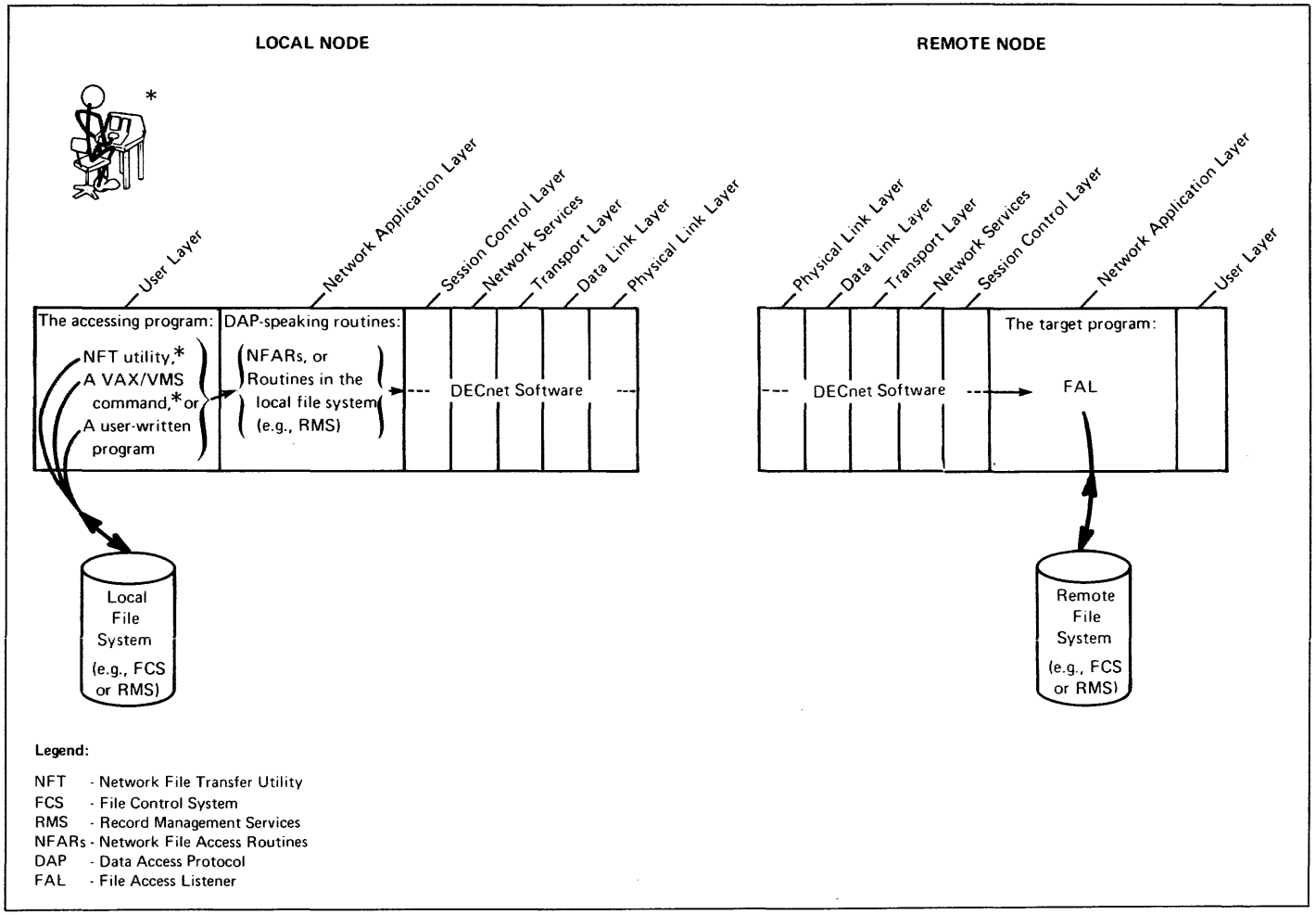
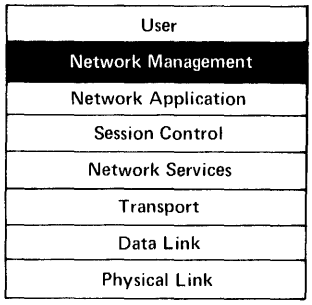


Figure 6–2: File Transfer Across a Network



Chapter 7
Network Management

Network Management allows system managers to control and monitor network operation. Network Management also provides information for use in planning the evolution of a network and correcting network problems.

The Network Management design has three outstanding characteristics:

- Both programs and terminals can access and control a DECnet network via a set of functionally discrete calls and commands.
- Control over a DECnet network can be either distributed or central. Distribution of control can be either partial or complete.
- Network Management is a set of primitive functions or "tools." The system or network manager can fashion them into a management system that meets his specific needs. This allows the managers of a network to construct their own network management philosophy.

Most of the Network Management modules reside in the Network Management layer. In addition to these, however, there are Network Management modules in the user and Network Application layers. Also, one Network Management module, the Event Logger, has a queue residing in each lower layer. This chapter discusses all Network Management modules, regardless of where they reside.

Since Network Management is modular, a DECnet system is not required to implement the architecture in its entirety.

7.1 Network Management Functional Description

Network Management performs the following functions:

- **Loading and dumping remote systems.** For example, a system manager can down-line load an operating system into an unattended, remote system.
- **Changing and examining network parameters.** For example, an operator can change line costs or node names.
- **Examining network counters and events that indicate how the network is performing.** For example, the Event Logger automatically records significant network events.

- **Testing links at both the data link and logical link levels.** For example, a system manager can send a test message that loops back to its origin from a specific point in the hardware connection.
- **Setting and displaying the states of lines and nodes.** For example, an operator can reconfigure a network by turning nodes and lines on or off.

7.2 Network Management Operation

This section summarizes the functions of each Network Management component and describes the operation of the major Network Management functions.

7.2.1 Components

The Network Management components are as follows:

User Layer

- **Network Control Program (NCP).** NCP is a utility at the user level that interfaces with lower level modules. NCP has a standard set of commands that each DECnet implementation uses.

Network Management Layer

- **Network Management Access Routines.** These routines provide generic Network Management functions. The routines communicate across logical links with the Network Management Listener using the Network Information and Control Exchange (NICE) protocol.
- **Network Management Listener.** The Network Management Listener receives Network Management commands from the Network Management level of remote nodes via the NICE protocol. In some implementations it also receives commands from the Network Management Access Routines via the NICE protocol. The Network Management Listener passes function requests to the Local Network Management Functions.
- **Local Network Management Functions.** This component takes function requests from the Access Routines, translating the requests into system-dependent calls.
- **Line Watcher.** Sensing service requests on a line from an adjacent node, the Line Watcher handles automatic remote load, dump, and trigger functions.
- **Line Service Functions.** The Line Watcher and the Local Network Management Functions interface to the Line Service Functions for services that require a direct data link (bypassing the Session Control, Network Services, and Transport layers).
- **Event Logger.** The Event Logger records significant events occurring in the lower layers. DNA specifies event types in the Network Management interface to each layer. An event processor within the Event Logger takes *raw*

events queued in each layer, and records events of the types specified by the system and system manager. Using the Event Logger Protocol, an event transmitter can inform event receivers at other nodes of event occurrences. Events travel to a specified *sink node* for console, file or monitor output.

Figure 7–1 shows the interrelationship of Network Management components in the user and Network Management layers.

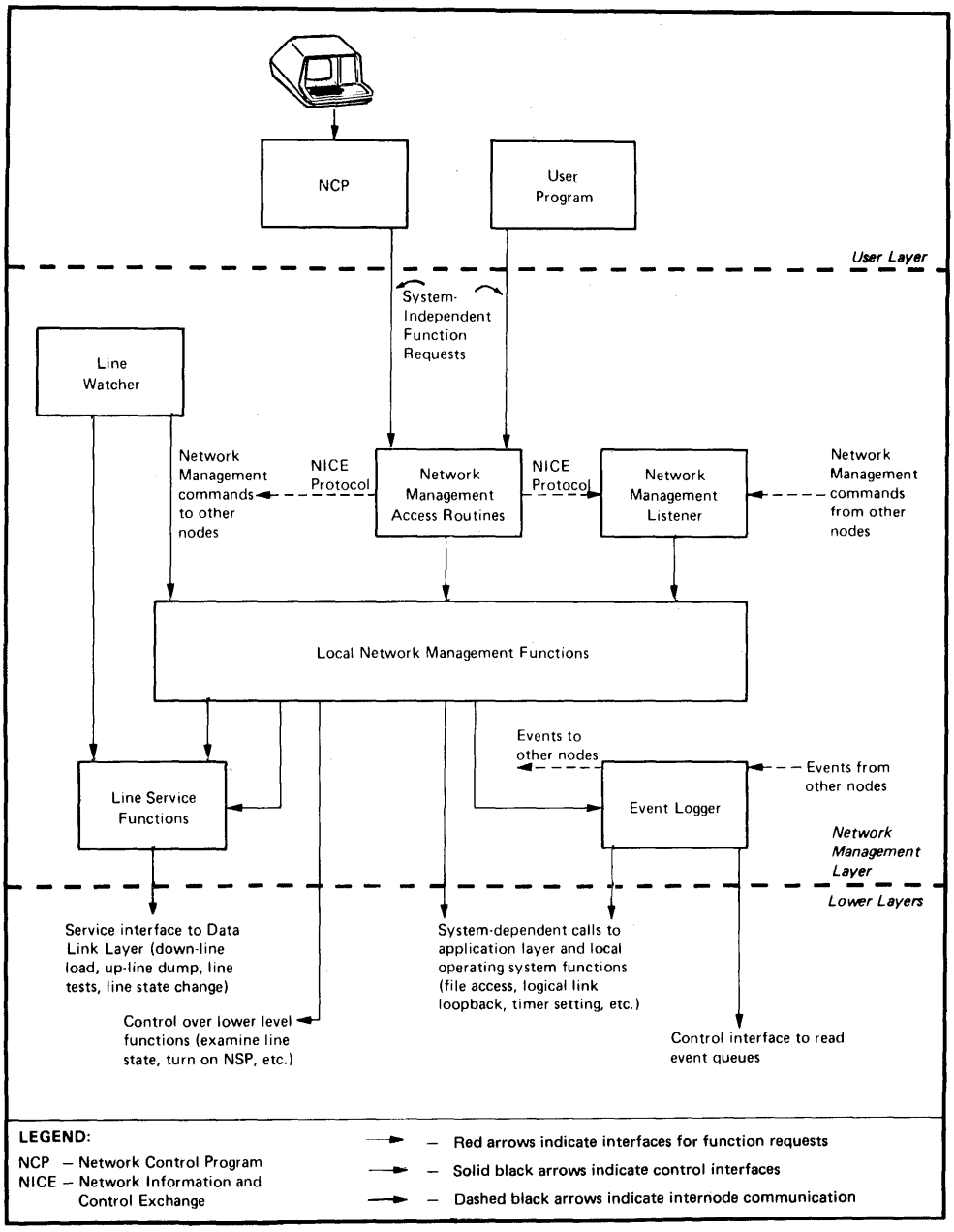


Figure 7–1: Interrelationship of Network Management Components at a Single Node

- **Loopback Access Routines and Loopback Mirror.** For logical link loopback tests, the Local Network Management Functions can interface to the loopback access routines, which use the Loopback Mirror Protocol to loop test messages from a remote Loopback Mirror.

7.2.2 Remote Loading, Dumping, Triggering, and Line Loopback Testing Functions

Network Management typically uses the Maintenance Operation Protocol (MOP) to perform remote loading, dumping, triggering and testing. (Refer to Sections 2.4–2.6). The target (the node being loaded, dumped, *etc.*) or the executor (the adjacent node executing the requests) or a remote *command* node can initiate the function. Figure 7–2 illustrates the down-line load request operation.

Line loopback testing is a procedure for isolating faults in a physical connection between two nodes. Line loopback tests consist of sending out test messages that are looped back at some point. By changing the loopback point, an operator can precisely locate problems.

There are three methods of performing line loopback tests:

1. Using NCP commands, the operator can set certain line devices to loopback mode, and then perform the loopback tests. This "automatic" method can be useful in testing unattended remote nodes.
2. To loop a message from another hardware point, such as a modem, the operator must set up a loopback device manually. This can be done by throwing a loopback switch on a modem or connecting a loopback plug in place of a modem. The operator can then execute the loopback test using NCP commands.
3. By not setting a hardware loopback device and using line loopback NCP commands, an operator can cause loopback test messages to loop back from the Line Service Functions in the Network Management layer of the adjacent node.

Figure 7–3 illustrates line loopback tests and associated NCP commands.

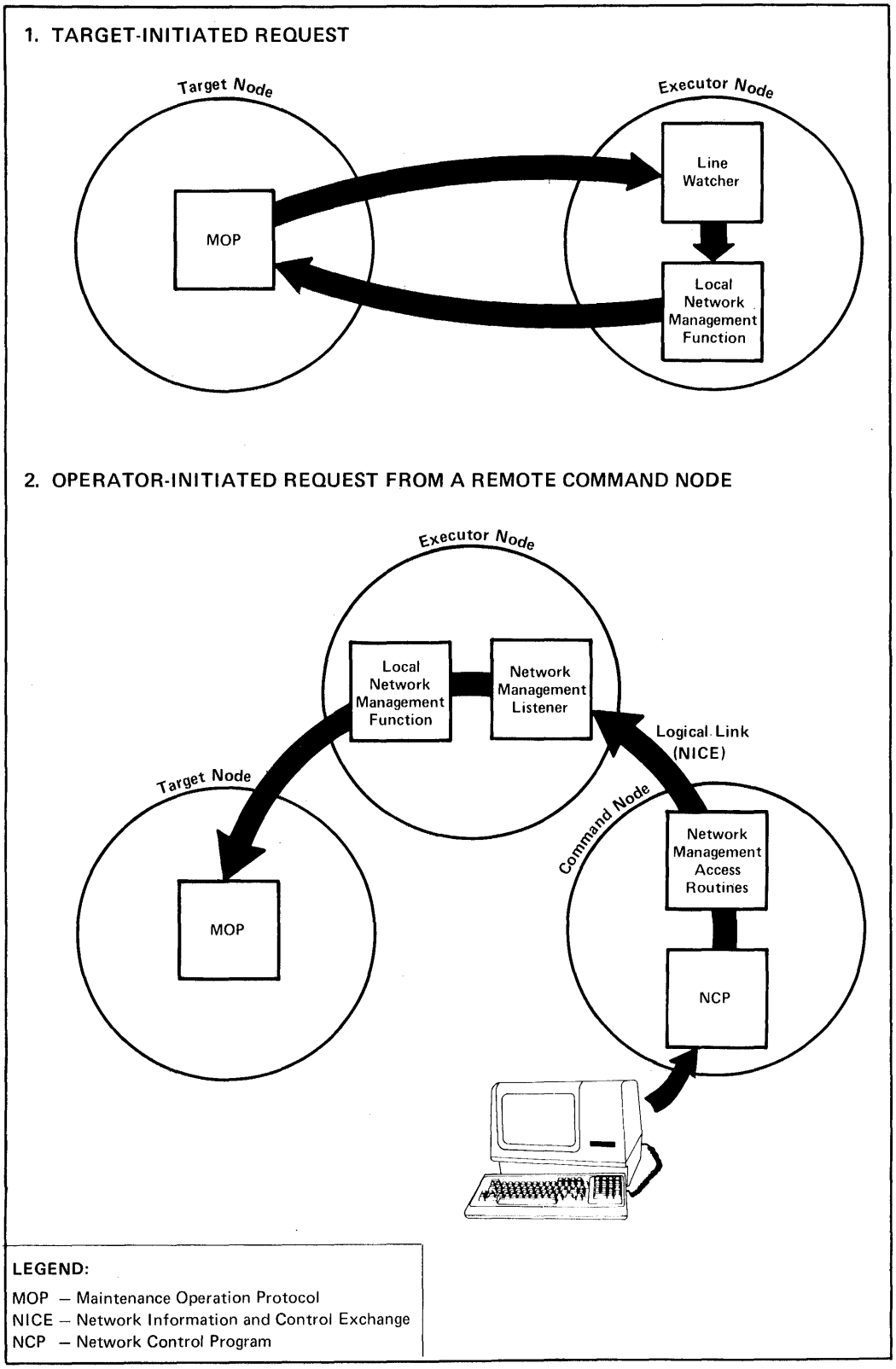


Figure 7-2: Down-Line Load Request Operation

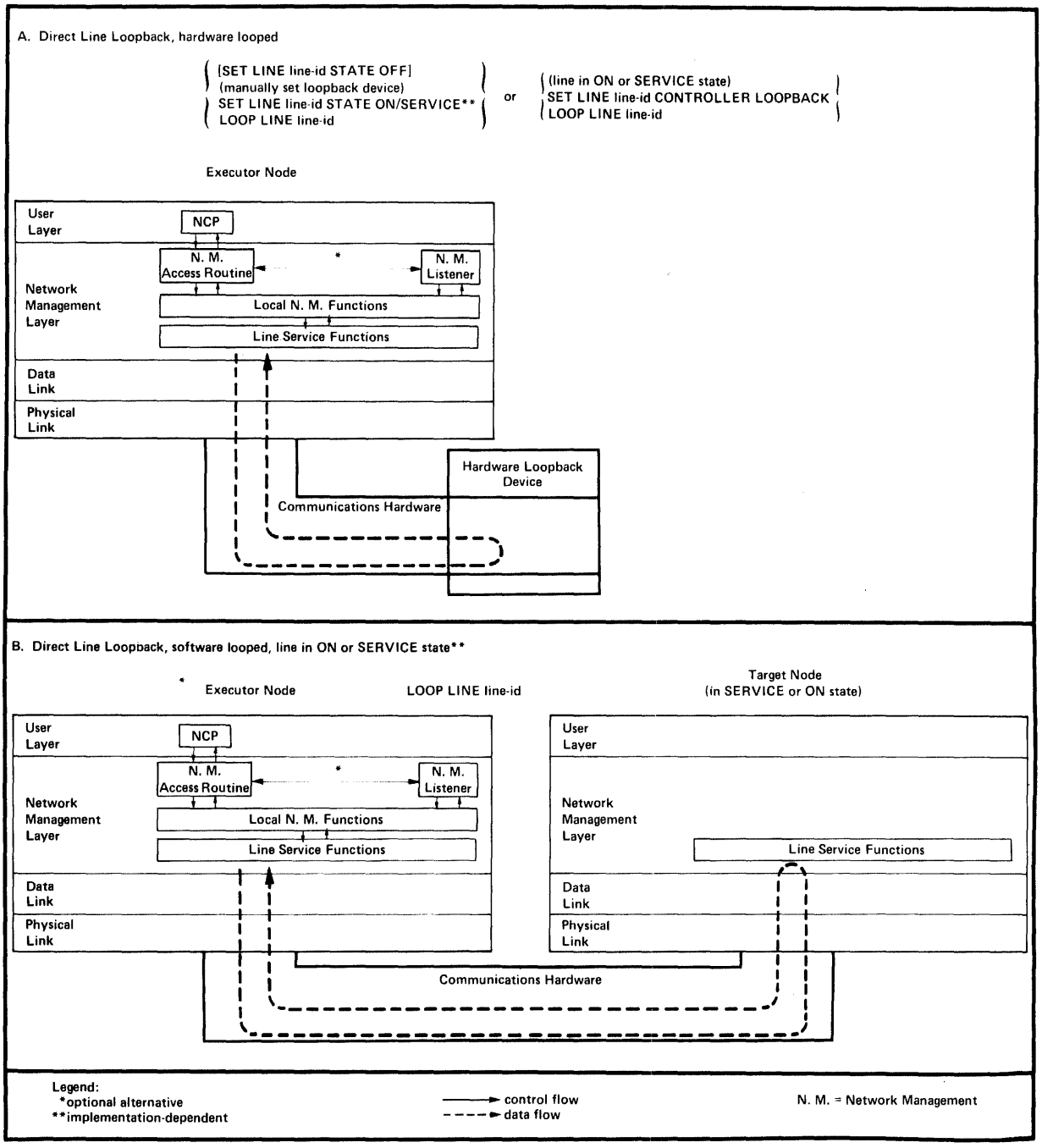


Figure 7-3: Line Loopback Tests

7.2.3 Node Loopback Testing

Node loopback testing consists of sending test messages over a logical link to be looped back at a specific point. The operator may set a hardware loopback device on a line, line device, or modem between the executor and adjacent target. Alternatively, software components can loop back test data. Different software components can be used. For example, FAL, a user program, or the Loopback Mirror can loop data back to their associated access routines. Figures 7–4 and 7–5 describe some types of node loopback tests.

One type of node loopback test uses the Network Management Loopback Access Routines and Loopback Mirror residing in the Network Application layer (Figure 7–4,B and Figure 7–5,C). These modules communicate over logical links using the Loopback Mirror Protocol.

Other loop tests use logical links, but do not use Network Management software. For example, Figure 7–5,B shows a test message travelling from one user task to another. Figure 7–5,D shows a file transfer used as a logical link test.

By using the NCP SET NODE LINE command, an operator can set up a special *loop node* path (Figure 7–4). In this case, if no hardware loopback device is set, the adjacent node's Transport layer loops back the test.

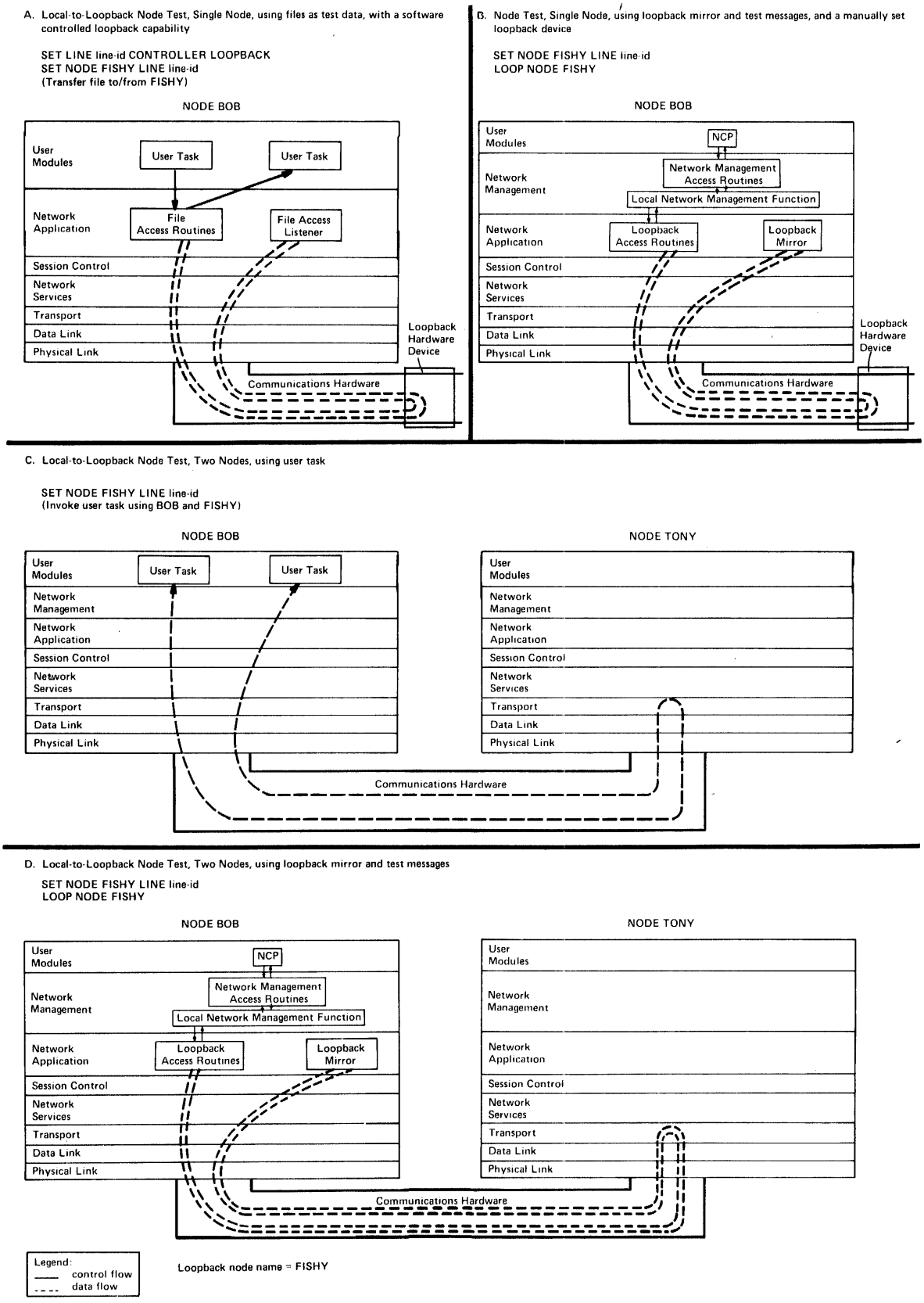


Figure 7-4: Examples of Node Level Testing Using a Loopback Node Name

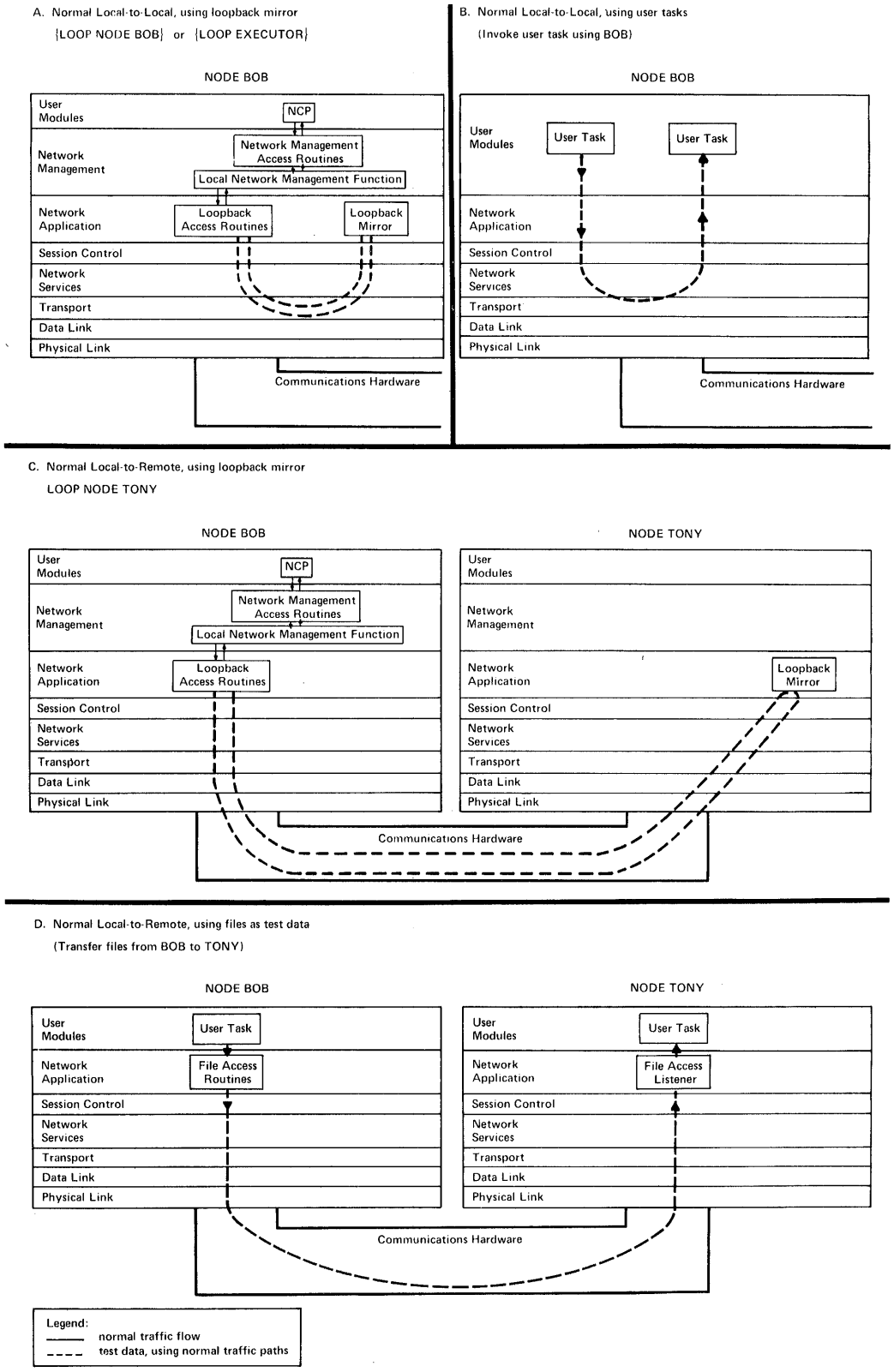


Figure 7-5: Examples of Node Level Logical Link Loopback Tests

7.2.4 Parameters, Counters, and Events

DNA specifies line, node, and logging parameters that Network Management can access. These are internal network values that are important enough for the system manager to examine, and, in some cases, change. These values are classified as characteristics or status. *Characteristics* generally remain constant until changed by a system manager. *Status* parameters reflect the condition of lines, nodes or loggers. For example, line state is a status parameter. The action of the network can change line and node states. The logging state is really a system-manager-controlled on/off switch.

Each parameter has a unique type number. Some examples of parameters are:

- Line cost
- Node address
- Line state
- Node cost
- Logging sink node

DNA also specifies line and node counters and events. These are internal network variables that keep track of errors and network activity at each layer.

The Network Management design specifies only counters and events that are useful to the system manager. For example, certain Transport counters enable a system manager to determine if there are too many packets being discarded by the network due to congestion. In this case, the manager can adjust parameters that control how packets are routed in order to better balance traffic loads. Additionally, the manager can decide whether or not he needs to order more communication facilities.

The Network Management design limits the overhead to the network involved in gathering statistics and errors as much as possible. For example, there are no Transport counters that detect bugs in the Transport code: Network Management assumes the code is correct. The Transport layer algorithms themselves have been designed to detect and recover from most failures which would disrupt the network operation.

The Local Network Management Functions interface to network counters (see Figure 7–1). The Event Logger receives event information (Figure 7–1). Changes in counter values trigger many events. The Local Network Management Functions display counters in response to user level requests. On the other hand, the Event Logger, once set up, records events for user use automatically.

Each DNA functional specification contains, if applicable, its Network Management interface, including counters and events. The events and counters for the Physical Link layer are specified in Appendix G of the *DNA DDCMP Functional Specification*. Appendix A of the *DNA Network Management Functional Specification* contains tables of all Network Management parameters, counters, and events.

7.3 Network Management Messages

The three Network Management protocols are:

1. **Network Information and Control Exchange (NICE) Protocol.** This handles most Network Management functions.
2. **Event Logger Protocol.** This handles event logging from remote nodes.
3. **Loopback Mirror Protocol.** This handles the node loopback function.

Table 7–1 describes the NICE messges.

Table 7–1: NICE Messages

Message	Description
Request Down-line Load	Requests a specified executor node to down-line load a target node.
Request Up-line Dump	Requests a specified executor node to dump the memory of a target node.
Trigger Bootstrap	Requests a specified executor node to trigger the bootstrap loader of a target node.
Test	Requests a specified executor node to perform a node or line loopback test.
Change Parameter	Requests a specified executor node to set or clear one or more Network Management parameters.
Read Information	Requests a specified executor node to read a specified group of parameters, counters, or events.
Zero Counters	Requests a specified executor node to either read and zero or zero a specified group of line or node counters.
System Specific	Requests a system-specific Network Management function.
Response	Provides request status and requested information in response to a NICE request.

There is only one event logger message, the Event message. It provides information when an event occurs. This information includes:

- The event sink (in other words, where the event is to be logged — one or more of console, file, or monitor)
- The event type and class
- The date and time the event occurred at the source node
- The name and address of the source node
- Whether the event related to a node or a line
- Specific data concerning the event

Table 7–2 describes the Loopback Mirror Protocol messages.

Table 7-2: Loopback Mirror Messages

Message	Description
Command	Requests a loop test and sends the data to be looped.
Response	Returns status information and the looped data.

Glossary

access control

Screening inbound connect requests and verifying them against a local system account file. Access control is an optional Session Control function.

active side

With regard to MOP loopback tests, the node that controls a test.

adjacent node

A node removed from the executor node by a single physical line.

ASCII

American Standard Code for Information Interchange. This is a seven-bit-plus-parity code established by the American National Standards Institute to achieve compatibility between data services.

bandwidth

The range of frequencies assigned to a channel; the difference, expressed in Hertz, between the highest and lowest frequencies of a band. The higher the bandwidth, the more the data throughput.

binary synchronous protocol

A data link protocol that uses a defined set of control characters and control character sequences for synchronized transmission of binary coded data between stations in a data communications system.

channel

The data path joining two or more stations, including the communications control capability of the associated stations.

characteristics

Parameters that are generally static values in volatile memory or permanent values in a permanent data base. A Network Management information type.

command node

The node where an NCP command originates.

congestion

The condition that arises when there are too many packets to be queued.

congestion control

The Transport component that manages buffers by limiting the maximum number of packets on a queue for a line. Also called transmit management.

control station

To the Data Link layer protocol, a station that has overall responsibility for orderly operation of a channel. See Figure G–1.

controller

The control hardware for a line. For a multiple line controller device the controller is responsible for one or more units. See Figure G–1. The controller identification is part of a line identification.

counters

Memory locations used to record error and performance statistics. A Network Management information type.

data flow

The movement of data from a source Session Control to a destination Session Control. NSP transforms data from Session Control transmit buffers to a network form before sending it across a logical link. NSP retransforms the data at the destination from its network form to its receive buffer form. Data flows in both directions (full-duplex) on a logical link.

data link

A logical connection between two stations on the same channel. In the case of a multipoint line, there can be multiple data links. See Figure G–1.

datagram

A unit of data passed between Transport and the Network Services layer. When a route header is added, it becomes a packet.

down-line load

To send a copy of a system image or other file over a line to the memory of a target node.

end node

A topological description of a nonrouting node. Since a nonrouting node cannot perform route-through and supports only a single line, it must be an end node. However, it is also possible for a routing node with a single line to be an end node.

end user module

A module that runs in the "user space" of a network node and communicates with Session Control to obtain logical link service.

error control

The NSP function that ensures the reliable, sequential delivery of NSP data messages. It consists of sequencing, acknowledgment, and retransmission mechanisms.

events

Occurrences that are logged for recording by Network Management.

executor node

The node in which the active Local Network Management Function is running (that is, the node actually executing an NCP command).

flow control

The NSP function that coordinates the flow of data on a logical link in both directions, from transmit buffers to receive buffers to ensure that data is not lost, to prevent buffer deadlock, and to minimize communications overhead.

framing

The DDCMP component that synchronizes data at the byte and message level.

full-duplex channel

A channel that services concurrent communications in both directions (to and from the station).

half-duplex channel

A channel that permits two-way communications, but in only one direction at any instant.

hierarchical network

A computer network in which processing control functions are performed at several levels by computers specially suited for the functions performed, for example, in a factory or laboratory automation.

hop

To the Transport layer, the logical distance between two adjacent nodes in a network.

host node

The node that provides services for another node (for example, during a down-line task load).

ISAM

Indexed Sequential Access Method. This access method is a combination of random and sequential access. Random access is used to locate a sequence of records and then access is switched to sequential to read the remaining records in the series.

key

A data item used to locate a record in a random access file system.

key field

For direct and indexed files, the position of the key within the record.

leased line

A nonswitched circuit leased from a public utility company (common carrier) for exclusive use.

line

A physical path. In the case of a multipoint line, each tributary is treated as a separate line.

line cost

A positive integer value associated with using a line. Messages are routed along the path between two nodes with the smallest cost.

line level loopback

Testing a specific data link by sending a message directly to the data link layer and over a wire to a device that returns the message to the source.

link management

The DDCMP component that controls transmission and reception on links connected to two or more transmitters and/or receivers in a given direction.

logging

Recording information from an occurrence that has potential significance in the operation and/or maintenance of a network in a potentially permanent form where it can be accessed by persons and/or programs to aid them in making real-time or long-term decisions.

logging sink node

A node to which logging information is directed.

logical link

A virtual channel between two end users in the same node or in separate nodes. Session Control acts as an interface between an end user requiring logical link service and NSP, which actually creates, maintains, and destroys logical links.

loop node

A special name for a node that is associated with a line for loop testing purposes. The NCP SET NODE LINE command sets the loopback node name.

master station

A station that has control of a channel at a given instant, for the purpose of sending data messages to a slave station (whether or not it actually does).

maximum cost

An operator-controllable Transport parameter that defines the point where the routing decision algorithm in a node declares another node unreachable because the cost of the least costly path to the other node is excessive. For correct operation, this parameter must not be less than the maximum path cost of the network.

maximum hops

An operator-controllable Transport parameter that defines the point where the routing decision algorithm in a node declares another node unreachable because the length of the shortest path between the two nodes is too long. For correct operation, this parameter must not be less than the network diameter.

maximum path cost

The routing cost between the two nodes of the network having the greatest routing cost, where routing cost is the cost of the least cost path between a given pair of nodes. In Figure 3–1, the maximum path cost is 9.

maximum path length

The routing distance between the two nodes of the network having the greatest routing distance, where routing distance is the length of the least cost path between a given pair of nodes. In Figure 3–1, the maximum path length is 4.

maximum visits

An operator-controllable Transport parameter that defines the point where the packet lifetime control algorithm discards a packet which has traversed too many nodes. For correct operation, this parameter must not be less than the maximum path length of the network.

message exchange

The DDCMP component that transfers data correctly and in sequence over a link.

monitor

A logging sink that is to receive a machine-readable record of events for possible real-time decision-making.

multiple line controller

A controller that can manage more than one unit. (DIGITAL multiple line controllers are also called multiplexers.) See Figure G–1.

multiplex

To simultaneously transmit two or more data streams on a single channel. In DNA, NSP is the only protocol that multiplexes.

multipoint channel

A channel connecting more than two stations. Also referred to as multidrop. See Figure G–1.

network diameter

The reachability distance between the two nodes of the network having the greatest reachability distance, where reachability distance is the length of the shortest path between a given pair of nodes. In Figure 3–1 the network diameter is 3.

node

An implementation of a computer system that supports Transport, Network Services, and Session Control. Each node has a unique node address.

node address

The unique numeric identification of a specific node.

node level loopback

Testing a logical link using messages that flow with normal data traffic through the Session Control, Network Services, and Transport layers within one node or from one node to another and back. In some cases node level loopback involves using a loopback node name associated with a particular line.

node name

An optional alphanumeric identification associated with a node address in a strict one-to-one mapping. No name may be used more than once in a node. The node name must contain at least one alpha character.

node name mapping table

A table that defines the correspondence between node names and node addresses or channel numbers. Session Control uses the table to identify destination nodes for outgoing connect requests and source nodes for incoming connect requests.

nonrouting node

A Phase III DECnet node that contains a subset of routing modules (select process and receive process) and can deliver and receive packets. It is connected to the network by a single line.

object type

Numeric value that may be used for process or task addressing by DECnet processes instead of a process name.

Other Data

The NSP Data Request, Interrupt Request, and Interrupt messages. These are all the NSP data messages other than Data Segment. Because all *Other Data* messages move in the same data subchannel, it is sometimes useful to group them together.

packet lifetime control

The Transport component that monitors lines to detect if a line has gone down, and prevents excessive looping of packets by discarding packets that have exceeded the maximum visit limit.

packet switching

See route-through.

parallel data transmission

A data communication technique in which more than one code element (for example, bit) of each byte is sent or received simultaneously.

parameters

DNA values to which Network Management has access for controlling and monitoring purposes.

passive side

With regard to MOP loopback tests, the node that loops back the test messages.

path

The route a packet takes from source node to destination node. This can be a sequence of connected nodes between two nodes.

path cost

The sum of the line costs along a path between two nodes.

path length

The number of hops along a path between two nodes.

physical link

An individually hardware addressable communications path. In terms of hardware, a physical link is a combination of either a channel and its controllers or a channel, controllers, and units. See Figure G–1.

piggybacking

Sending an acknowledgment within a returned data message.

pipelining

Sending messages without waiting for individual acknowledgment of each successive message.

point-to-point channel

A channel connecting only two stations.

raw event

A logging event as recorded by the source process, incomplete in terms of total information required.

reachable node

A node to which a routing node believes it can direct a packet.

reassembly

The placing of multiple, received data segments by NSP into a single Session Control receive buffer.

remote node

To one node, any other network node.

request count

This term has two definitions in NSP. 1) Variables that NSP uses to determine when to send data. 2) Values sent in Link Service (Data Request and Interrupt Request) messages.

The flow control mechanism adds the request counts received in Data Request and Interrupt Request messages to the request counts it maintains to determine when to send data.

retransmission

The resending of NSP or DDCMP data messages that have not been acknowledged within a certain period of time. This is part of NSP's and DDCMP's error control mechanisms.

RMS

Record Management Services. This file system will be used on all major DIGITAL systems except where space is limited (for example, RT-11). In addition to access modes provided by previous file systems, RMS provides random access for direct and indexed files and ISAM.

route-through

The directing of packets from source nodes to destination nodes by one or more intervening nodes. Routing nodes permit route-through. Also called packet switching.

routing

Directing data message packets from source nodes to destination nodes.

routing node

A Phase III DECnet node that contains the complete set of Transport modules, and can deliver, receive, and route packets through.

satellite node

With regard to MOP functions, the node being loaded, dumped, tested, or restarted. A satellite node is dependent on its host for these functions.

segment

The data carried in a Data Segment message. NSP divides the data from Session Control transmit buffers into numbered segments for transmission by Transport.

segmentation

The division of normal data from Session Control transmit buffers into numbered segments for transmission over logical links.

slave station

A tributary station that can send data only when polled or requested to by a master, control station. In some multiplex situations a tributary can act as both slave and master.

star topology

A network configuration in which one central node is connected to more than one adjacent, end node. A star can be a subset of a larger network.

station

With regard to the Data Link layer protocol, a termination on a data link. A station is a combination of the physical link (communication hardware) and the data link protocol implementation. See Figure G–1.

status

Dynamic information relating to a network such as a line state. A Network Management information type.

subchannel

A logical communications path within a logical link that handles a defined category of NSP data messages. Because Data Segment messages are handled differently from *Other Data* messages, the two types of messages can be thought of as travelling in two different subchannels.

synchronous serial data transmission

A data communication technique in which the information required to determine when each byte begins is sent at the beginning of a group of bytes (the "sync bytes"). The time interval between successive bytes in the group is zero. The time interval between successive groups of bytes is unspecified.

target node

The node that receives a memory image during a down-line load, generates an up-line dump, or loops back a test message.

topology

The physical arrangement and relationships of interconnected nodes and lines in a network. A legal topology satisfies the requirements of the Transport specification.

transparent data

Binary data transmitted with the recognition of most control characters suppressed. DDCMP provides data transparency because it can receive, without misinterpretation, data containing bit patterns that resemble DDCMP control characters.

tributary

A station on a channel that is not a control station. See Figure G–1.

unit

The hardware controlling one channel on a multiple line controller. A unit, a controller, and associated Data Link modules form a station. See Figure G–1.

unreachable node

A node to which a routing node has determined that the path exceeds the maximum hops of the network.

up-line dump

To send a copy of a target node's memory image up a line to a file at the host node.

wild card

With regard to DAP, an asterisk (*) that replaces an element in a file specification. The asterisk specifies all known items in the range indicated by its position. For example, FILE.*;* specifies all known versions and types of all files named FILE.

With regard to Network Management, an asterisk that replaces an element in a line identification. The asterisk specifies all known lines in the range indicated by its position in the identification. For example, DMC* specifies all known controllers on line DMC.

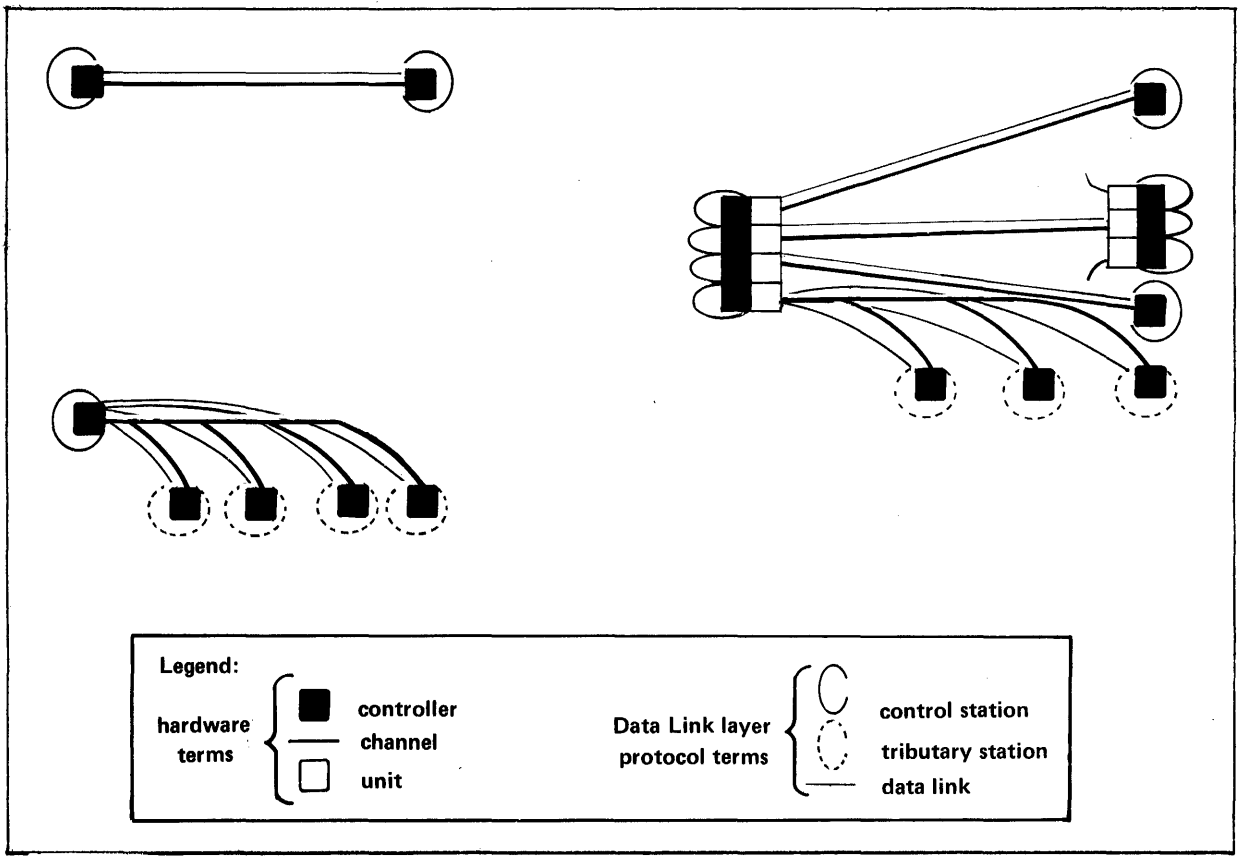


Figure G-1: Link Terminology

READER'S COMMENTS

NOTE: This form is for document comments only. DIGITAL will use comments submitted on this form at the company's discretion. If you require a written reply and are eligible to receive one under Software Performance Report (SPR) service, submit your comments on an SPR form.

Did you find this manual understandable, usable, and well-organized? Please make suggestions for improvement.

Did you find errors in this manual? If so, specify the error and the page number.

Please indicate the type of user/reader that you most nearly represent.

- ☐ Assembly language programmer
- ☐ Higher-level language programmer
- ☐ Occasional programmer (experienced)
- ☐ User with little programming experience
- ☐ Student programmer
- ☐ Other (please specify)_____

Name_____ Date_____

Organization_____

Street_____

City_____ State_____ Zip Code or Country_____

—Do Not Tear - Fold Here and Tape —

digital

No Postage Necessary if Mailed in the United States

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO.33 MAYNARD MASS.

POSTAGE WILL BE PAID BY ADDRESSEE

SOFTWARE DOCUMENTATION
146 MAIN STREET ML 5-5/E39
MAYNARD, MASSACHUSETTS 01754

— Do Not Tear - Fold Here and Tape —

Cut Along Dotted Line