Chapter 1
Fundamentals of Computer Communication Networks

1.1. WHY DO WE NEED PROTOCOLS?

First we attempt to answer a more fundamental question: What is a protocol? According to McQuillan and Cerf [MCQU 78], "the word protocol has been borrowed from the parlance of conventional social behavior to describe the orderly exchange of information between separate pieces of equipment." We shall, however, use the word *protocol* with a more concrete meaning. A protocol in this text is a set of parallel programs that provides one or more functions within a computer communication network environment, as defined by the protocol's service specification. There is a wide variety of such functions. They include, for examples, the management of logical channels, the opening/closing of logical channels, reliable data transfers at different levels (data link, host-to-host, etc.), routing, flow and congestion controls, deadlock avoidance, multiple access in broadcast channels, and concurrency control of distributed databases, among many others. Most of these functions fall into two broad categories: (1) communication functions, and (2) network resource allocation functions. Protocols that implement these functions will be referred to as *communication protocols* and *resource allocation protocols*, respectively.

The ultimate objective of a computer communication network is to allow *user processes* located in different computing equipment to exchange data reliably. This is really the same objective as interprocess communication within a multiprogramming or a multiprocessor environment. However, we must do without the reliability and speed of computer buses. As the geographical distance between communicating user processes increases, the general trend is that the speeds of the available communication channels decrease, their error rates go up, their propagation delays go up, and their costs go up drastically.

For examples, long-distance telephone lines at 56 Kbps with a bit error rate of 1 in 10^7 are considered to be of very high quality. Satellite channels can provide transmission rates of several to tens of Mbps over long distances, but they have a propagation delay of about 0.27 seconds. Earth stations for satellite communications at the 4 and 6 GHz frequencies are large and expensive. Satellite communications at higher frequencies, such as 12 and 14 GHz, permit the use of smaller, less expensive earth stations suitable for installation on user sites [BARN 77]. Using forward error correction coding (at a rate of approximately two-thirds), a bit error rate of 1 in 10^7 is achievable. However, transmissions at such high frequencies suffer from the problem of rain attenuation which decreases availability. On the other hand, local area networks which span only hundreds to thousands of feet have relatively good transmission characteristics. Presently, transmission speeds range from a few hundred Kbps for some to Ethernet's 10 Mbps and to Hyperchannel's 50 Mbps. Error rates ranging from 1 packet in 6,000 to 1 packet in 2,000,000 were reported for Ethernet [SHOC 80].

In Figure 1-1, we illustrate a set of computing equipment to be referred to as *user* (or *host*) nodes. A network of communication facilities provides the physical interconnection of these user nodes. Logical communication paths between processes residing in user nodes are provided by protocols. Because of the relatively high cost of the communication facilities, resource allocation protocols are often employed to dynamically share these facilities in some fashion. Because of errors and relatively large and unpredictable transit

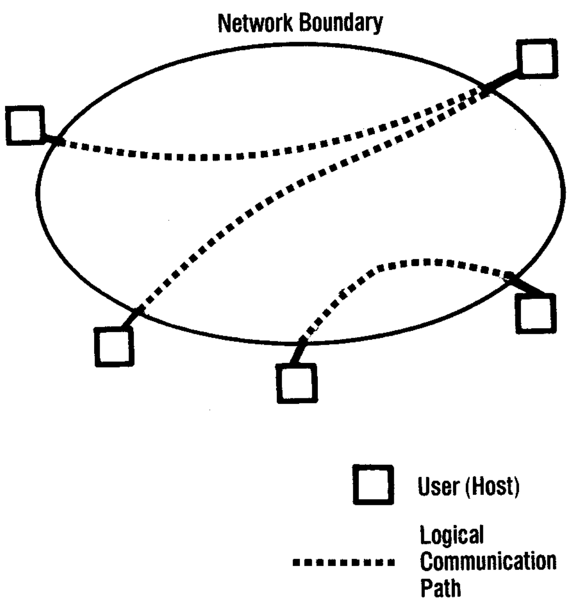


Figure 1-1. Logical communication paths between users.

delays, protocols are usually needed to provide the functions of error control and in-sequence delivery. For the same reasons, protocols are usually needed to reach various agreements between users, e.g., opening/closing of a dialog and security keys to be used. These are referred to as communication protocols.

1.2. SYNCHRONIZATION FUNCTIONS TO ACHIEVE COMMUNICATION

Network users correspond to various kinds of terminals and computers, large or small, which are generally referred to as data terminating equipment in the protocol standards literature. Each user node typically has a separate device, which we shall call a *transceiver*, with the the functional capability of sending and receiving bits serially over a communication link (or path). Examples of transceivers are modems for broadband channels or line driver-receiver units for baseband channels.

For this section, we shall ignore the complicating issues of network topologies and switching techniques. These will be addressed in Sections 1.3 and 1.4. Let us identify the basic functions for achieving communication between two processes residing in user nodes connected by a single communication link (Figure 1-2). These functions are referred to as *synchronization functions* and they can be structured into a hierarchy of five levels [LAM 83]. Figure 1-3 illustrates the levels as well as the concept of increasing time granularity which can be explained as follows. Consider the time points of event occurrences at the two communicating user nodes. Level *n* has a larger time granularity than that of level *n*-1 because the time points at which "synchronism" is achieved for level *n* are embedded in the time points at which "synchronism" is achieved for level *n*-1. This concept will become clear as we describe what the synchronization functions are. A discussion of the five levels of functions will proceed from the bottom of the hierarchy upwards.

Bit synchronization

This level deals with the receiver's ability to retrieve bits serially from an incoming signal. The users must agree upon the encoding and modulation techniques. The receiver must also be able to retrieve bit-timing information from the incoming signal. The subject of bit synchronization is beyond the scope of this text. The reader is referred to books by Schwartz [SCHW 80] and Viterbi [VITE 66] that cover this subject.

Frame synchronization

A frame is defined here to be a serial block of bits containing both data and control information. It is the basic unit of communication between two user nodes connected by a physical link. (This is consistent with the meaning of "frame" in the class of data link protocols exemplified by HDLC [LAM 83].) This level establishes conventions to delimit within a continuous stream of bits the beginning bit and the ending bit of a frame. Another function at this level is to detect errors in frames received and to filter out corrupted frames. With frame synchronization, uncorrupted blocks of bits containing data and/or control information can be sent from one user node to

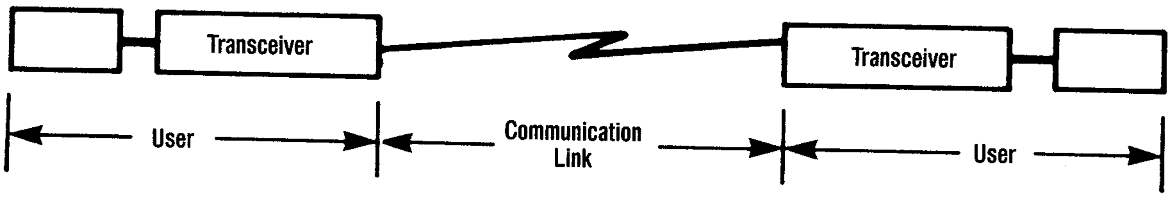


Figure 1-2. Two communicating users.

another via a dedicated physical link, although losses may occur.

Multiple access synchronization

This level of function is needed if a physical communication channel is shared by multiple sender-receiver pairs. Since frames transmitted by different senders are interleaved in time, an addressing scheme is needed to identify the sender and receiver of each frame. In addition, control messages and algorithms are needed to synchronize the actions of senders to avoid (or resolve) channel-access conflicts efficiently. With multiple access synchronization, uncorrupted frames can be sent from one user node to another via a *shared communication* link. Again, losses may occur.

Content synchronization

Given that user nodes can send frames to each other, this level of function is concerned with the information content of each frame, in particular: (1) how to differentiate between data and control information within a frame, (2) how to encode control messages, and (3) how to use control messages to perform error control and to achieve in-sequence delivery of frames, if these functions are desired by the user processes.

Dialog synchronization

The basic function at this level is the initiation and subsequent termination of a dialog between two user processes during which the transfer of frames containing data can take place. (Instead of a dialog between two processes, one can also think of a broadcast from one to many or a conference call involving multiple user processes.) In addition to reaching agreement on the status (up or down) of a logical connection between user nodes, other possible synchronization functions at this level are:

1. agreement on the relationship between the user nodes, e.g., balanced, and master-slave;
2. agreement on the mode of interaction suitable for a particular application, e.g., file transfer, database inquiry/response, and digital speech traffic;
3. agreement on flow control parameters; and
4. agreement on encryption/decryption keys for secure data transfer.

Recovery actions are needed when a disagreement is detected or when a connection is deemed to be lost (for instance, lack of response to a previous inquiry). This level of function also includes the responsibility of reporting errors and disagreements that cannot be recovered to the user processes.

In summary, we observe that bit and frame

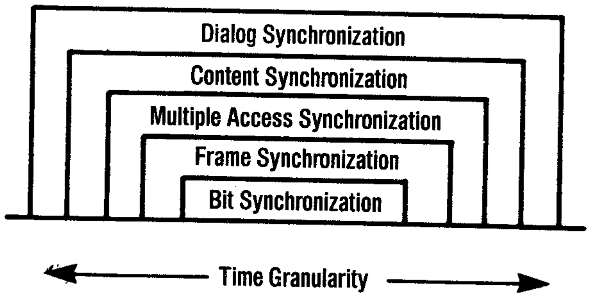


Figure 1-3. A hierarchy of synchronization functions.

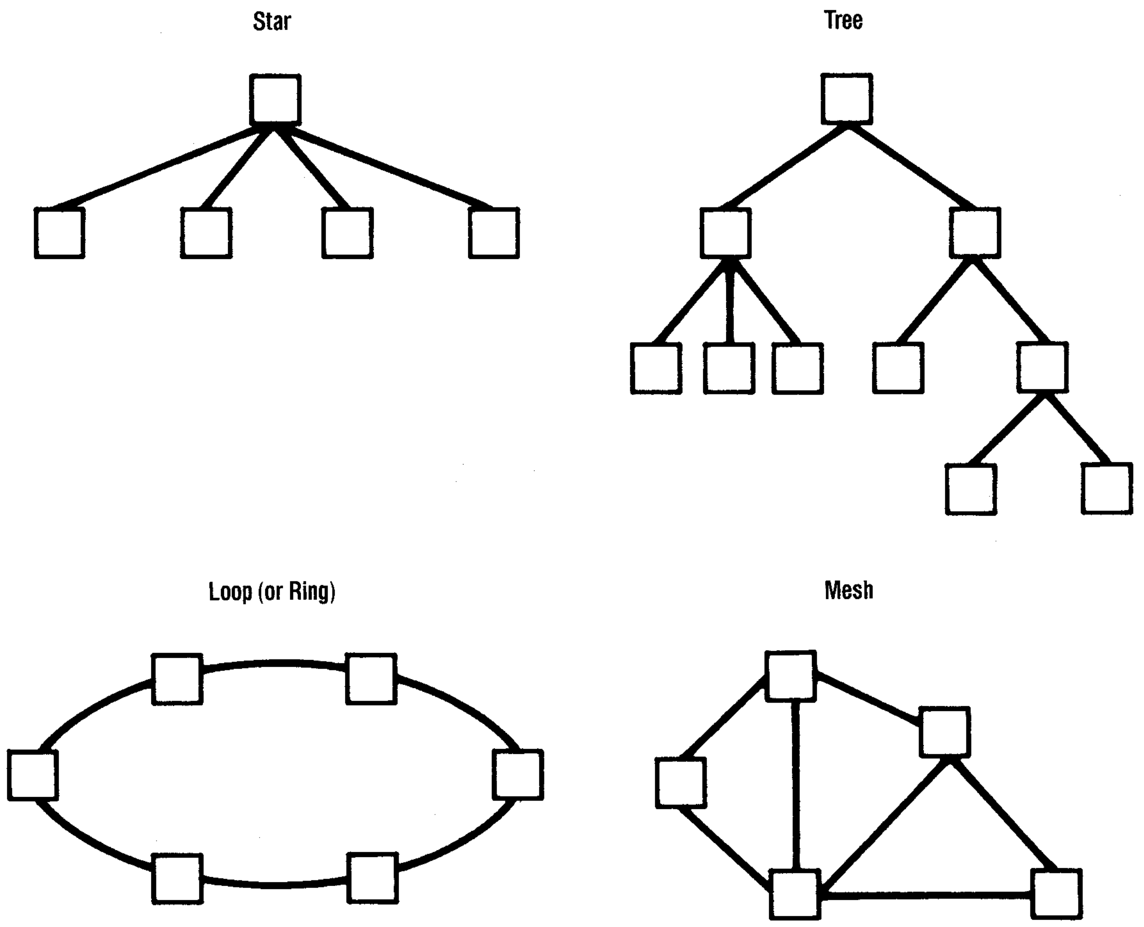


Figure 1-4. Topologies of networks with point-to-point links.

synchronizations are necessary for the most rudimentary form of communication between two different machines. Functions at the top three levels are not absolutely necessary, but they are useful for improving the efficient utilization of the communication link (in the case of multiple access synchronization functions) or for improving the quality and characteristics of the logical connection (in the case of content and dialog synchronization functions).

1.3. NETWORK TOPOLOGY

In Figure 1-1, the topology of the network of communication facilities is not shown. The topology of a network depends on the communication media involved. With point-to-point links, some common topologies are shown in Figure 1-4: star, loop (or ring), tree, and mesh. We define a mesh topology to be one that has at least two disjoint paths between any pair of nodes. Note that this definition is also satisfied by a loop. The loop is a limiting case of our definition and will be considered separately.

Broadcast networks are those that employ a broadcast channel for interconnection. Figure 1-5 illustrates three broadcast networks employing a coaxial cable, a radio channel and a satellite channel. Broadcast networks have the property that any node can transmit into the broadcast channel, and each transmission in the channel can be received by all nodes in the network.

The topology of a network may be organized into several levels. Typically, there are two levels,

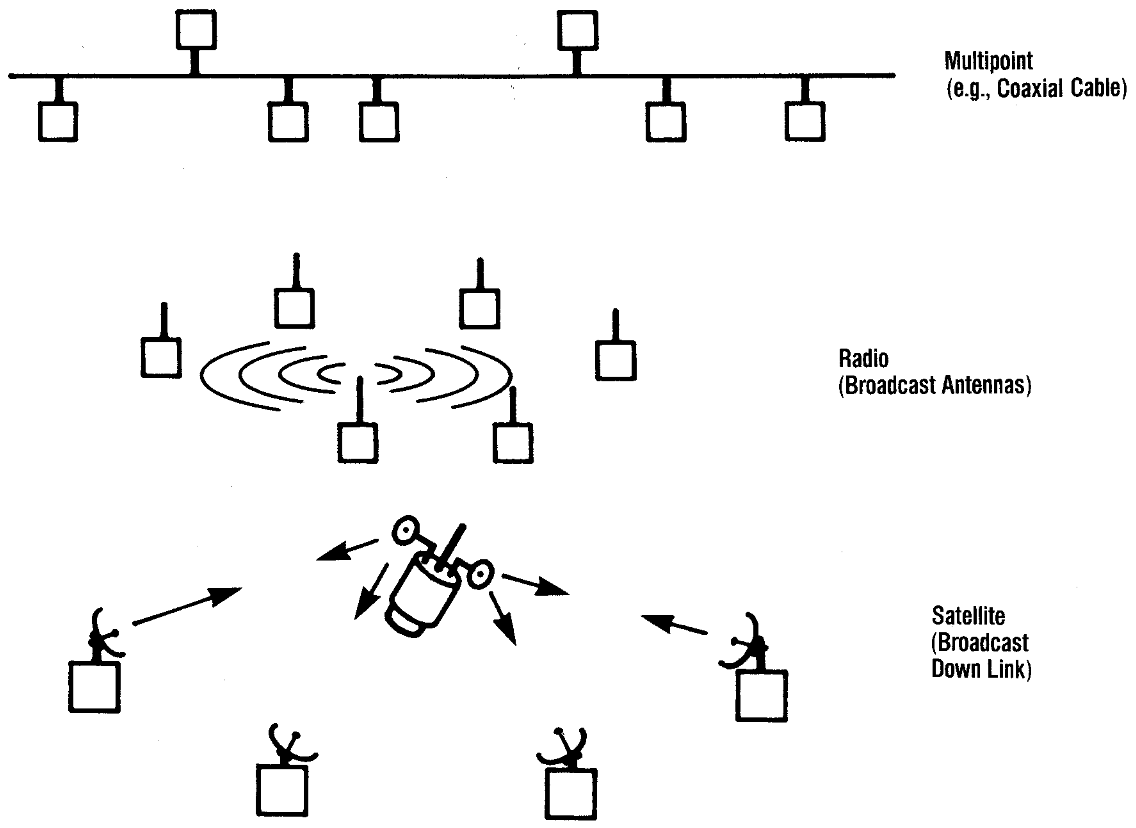


Figure 1-5. Broadcast networks.

i.e., a wide area network (WAN) serving as the backbone at the upper level, with access to it provided by many local area networks (LANs) at the lower level. Any one of the above topologies can be used for a WAN or a LAN. However, the topologies typically used for a LAN are: star, multipoint, loop, tree, and radio. The topologies typically used for a WAN are: mesh (for reliability), and satellite (for high capacity or for complete connectivity). A WAN may also be constructed with a combination of point-to-point links and satellite broadcast channels. LANs that access the WAN may have different topologies at different locations. (An example of a two-level topology is shown in Figure 1-6.)

A hierarchy of more than two levels may be used for several reasons. First, the topology of a very large network may be organized into several levels for efficiency considerations (e.g., to reduce the size of routing tables). The telephone network is a prime example [MART 76]. Second, the interconnection of autonomous WANs will immediately give rise to a three-level topology, as shown in Figure 1-7. (The interconnection of networks is addressed in Chapter 6.)

1.4. SWITCHING TECHNIQUES

The transmission channels of a communication network constitute the network's primary resource. This resource may be in the form of a single point-to-point communication channel, a network of point-to-point channels (any topology), a broadcast channel, or some combination of the above. Probably the first and most important choice in designing a network's resource allocation mechanisms is the network's *switching technique.*

1.4.1. Sharing a Point-to-Point Channel

Consider a point-to-point communication link between two clusters of users, such as that shown in Figure 1-8. Assume the link to be full-duplex, i.e., there are two communication channels for independent transmission in each of the two directions. (A half-duplex link is one in which transmission can take place in either direction but not simultaneously.)

Multiplexing technology

Each communication channel can be multiplexed into a number of subchannels. There are two different kinds of multiplexing: frequency division multiplexing (FDM) and time division multiplexing (TDM). Figure 1-9 illustrates FDM. The channel bandwidth is subdivided, and a subchannel corresponds to a segment of the bandwidth for all time. Figure 1-10 illustrates TDM. Time is divided into slots which are organized into frames. The entire bandwidth is available to a subchannel, but only during a designated time slot in consecutive frames. The term space division multiplexing is sometimes used to refer to a collection of subchannels, each of which is a separate physical link (e.g., a bundle of wires).

Fixed assignment of channels

This is the simplest resource allocation method. The communication channel to be shared is multiplexed into a number of subchannels, sufficient to allocate one to each sender-receiver pair present. Note that the subchannels do not have to be the same "size," but in practice a subchannel must be large enough to satisfy the response time requirement of the sender-receiver pair using it. For data communication, fixed-assigned subchannels are likely to be very poorly utilized for two reasons. First, user pairs with fixed-assigned subchannels may not be engaged in a dialog most of the time. This problem can be remedied by the demand assignment strategy to be discussed below. Second, even if a user pair is engaging in a dialog, a subchannel will still be lightly utilized if the sender generates data in a bursty manner, i.e., bursts of traffic in between long periods of silence. Highly bursty traffic is typical of most interactive computer applications [LAM 78].

Demand assignment of channels

The utilization of subchannels can be improved by having fewer subchannels than sender-receiver pairs, with the assumption that only some of the user pairs present are engaging in a dialog at any time. A subchannel is allocated only when there is a request to initiate a new dialog and is deallocated at the termination of the dialog. Demand-assigned subchannels will have higher utilization than fixed-assigned subchannels. The price to pay is some processing capability in the

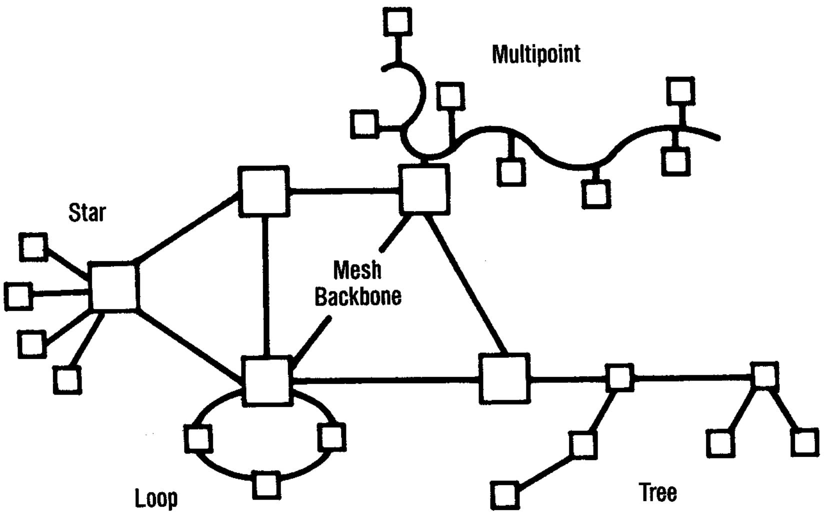


Figure 1-6. An example of a two-level topology.

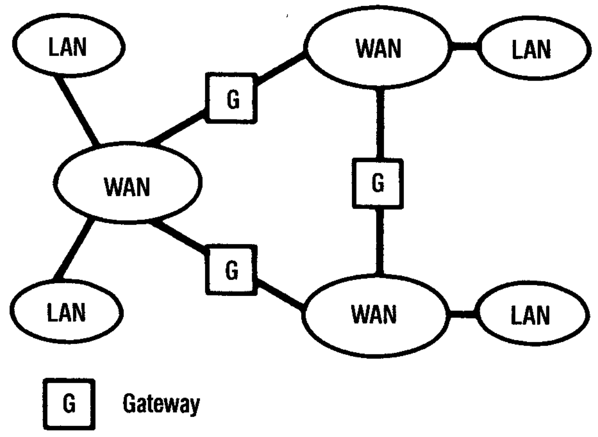


Figure 1-7. WANs interconnected by gateway nodes.

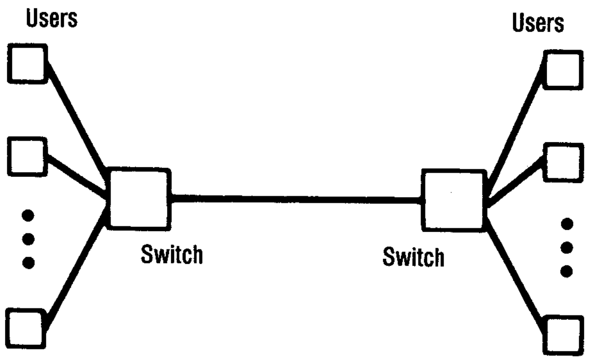


Figure 1-8. A shared point-to-point link.

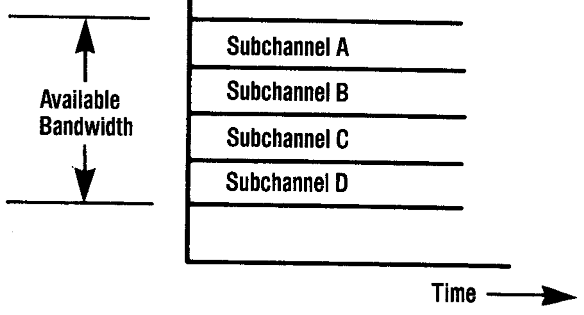


Figure 1-9. An illustration of Frequency Division Multiplexing (FDM).

switching nodes, as shown in Figure 1-8, as well as some delay that will be incurred whenever a request cannot be granted because no free subchannel is available. Note that a subchannel is allocated to a user for the entire duration of a dialog. If the user generates bursty traffic, there will still be long periods of silence during which the subchannel will not be utilized.

Statistical multiplexing

This technique is also called asynchronous time division multiplexing (ATDM). It is intended to allocate transmission bandwidth to a sender, one data unit at a time; in particular, allocation is made only when a data unit is waiting to be sent [CHU 69]. The switching node in this case is sometimes referred to as a data concentrator. Logically, it corresponds to simply a single-server queue (Figure 1-11). The communication channel is the server and the queue is formed of the data units generated by the user nodes that share the channel. This method will allow the communication channel to be utilized very efficiently even if each user node generates bursty traffic individually. What price do we pay for this improved efficiency? First, since data units belonging to different sender-receiver pairs are interleaved in the queue and in the communication channel, they must contain some form of identification (e.g., sender and receiver addresses, or logical channel number). Second, buffers are needed for the queue. And third, some processing capability is needed in the switching node to manage the queue and its scheduling.

The superiority of statistical multiplexing is based upon the phenomenon of scaling in queueing theory. The average delay-throughput performance of a single high-speed channel is better than that of multiple low-speed channels, even though the total amount of bandwidth is the same in each case [KLEI 76].

1.4.2. Sharing a Network of Point-to-Point Links

For user nodes interconnected by a network of communication channels and switching nodes, a routing algorithm is needed to establish a path, statically or dynamically, between each pair of user nodes. (Routing algorithms are covered in Chapter 5.) For this section, it suffices for us to consider a single path between a pair of communicating user nodes as shown in Figure 1-12. The switching techniques can again be divided into two categories: (1) allocation of channels to form end-to-end paths between pairs of users, and (2) allocation of bandwidth to data units (belonging to different source-destination user pairs) waiting in queues.

Channel-based techniques

Each link between two nodes in Figure 1-12 is a collection of channels or circuits. (We follow the common usage that a channel is a means for one-way transmission and a circuit is a means for two-way transmission. Also, we no longer care whether these channels are obtained by time, frequency, or space division multiplexing.) The path determined by a routing algorithm consists of a sequence of communication channels (circuits). As before, these channels (circuits) may be fixed-assigned indefinitely or *demand-assigned* only when there is a request to initiate a dialog between two user nodes and then deallocated at the termination of the dialog. These techniques are known as *circuit switching* (CS) and have been used for a long time in telephone networks.

Circuit switching has worked well as a resource allocation mechanism for networks that handle voice traffic. Channel-based techniques, in general, are appropriate if the users tend to communicate at a fairly constant rate for a long period of time (relative to the time necessary to set up the end-to-end path). Note that with circuit switching, the communicating user nodes must have compatible equipment; the network provides none of the functions mentioned in Section 1.2. In particular, the communicating user nodes must use the same code and operate at the same speed etc.

Figure 1-13(a) illustrates the timing of events along a path between two user nodes when circuit switching is used. Suppose that a user makes a request to set up a connection between itself and another user. A call-request signal then travels through the network according to some routing algorithm, seizing available channels along the way to form an end-to-end path. A call-accept signal travelling back informs the requesting user that the communication path has been set up and that communication may begin. It is obvious from Figure 1-13(a) that, if the connection time is short relative to the setup time or if there are long periods of silence during a connection, the circuits will be poorly utilized.

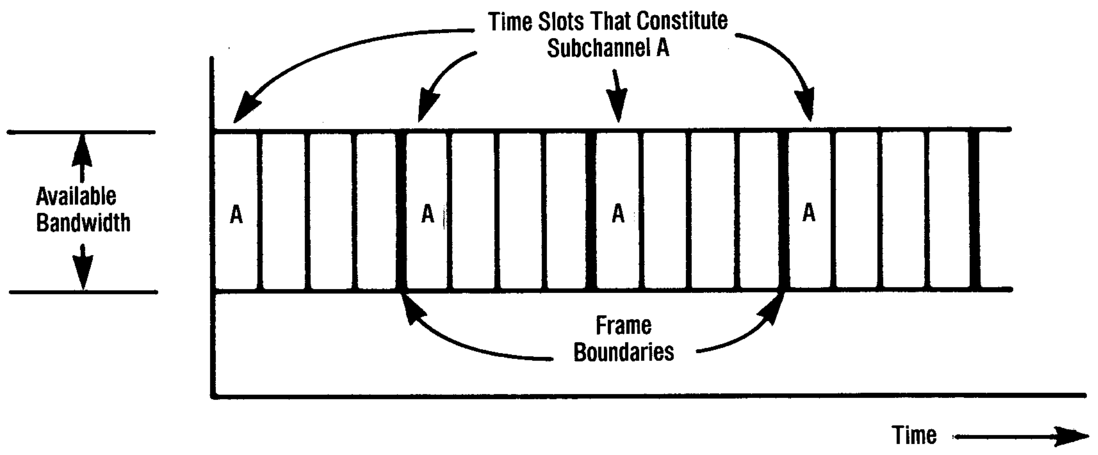


Figure 1-10. An illustration of Time Division Multiplexing (TDM).

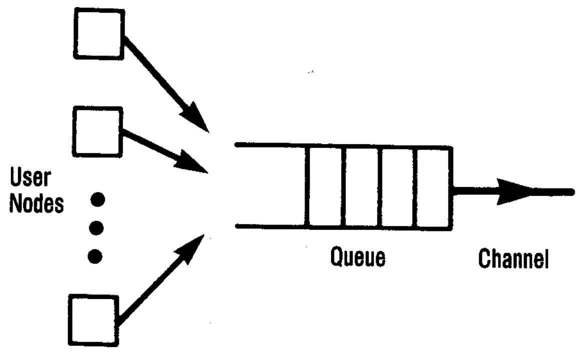


Figure 1-11. A single-server queue.

Queueing-based techniques

The use of queueing to share a communication channel can be extended to a network of channels that interconnect geographically distributed users. As before, the entire bandwidth of each communication channel is allocated to waiting data units, one at a time.

Each data unit travels from its source user node to its destination user node via a sequence of switching nodes along a path determined by some routing algorithm. Upon the receipt of a data unit, each switching node stores a copy of it and forwards the data unit to the next node in its path. Some queueing delay is encountered if the outgoing channel selected is busy. (The stored copy will be discarded when a positive acknowledgement is received from the next node. See discussions on error control in Chapter 2.)

A network operating in this fashion is called a ***store-and-forward*** (SF) network. There are two subcategories: message-switching (MS) networks and packet-switching (PS) networks. The distinction is made by considering what happens to the blocks of data bits that are exchanged between communicating processes residing in user nodes. Such a data block together with any necessary control information will be referred to as a *data message*. A network is said to be a message-switching network if data messages constitute the data units that travel through the network in a store-and-forward manner. A packet-switching network is one in which the data units that travel through the network have a maximum size, so that data blocks to be exchanged between communicating user processes may have to be split into smaller segments; these segments together with any necessary control information, referred to as *data packets*, constitute the data units that traverse the network in a store-and-forward manner. The maximum size of packets ranges from a few hundred bits to about 10,000 bits in most operational networks. The splitting may be done either in the source user node or by the network. (It may be done in the network layer or some

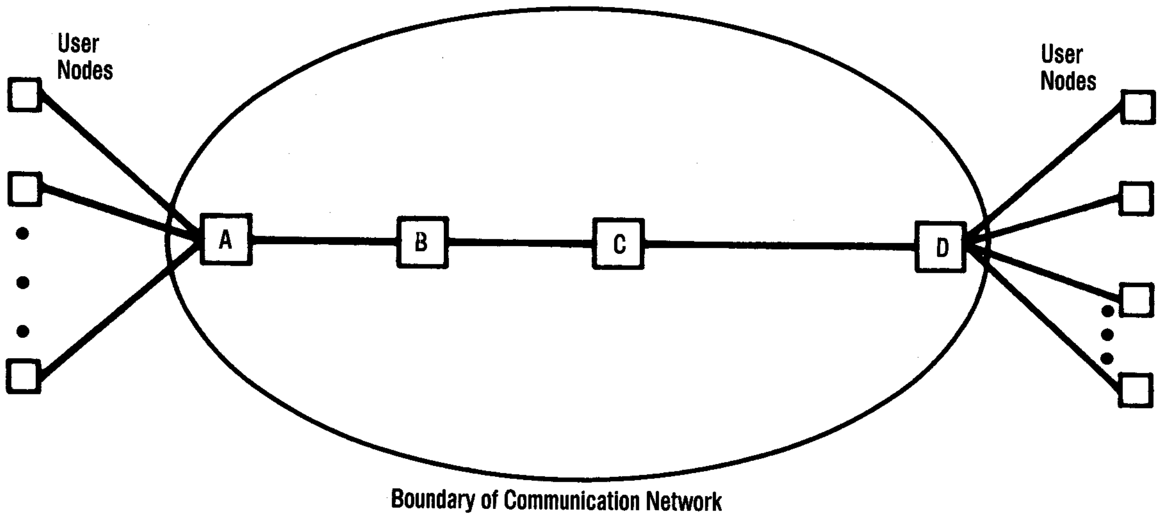


Figure 1-12. Switching nodes in a path between communicating user nodes.

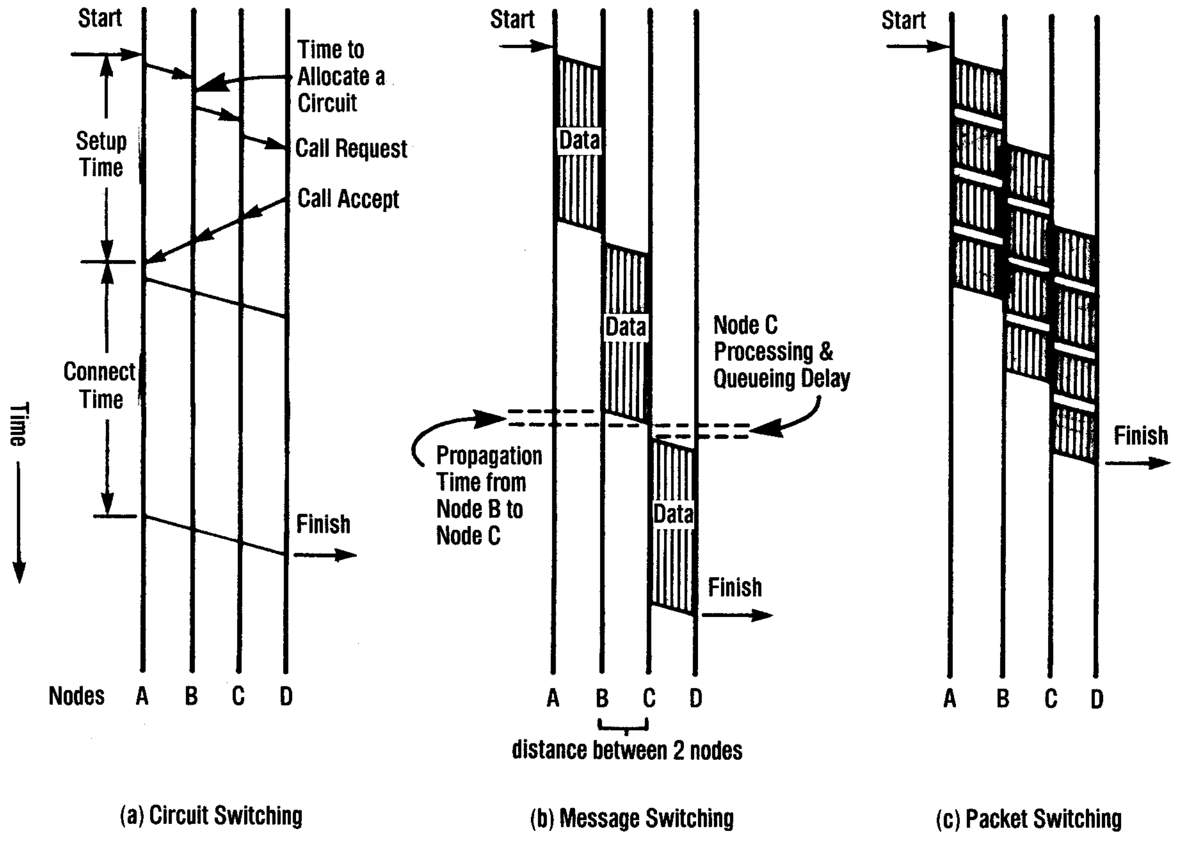


Figure 1-13. Timing of events in circuit, message, and packet switching.

protocol layer above it. Protocol layers will be addressed in Section 1.5.)

Note that both the end-to-end throughput available from a circuit-switched connection and its delay are constant for all practical purposes. However, logical connections offered by SF networks are characterized by time variations in both throughput and delay as the network load fluctuates. In general, communication channels are utilized more efficiently in an SF network than in a circuit-switching network if network users generate bursty traffic.

The switching nodes in an SF network must have significant processing capabilities and buffering capacities. As a result, speed and code conversion as well as a variety of other functions can be incorporated into an SF network to improve the quality and characteristics of its logical connections.

Packet switching vs. message switching

Examples of message-switching networks include the Telex network, the AUTODIN I network, and the SITA airline system. Most of the more recently developed networks and network architectures however are based upon packet switching, e.g., ARPANET, Telenet, Transpac and IBM's Systems Network Architecture (SNA), to name just a few. Packet switching offers many advantages over message switching but it also requires the network to take on additional responsibilities (functions). The following are the advantages:

1. Packet switching reduces the end-to-end delay required to deliver a data message from one user node to another user node. Three factors contribute to a reduction in delay. First, when several data packets are sent instead of

a single, large data message, the end-to-end delay can be reduced due to the effect of pipelining. Compare Figure 1-13(b) and Figure 1-13(c) and recall that in an SF network, each switching node must have received a data unit completely before forwarding it to the next node. Second, packets constituting the same message may travel in parallel paths to the destination. Third, according to queueing theory, a reduction in the variance of data unit sizes generally gives rise to a smaller average queueing delay.

2. Data packets having a bounded maximum size facilitate the management of buffers in switching nodes.
3. For a fixed number of error detection bits, the probability of undetected errors is greater for data messages than it is for the smaller data packets.

To implement packet switching instead of message switching, additional functional capabilities are required either of the user nodes or of network switching nodes: (1) the splitting of a data block to form data packets, and (2) the reassembling of a data block from its constituent data packets before delivery to the destination user process.

Header switching

In any node within an SF network, each data unit (message or packet) must be received in its entirety and checked to be error-free before it is forwarded to the next node on its path. Figures 1-13(b) and 1-13(c) show that pipelining reduces end-to-end delays when long data blocks are transported in smaller data packets. The degree of pipelining can be further increased if a data unit can be forwarded as soon as its control information header is received and the outgoing channel becomes available. Such a method was proposed and studied by Kermani and Kleinrock [KERM 79] and was referred to as "virtual cut-through." We have adopted the term "header switching" coined by this author [LAM 80] because it is more suggestive of what goes on and in the same vein as packet switching and message switching (Figure 1-14).

Each data unit typically consists of two parts: a control header followed by a data field. The header contains the data unit's identification (source-destination addresses or logical channel number) so that receipt of the header at an intermediate node is sufficient to determine the outgoing channel for the data unit. To implement header switching, the header should be protected by an error-detection code and checked at each intermediate node. On the other hand, no error checking is performed for the data field of a data unit. Hence there is no error control for data across each communication link. This is a price to pay for the increased pipelining. Error control is thus the responsibility of the entry and exit nodes of the network or of the source-destination user nodes. Note that the error rate of the entire path is the cumulative raw error rate of the links in the path. Thus header switching is appropriate for networks in which communication links have very low error rates and/or paths are not too long (in number of links). The use of header switching was suggested by this author for the environment of local area networks [LAM 80].

It is interesting to note that pipelining is

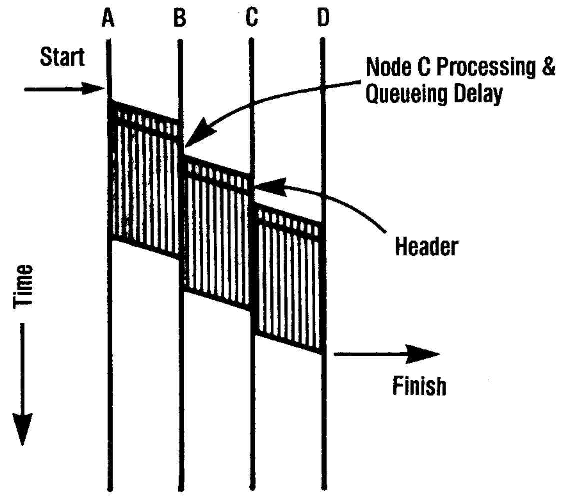


Figure 1-14. Timing of events in header switching.

achieved to an even greater extent in ring (or loop) networks than it is in header switching. In a ring topology, intermediate nodes need not make routing decisions. Also, error checking of the header of a data unit is usually not done in ring networks, assuming an environment of very low link error rates. Hence, in ring networks, each bit can be repeated downstream by a node as soon as the bit is received from upstream.

1.4.3. Sharing a Broadcast Channel

A broadcast channel is one that interconnects a population of geographically distributed users such that any user can transmit into the channel and any transmission in the channel can be received by all users. Broadcast channels can be realized using a variety of communication media: coaxial cable, radio, and satellite, to name the obvious. Techniques for sharing use of a broadcast channel are called *multiple access* techniques. They fall into two categories.

Channel-switching techniques

As in the sharing of a point-to-point communication channel, FDM or TDM can be used to multiplex the channel into subchannels. For a broadcast channel, these techniques take on the new names of frequency division multiple access (FDMA) and time division multiple access (TDMA), respectively. As before, a fixed assignment or a demand assignment strategy can be employed. Due to the broadcast nature of the transmission, the same subchannel can be used by a sender for all of its intended receivers. (With point-to-point channels, each sender-receiver pair requires the assignment of a subchannel.) In this case, transmissions must be identified with destination addresses and potential receivers of transmissions from a particular sender must monitor all transmissions in the subchannel allocated to that sender.

With FDMA, multiple modems that operate at different frequencies are needed for the different subchannels. In the sharing of a point-to-point communication link (Figure 1-8), all of these modems can be located in the switching node and shared by traffic from all user nodes. But with a broadcast channel and geographically distributed users, FDMA techniques would require either the deployment of multiple modems or a modem that is "frequency agile" for each user.

Packet-switching techniques

Data packets from geographically distributed users who access the same broadcast channel cannot be lined up to form a queue as in the sharing of a point-to-point link. The allocation of bandwidth on a per-packet basis will have to be achieved by two strategies: (1) by allowing conflicts to occur and employing a distributed algorithm to resolve conflicts and (2) by devising a protocol to coordinate the users so that a logical queue, one that is physically distributed, is formed. Packet switching techniques for the multiple access of a broadcast channel constitute the subject of Chapter 3 in this text.

1.4.4. Hybrid Networks

A network is sometimes described as "hybrid" because it handles a variety of traffic types (computer data, voice, facsimile, video etc.) or because it incorporates a mixture of switching techniques (packet switching, circuit switching, etc.) or because it employs a mixture of communication media (point-to-point circuits, satellites etc.). Quite often, to handle different traffic types in a network implies that it is desirable for the network to have multiple switching techniques and different communication media. Conventional wisdom seems to indicate that circuit switching is suitable for voice traffic and nonbursty data traffic, whereas packet switching is suitable for interactive computer applications that typically generate bursty traffic. The use of packet switching for digital voice traffic has been of interest to some researchers in recent years [FORG 77, GONS 83]. For the transmission of very large data and facsimile files as well as video, a satellite channel may be needed to provide the large bandwidths that are difficult to come by in terrestrial networks over long distances.

Integrated circuit and packet switching

Circuit and packet switching can be integrated in the sharing of a channel, point-to-point or broadcast, by partitioning the channel into two subchannels: one for circuit switching and one for packet switching. The subchannel allocated to circuit switching is further divided into smaller subchannels. The packet-switching subchannel is allocated on a per-packet basis, as described before.

A novel feature that can be incorporated into an integrated circuit and packet switching system is to make the partition between the CS and PS subchannels dynamically movable. Specifically, it means that any currently unallocated CS subchannel capacity is used to transmit packets waiting for the PS subchannel. Integrated CS/PS with a movable partition is conceptually feasible whether the multiplexing is done by frequency or time division. However, the movable-partition strategy is easiest to illustrate (and probably easiest to implement) assuming the use of TDM. We illustrate the integration of CS and PS based upon TDM in Figure 1-15. As in Figure 1-10 (illustrating a TDM channel), time is divided into consecutive frames. Each frame is partitioned into two subframes, one for CS traffic and one for PS traffic. The CS subframe is further divided into time slots. The number of CS subchannels available for fixed or demand assignment is equal to the maximum number of time slots in a CS subframe.

We have shown two slightly different versions of a dynamically movable partition in Figures 1-16(a) and 1-16(b), depending upon whether a slotted system is used for PS traffic or not. In Figure 1-16(a), a slotted system is used for PS traffic. In this case, the entire frame is slotted. End-to-end circuits are allocated time slots designated by their positions within the CS subframe. Any unallocated time slots within the CS subframe can be used for the transmission of waiting PS traffic.

Figure 1-16(b) illustrates an unslotted system for PS traffic. In such a system, the PS subframe is not slotted. Furthermore, CS traffic is "packed" at the beginning of the CS subframe, so that the remaining time in the frame is available to PS traffic in a contiguous piece. As a result, variable-length packets or messages can be packed inside it efficiently. The price to pay is the need to identify each block of CS transmission (with, say, a logical channel number that identifies the end-to-end circuit).

1.5. ARCHITECTURAL MODELS

Suppose we have a network of communication channels and switching nodes. The collection of programs (in hardware or software) to implement functions to achieve communication between processes in user nodes is extremely complex. The design and development of these parallel programs require adherence to sound structured programming principles. We shall begin by enunciating

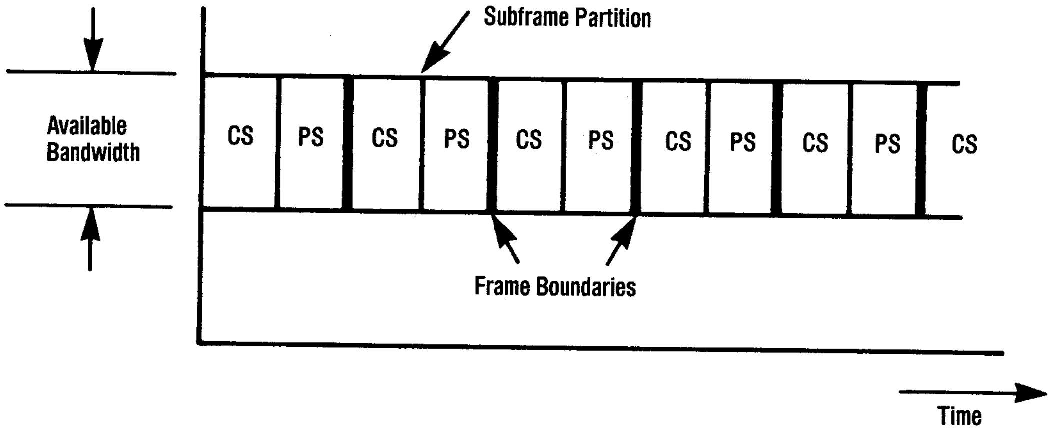


Figure 1-15. Integrated CS/PS based upon TDM.

three principles that we consider to be fundamental to a well-structured architectural model. This is the subject of Section 1.5.1. Some design and implementation issues are discussed in Section 1.5.2. Examples of architectural models are shown in Section 1.5.3 for both networking and internetworking (interconnection of autonomous networks).

1.5.1. Three Structuring Principles

Consider a protocol system in which an abstract "communication process" provides the ability for a population of user processes to send data to each other (Figure 1-17(a)). A line joining two processes in our figures is used to denote a two-way communication path. The path is internal to a machine if the processes reside in the same machine; it represents a communication link if the processes reside in different machines. Two processes connected by a line in our illustrations of architectural models are said to be *adjacent*. For the following discussions, a process is either a sequential process or a concurrent process. A sequential process is resident in a single machine. A concurrent process consists of several subprocesses resident in the same machine or different machines; each subprocess may in turn be either a sequential or concurrent process.

Stepwise refinement

We consider architectural models that can be obtained by a stepwise refinement procedure, as illustrated by the example shown in Figure 1-17. In performing stepwise refinement, a process with multiple functions in the current version of an architectural model is replaced by a set of interconnected processes (any topology) that provides the same functions. The communication process 1 in Figure 1-17(a) has multiple functions including the establishment and management of logical connections between pairs of user processes. This function is peeled off for implementation in separate processes which are labelled 1.1 in Figure 1-17(b). (Identical processes are labelled with the same number.) Next, the communication process 1.2 is refined to reveal its

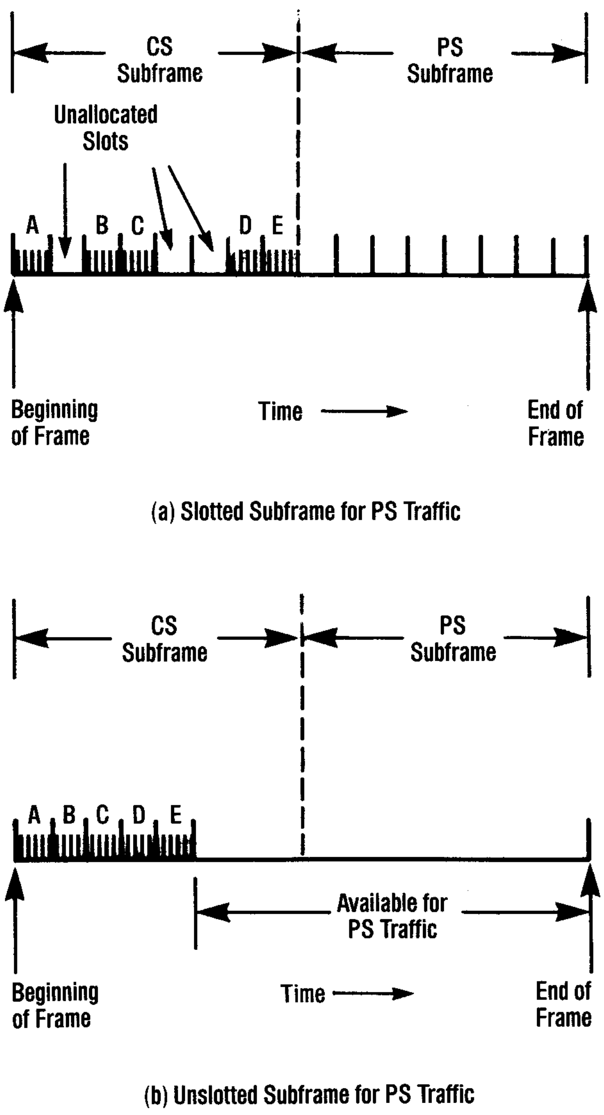


Figure 1-16. Two variants of an integrated CS/PS system.

internal structure, which is a set of geographically distributed processes interconnected by point-to-point communication links (Figure 1-17(c)).

We now make the observation that all of the 1.1 and 1.3 processes in Figure 1-17(c) are interconnected by physical links. Thus each must have the data link management and error control functions for every one of its physical links. Each such process is then split into two or more processes, one for the network management functions and the other(s) for the data link functions (one process for each separate data link). Thus, we ar-

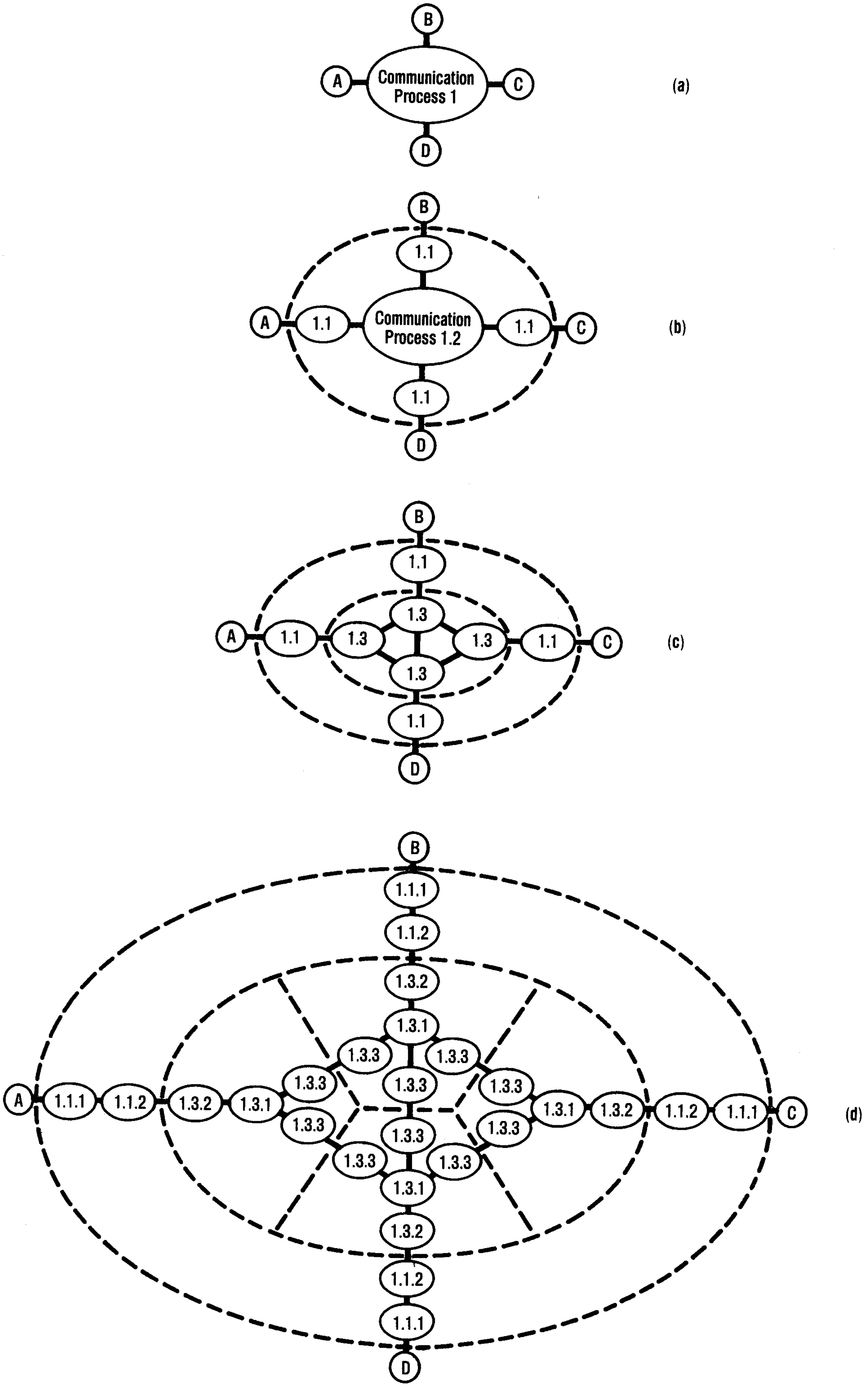


Figure 1-17. An illustration of stepwise refinement.

rive at the architectural model shown in Figure 1-17(d).

Hierarchy

At the time that new processes are created by stepwise refinement, they are classified into different levels of a process hierarchy. The user processes are at the top level of the process hierarchy. New processes created by stepwise refinement may reside at the same level or at one of the lower levels. The processes in Figure 1-17 are hierarchically ordered in Figures 1-18(a) to 1-18(d). To avoid drawing three-dimensional pictures, we have shown only the processes along a single path between user process A and user process C in Figure 1-17. We have further partitioned each data link process in Figure 1-18(d) into two processes, one for the physical data link and one for the logical data link functions, as shown in Figure 1-18(e). The graph in Figure 1-18(e) is also partitioned vertically to show the physical location of processes in different nodes. There are two types of nodes: user (or host) nodes and network switching nodes. Although not shown in Figure 1-18, two adjacent processes residing in the same node can be more than one level apart, i.e., one or more levels can be bypassed by a direct communication path.

Interprocess communication restriction: Two processes in the process hierarchy can communicate only if either they are within the same node and are adjacent to each other or they are at the same level of the process hierarchy.

The interaction between two processes residing in the same node can be handled by any of the interprocess communication techniques for a multiprogramming environment [BRIN 73]. Two processes at the same level of the process hierarchy can interact via the exchange of messages. If the processes are not adjacent to each other, messages are relayed by intermediate processes along a path between them. A routing protocol needs to be implemented when more than one path exists.

Most protocols of interest in the literature are concerned with communications among a subset of processes at the same hierarchical level but located in different nodes. Such communications are known as peer-to-peer communications and such protocols are referred to as *peer protocols*. The processes at each level of the process hierarchy together with their peer protocols are said to constitute a *protocol layer* of the architectural model. The simplified picture in Figure 1-19 is often shown in the literature instead of the process hierarchy in Figure 1-18(e). Conceptually, processes in each layer of the hierarchy (except layer 1) are provided with some communication services by the immediately next lower layer; these processes contribute some additional functions of their own and provide improved communication services to processes in the next higher layer of the hierarchy.

Peer protocols are defined for subsets of processes in each layer. Each peer protocol can provide a communication service to other processes in the same layer as well as to adjacent processes in a higher layer. (A process is adjacent to a subset of processes if it is adjacent to any process within the subset.) We shall refer to processes that use the communication service provided by a peer protocol as the *protocol users*. Several examples can be found in Figure 1-18(d). A data link protocol can be defined for processes 1.1.2 and 1.3.2 to provide a communication service to processes 1.1.1 and 1.3.1 in the next higher layer. A networking protocol can be defined for all the 1.3.1 processes in the subnetwork of switching nodes to provide a communication service to all the 1.1.1 processes in the same layer and located in user nodes. A network access protocol can be defined for each pair of 1.1.1 and 1.3.1 processes. The entire network layer also provides a communication service to user processes.

Data transparency

The issue of data transparency at the interface between a protocol and its users needs to be considered for both send operations by a protocol user and receive operations by a protocol process. We define *send transparency* to mean that a protocol user can give any sequence of bits to the protocol for delivery to another user. We define

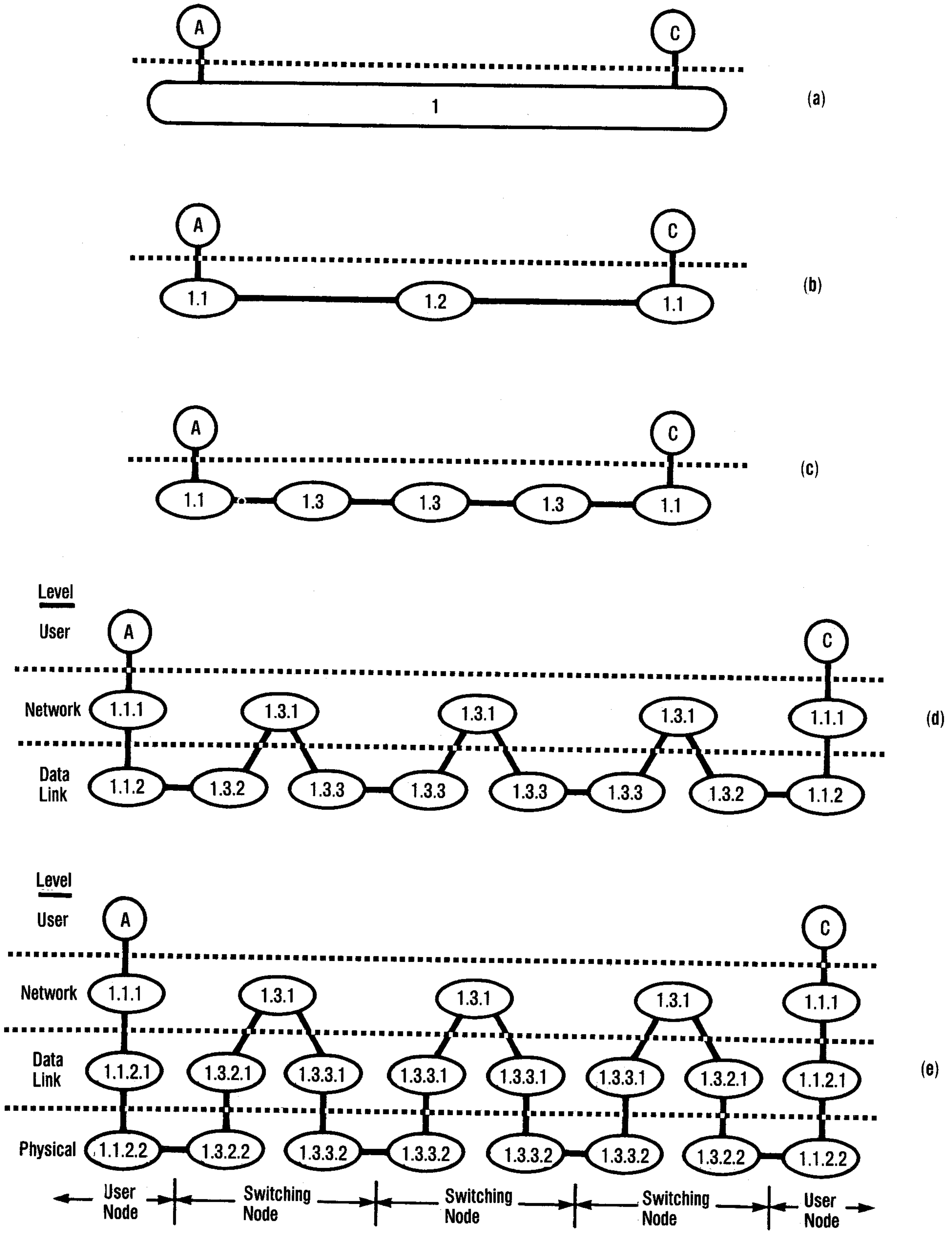


Figure 1-18. An illustration of hierarchy.

receive transparency to mean that any sequence of bits given by a protocol user to the protocol for delivery need not be interpreted by the protocol. Both send and receive transparency across an interface are required for the protocol interface to be called a ***transparent interface***. The arbitrary bit sequences that are sent across a transparent protocol interface are referred to as ***transparent data***. Typically, data transparency cannot be achieved at a protocol interface because of restrictions imposed by communication services utilized by the protocol itself. In this case, whether send transparency or receive transparency is affected depends upon whether the protocol processes or the protocol's users undertake the responsibility for satisfying those restrictions.

Data transparency is usually enforced in the design of all protocols in an architectural model, with the exception of the bottom layer(s) that must deal with the functions of bit synchronization and frame synchronization. In order to achieve bit synchronization (or phase synchronization), the sending node needs to guarantee that the signal, encoding a bit stream, has adequate transitions; if not, some bits may have to be inserted into the bit stream or the bit stream will have to be scrambled to create transitions [DAVE 72]. The receiving node then performs an inverse operation of deleting inserted bits or unscrambling to restore the original bit stream. To achieve frame synchronization, it is necessary to examine the stream of data bits and to insert/delete bits so that the "marker" of frame boundaries, in the form of a special sequence of bits, can be made unique. (Such frame synchronization techniques are described in Chapter 2.)

In practice, the operations of bit insertion/deletion and scrambling/unscrambling are typically done in hardware. Given such hardware, protocols of interest in this text (at the data link level and above) will be considered to have transparent interfaces.

1.5.2. Implementation and Design Issues

Data units given to a peer protocol for delivery will be referred to as the protocol's ***service data units*** (SDUs). A service data unit contains control information of upper layer protocols and possibly some actual data from an end-user process. To have a transparent interface, the protocol must regard the entire SDU as "pure data." The method of ***encapsulation*** is generally used to implement a transparent interface. In this method each SDU received from a protocol user is wrapped with a header and a trailer. (A trailer is necessary only in a data link protocol for marking the end of a frame.) We refer to a wrapped SDU as a ***protocol data unit*** (PDU). If the protocol utilizes a communication service offered by a protocol in a lower layer, the PDU is then given to the lower layer protocol which will in turn regard it as "pure data." The header-trailer of a

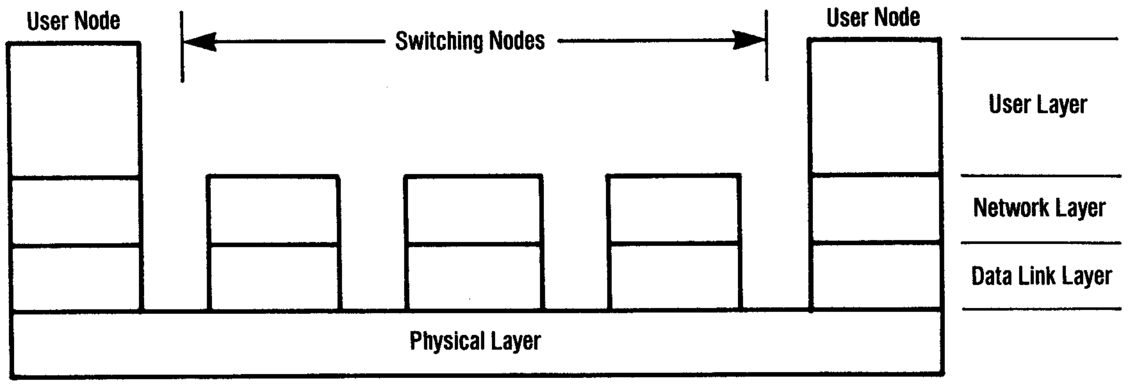


Figure 1-19. An illustration of protocol layers.

PDU provides the vehicle for processes that participate in this protocol for sending control messages to each other. Furthermore, if a process wants to send a control message, it can generate a PDU consisting of a header-trailer only, with nothing wrapped inside.

The header-trailer of a PDU is stripped away before the encapsulated SDU is delivered to an external process (Figure 1-20). Note that the interprocess communication restriction in the previous section dictates that external processes that utilize this protocol for the exchange of SDUs must reside in the same layer. These user processes can be at the same level as the protocol processes or at a higher level in the process hierarchy.

Figure 1-21 is an illustration of the series of wrappings and unwrappings that occur when a piece of data travels from user process A to user process C, as shown in Figure 1-18(d). Network-access protocols are defined between pairs of 1.1.1 and 1.3.1 processes. A network protocol is defined for all the 1.3.1 processes. Data link protocols are defined for pairs of 1.1.2 and 1.3.2 processes and for pairs of 1.3.3 processes.

In Figure 1-21, network access protocol headers are identified by the number 3. Network protocol headers are identified by 3'. Data link protocol header-trailers are identified by the number 2.

Note that the 1.3.1 processes participate not only in the network protocol but also in the network-access protocols. Each SDU for the network protocol is wrapped in the header of a network-access protocol when it arrives. For each data unit received, a 1.3.1 process unwraps the network-access protocol header, delivers the SDU to itself, and then rewraps it with a network protocol header.

The method of encapsulation implicitly assumes the *positional significance* approach for separating control and data information. A special bit pattern serves as a marker of frame boundaries. (A frame is a data unit transmitted across a physical link.) Data and control information are physically separated into different fields according to offsets from the beginning and end of each frame. A process participating in a protocol receives protocol data units either directly from another protocol process or indirectly from a lower layer protocol. Because of the interprocess communication restriction stated in Section 1.5.1, the unwrapping of the protocol's header-trailer is always performed at the beginning and the end of the data unit received (by simply peeling away fixed numbers of bits from the beginning and the end of the data unit received.)

Note that several peer protocols are usually present within a protocol layer, and a process can participate in more than one such protocol. As a

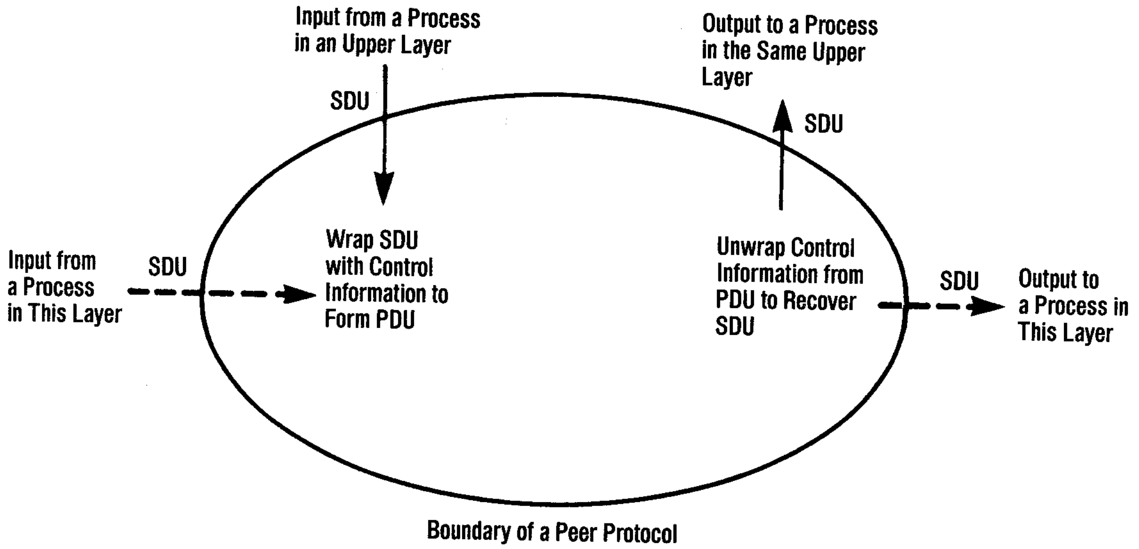

Input from a Process in an Upper Layer
SDU
Output to a Process in the Same Upper Layer
SDU
Input from a Process in This Layer
SDU
Wrap SDU with Control Information to Form PDU
Unwrap Control Information from PDU to Recover SDU
SDU
Output to a Process in This Layer
Boundary of a Peer Protocol

Figure 1-20. Wrapping and unwrapping in a peer protocol.

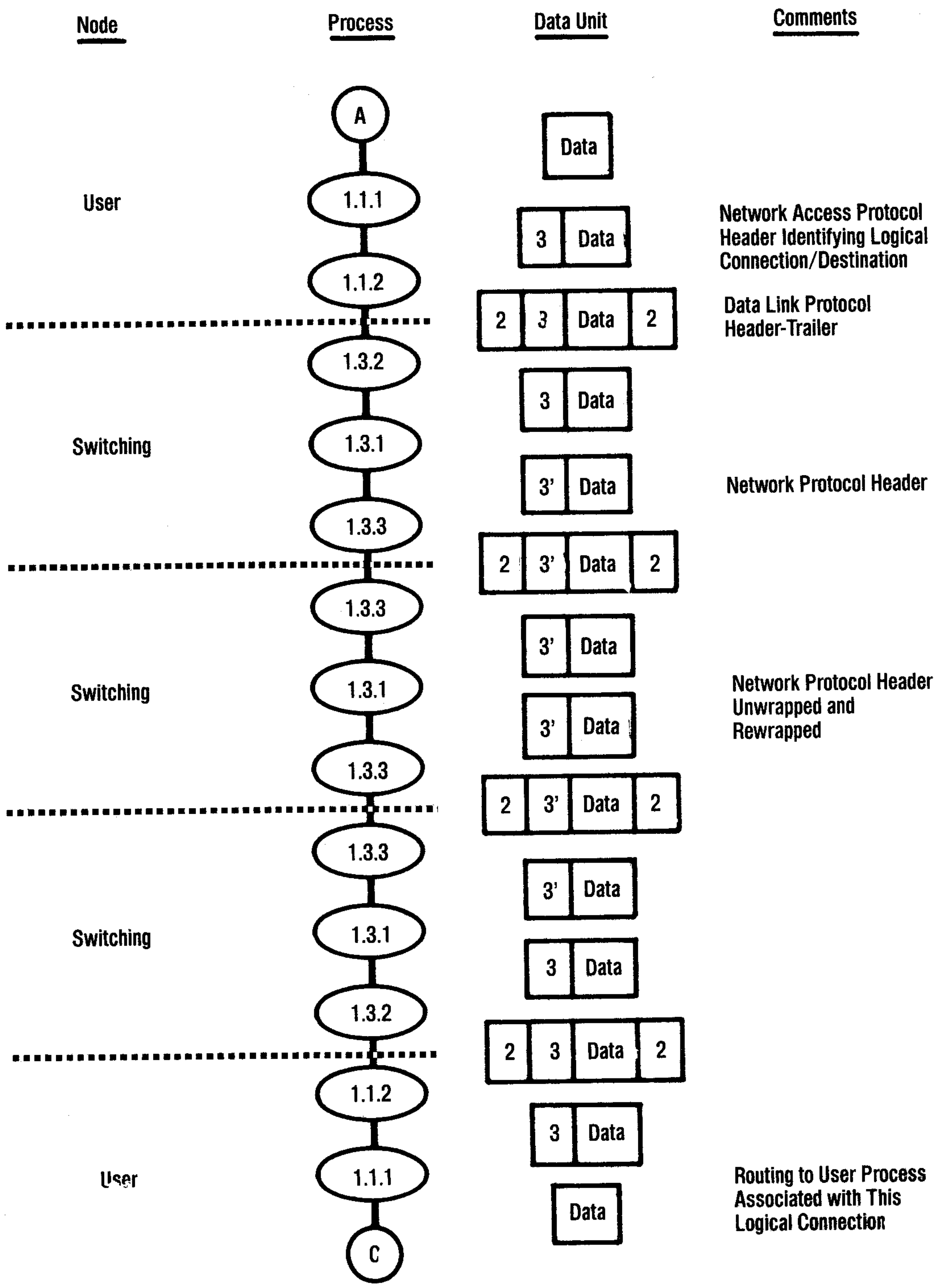

Node
Process
Data Unit
Comments
A
User
1.1.1
1.1.2
Data
3 Data
Network Access Protocol Header Identifying Logical Connection/Destination
2 3 Data 2
Data Link Protocol Header-Trailer
Switching
1.3.2
1.3.1
1.3.3
3' Data
Network Protocol Header
2 3' Data 2
Network Protocol Header Unwrapped and Rewrapped
User
C
Routing to User Process Associated with This Logical Connection

Figure 1-21. Wrappings and unwrappings along an end-to-end path.

result, data units and their header-trailers will sometimes have to be identified by both the layer to which they belong and the specific protocol within the layer for which they are used.

The properties of the communication service provided by a peer protocol depend upon the functions implemented by the protocol as well as the properties of the communication services utilized by the protocol processes. Two common types of services are described below.

1. *Datagram service:* Service data units are called datagrams. Each datagram is treated independently by the protocol. The protocol does not provide end-to-end error control. Losses may occur. (However, losses may be minimal if error-control functions are implemented in lower layer protocols.) Out-of-sequence delivery may occur; i.e., the protocol may deliver datagrams in a different order from the one in which they are given to the protocol (due to retransmissions or the use of parallel paths in a network).
2. *Virtual circuit service:* The protocol provides end-to-end error control as well as in-sequence delivery of service data units. Thus the communication service provided is a reliable FIFO logical pipe.

The provision of a datagram or virtual circuit service, or something in between, by protocols in different layers of a communications architecture is a design decision rather than an architectural consideration. Such decisions are concerned with the partitioning and placement of networking functions in a fashion that is optimal for a particular network environment.

There are several other design and implementation issues that impact the communication service provided by a peer protocol, regardless of whether it provides a datagram or virtual circuit service, namely: input control, maximum data unit sizes, segmenting, and blocking. We address these issues next.

Input control

The communication service provided by a protocol may not be available at all times for input. Controlling the input of service data units is necessary to deal with temporary overloads for protocols that implement statistical multiplexing of communication facilities. Input control of a source user is also necessary if the corresponding destination user is not accepting delivery of service data units from the protocol fast enough [GERL 80].

Maximum data unit sizes

For implementation purposes, a maximum size needs to be specified for the protocol data units of each protocol in an architectural model. The determination of the protocol data unit size is based upon some fundamental design and implementation considerations such as the size of buffers in switching nodes, the maximum number of data bits for which an error-detection code will work reliably, the maximum size of a data unit for which an error-control scheme is efficient (the probability of having errors in a block increases with the block length), and the reduction of end-to-end delay because of pipelining in a store-and-forward network (as discussed earlier). The maximum size of protocol data units may vary among the different protocols in the same architecture. Also, the maximum size of the service data units and protocol data units of the same protocol may differ. Such differences in an architectural model are handled by incorporating the functions of segmenting and blocking in its implementation.

Segmenting

A peer protocol providing a communication service can have a segmenting function such that a service data unit given to the protocol for delivery may be split into several segments (Figure 1-22). Each segment is wrapped individually with the protocol header-trailer to form an individual protocol data unit. When a service data unit is segmented, each segment is given a sequence number. The protocol data unit header includes the sequence number of the segment that it wraps as well as an indication of whether the segment is the last in a sequence.

The protocol may or may not reassemble segments before their delivery to a destination user. If a protocol implements a segmenting function but not a reassembly function, then it provides a communication service that handles transparent data although it may fragment units of such data. Note that these fragments of transparent data cannot be reassembled by another protocol in the same layer or in a higher layer. (The fragments have no sequence number information.)

Protocols providing either a datagram or virtual circuit service can implement segmenting and reassembly functions. Segmenting without reassembly, however, seems inconsistent with the objective of a virtual circuit service.

There are several reasons for a protocol to implement segmenting. An obvious reason is that the maximum protocol data unit size is smaller than the maximum service data unit size (or the service data unit size of a lower layer protocol utilized by it is too small).

Blocking

A peer protocol may also combine several of its protocol data units, each of which encapsulates a separate service data unit, into a single block. If given to another protocol for delivery, the entire block constitutes a single service data unit of the other protocol. This is known as blocking (Figure 1-23). One reason for blocking is to improve the efficiency of error control in a data link protocol [CHU 74]. Blocking is seldom implemented in practice.

When a protocol combines several of its protocol data units into a block, then the header of each protocol data unit must contain a count of its length, so that separate protocol data units can later be recovered from the block prior to delivery of their contents to destination users. Instead of a count, the trailer of each protocol data unit may contain a unique marker which serves to separate different protocol units within the block. However, to implement unique markers, the principle of data transparency may have to be compromised. Of course, if the protocol in question is a data link protocol, a unique marker has already been implemented for the purpose of frame synchronization.

Relationships between types of data units

Note that the wrapping of transparent data with a protocol's header-trailer must be done *after* segmenting, but *before* blocking, in order for each piece of transparent data to be uniquely identified. The following relationships exist between protocol data units and service data units in an architectural model. For a given protocol, a protocol data unit contains either an entire service data unit, a segment of a service data unit, or no service data unit. A service data unit of a protocol consists of one or *more* of the protocol data units of some protocol that utilizes this protocol.

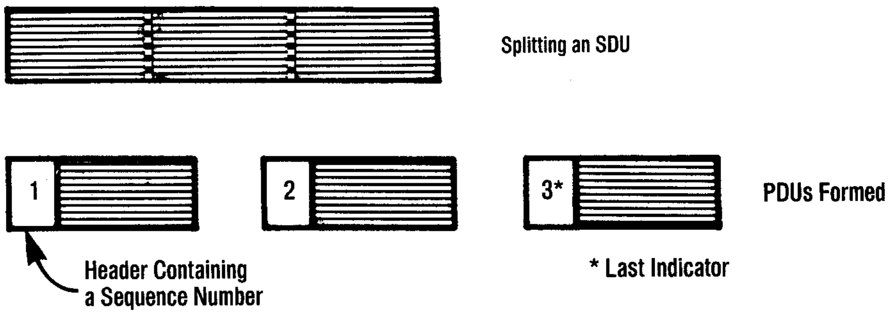


Figure 1-22. Segmenting an SDU to form multiple PDUs.

Let us reconcile our use of the terms protocol data units and service data units with the terms data packet and data message as used earlier in Section 1.4 in conjunction with our discussions of switching techniques. The conceptual message-switching network described in Section 1.4 had no segmenting or blocking function in any layer. *Data message* was a general term referring to any data unit in any layer. The conceptual packet-switching network described in Section 1.4 had no blocking function in any layer but had a segmenting function in one layer. Segmenting was done once in the network layer (or a higher layer) of a user node or in the network layer of a switching node, and the term *data packet* referred generally to data units in any layer after segmentation had occurred.

The term *frame* was used to refer to the protocol data units of a data link protocol. The term *datagram* was used to refer to the service data units of a peer protocol (typically a network layer protocol) that provides a datagram service.

Transparency and service characteristics

For a protocol with a limitation on the size of its service data units, one might argue that the protocol's interface is not transparent in an absolute sense. A protocol user does not need to examine the contents of a service data unit, but the user does have to determine its length and break it up if necessary.

For a protocol with possible delivery of fragmented service data units, the principle of data transparency as stated earlier is not violated. But fragmentation, together with the service characteristics of loss and reordering of service data units, is obviously undesirable for an end-to-end communication service. However, they may be acceptable characteristics of communication services provided by lower layer protocols in a communications architecture.

1.5.3. Models for Networking and Internetworking

Practically, every network that has been built has its own architectural model. There are numerous possibilities in the choice of communication facilities and switching techniques. The quality, characteristics, and type of communication services desired by user processes determine the network functions required, and there are numerous ways of allocating these functions to different protocol layers and dividing them between user nodes and switching nodes. Many of the successful networks in existence however, seem to conform to a great extent the three structuring principles described in Section 1.5.1. For examples, protocol layers in Ethernet, ARPANET and TYMNET architectures are shown in [CERF 78]. Protocol layers in SNA are shown in [CYPS 78]. A comparison of the functions in SNA and the Open Systems Interconnection (OSI) protocol layers is given in [RUTL 82].

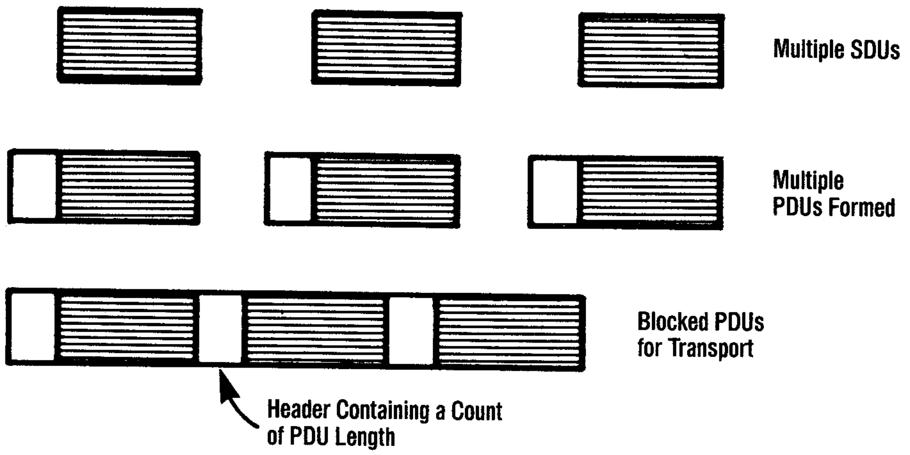

Multiple SDUs
Multiple PDUs Formed
Blocked PDUs for Transport
Header Containing a Count of PDU Length

Figure 1-23. Blocking of PDUs.

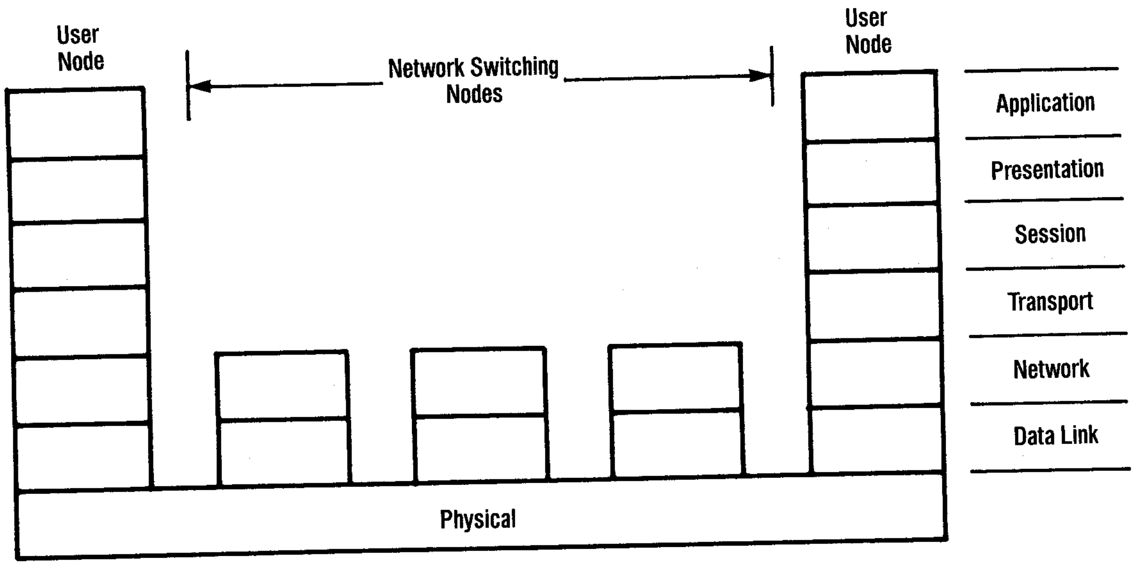


Figure 1-24. The seven layers of the OSI model.

The OSI reference model

The International Organization for Standardization (ISO) has been developing a reference model to guide standards development in networking. This model is known as the basic reference model for Open Systems Interconnection (OSI). The OSI model partitions the communication and networking functions into seven layers, as shown in Figure 1-24. Again, to avoid drawing a three-dimensional picture, we illustrate the OSI layers for a communication path consisting of three switching nodes between two user nodes (the same as Figures 1-18 and 1-19).

A brief description of the main functions of the OSI layers follows. The reader is referred to [OSI 81] for details. Note that not all functions allocated to a layer must be included in an implementation.

A *physical layer* protocol provides the function of bit synchronization, i.e., the delivery of a stream of bits between processes that utilize the physical layer protocol.

A *data link layer* protocol provides the functions of frame synchronization, and error control, as well as the sharing and management of a physical communication link.

Network layer protocols provide the ability for user nodes that are not connected by a dedicated physical link to send data units to each other. Specific network layer protocols depend upon the network topology, communication facilities, and switching techniques. Functions of routing, flow control, and congestion control are included in this layer for networks of point-to-point links. The function of multiple access is the responsibility of this layer for nodes interconnected by a broadcast channel. Protocols for network access by user nodes are also included in this layer.

A *transport layer* protocol provides the transportation of data units between processes residing in different user nodes. It is concerned with the establishment and management of end-to-end logical connections between user nodes as well as with improving the quality and characteristics of the communication service provided by the network layer (e.g., functions of end-to-end error control, in-sequence delivery, and reassembly).

A *session layer* protocol provides the means for binding two presentation layer processes into a relationship and then unbinding them, for synchronizing their dialog, and for managing their data exchange. (Note that many sessions may be multiplexed onto a transport layer connection.)

A *presentation layer* protocol represents information to communicating application processes in such a way that preserves meaning while resolving syntax differences (e.g., different codes, formats, and data compression techniques).

Application layer protocols provide services to the external users of the OSI environment. User specifiable parameters are made known to the OSI environment via the application layer.

The objective of the OSI model is to provide a framework to guide the development of protocol standards in the future and to foster compatibility in general. Functions described in [OSI 81] and their partitioning into the seven layers obviously do not constitute the only feasible architectural model. In the architectures of several existing networks [CERF 78, CYPS 78] many functions and some layers in the OSI model are missing. Some of the functions that are present are allocated to protocol layers in these architectures differently from the way they are allocated in the OSI model. Nevertheless, it is important to note that the relative positions of most functions within the hierarchy of functions are roughly the same in these architectures as in the OSI model. Of the architectural models shown in [CERF 78, CYPS 78], the SNA model is the only one that also has seven layers and appears to be closest to the OSI model. (There are some significant differences, as discussed in [RUTL 82].) This is probably no accident because both OSI and SNA are relatively new developments compared to the other architectures. Their evolutionary developments during the past few years must have influenced each other to some extent.

Synchronization functions revisited

It is instructional to reexamine the five basic synchronization functions described in Section 1.2 for achieving communication between two parties and to consider the presence or absence of these functions in peer protocols within each layer of the OSI model.

Bit synchronization is provided by a physical layer protocol between any two end points of a communication link (the link may be point-to-point or multipoint). This function is present only in the physical layer.

Frame synchronization is provided by a data link layer protocol between any two end points of a communication link. This function is present only in the data link layer. Methods for frame synchronization are described in Chapter 2.

Note that within the same architecture, different protocols can be specified within the physical layer and within the data link layer for different communication links.

Multiple access synchronization was described in Section 1.2 in the context of sharing a point-to-point communication channel. By expanding its definition to mean the sharing of communication facilities in general (physical or logical), the multiple access function is present in most layers; e.g., it is present in the sharing of a broadcast channel by a data link layer protocol or by a network layer protocol, in the form of routing and flow control functions in the network layer of a store-and-forward network, and in a session layer protocol that multiplexes a logical connection provided by a transport layer protocol into multiple logical connections.

The *content synchronization* function concerned with the separation and encoding of data and control information in protocol data units is present in protocols within all layers. The method of encapsulation to achieve transparency, as described in Section 1.5.2, implicitly assumes a positional significance approach for separating data and control information. Content synchronization also encompasses the functions of error control and in-sequence delivery. These functions may be present in protocols within all layers; e.g., a data link layer protocol at the two end points of a physical link, a network layer protocol at the two end points of a network path, or a transport layer protocol between two user nodes. In practice, the error-control function is most important in the data link layer where error rates are high. The in-sequence delivery function is most important in the transport layer.

Dialog synchronization is present in some form in all protocols within all layers. Note that not all processes within a layer will communicate with each other. The subset of processes within a layer that provides a service must interact so that the necessary logical connections will be made. Such start-up interactions must be an integral part of every protocol. For example, a data link protocol needs to establish and maintain the up/down status of a physical link, a routing protocol needs to establish and maintain the up/down status of all switching nodes that execute the routing algorithm and participate in the storing and forwarding of data units, and a transport protocol needs to establish and maintain the up/down status of an end-to-end logical path.

Models for internetworking

There are numerous difficult technical issues (in addition to political and administrative ones) in the interconnection of separate autonomous networks. The technical issues are discussed in Chapter 6.

The architectural model for the interconnection of networks depends upon the level at which interconnection occurs, the extent to which these networks are compatible with each other, and issues such as whether the interconnection gateways are located in host nodes or in switching nodes (in the form of gateway halves).

Two possible architectural models are shown in Figures 1-25 and 1-26. They appear here primarily as interesting examples of structured architectural models. They give the flavors of two approaches to internetworking, each of which has its advocates. For other approaches and details, the reader is referred to Chapter 6. In both Figures 1-25 and 1-26 we use the names of protocols to identify processes that participate in those protocols. The names A and B refer to protocols of networks A and B, respectively.

Figure 1-25 shows a model for interconnecting networks that have the same interface for network access by a packet-mode user node: in this example, the X.25 interface of the International Telegraph and Telephone Consultative Committee (CCITT). The X.25 interface embodies three protocols between a user node and a network switching node, one each for the network layer,

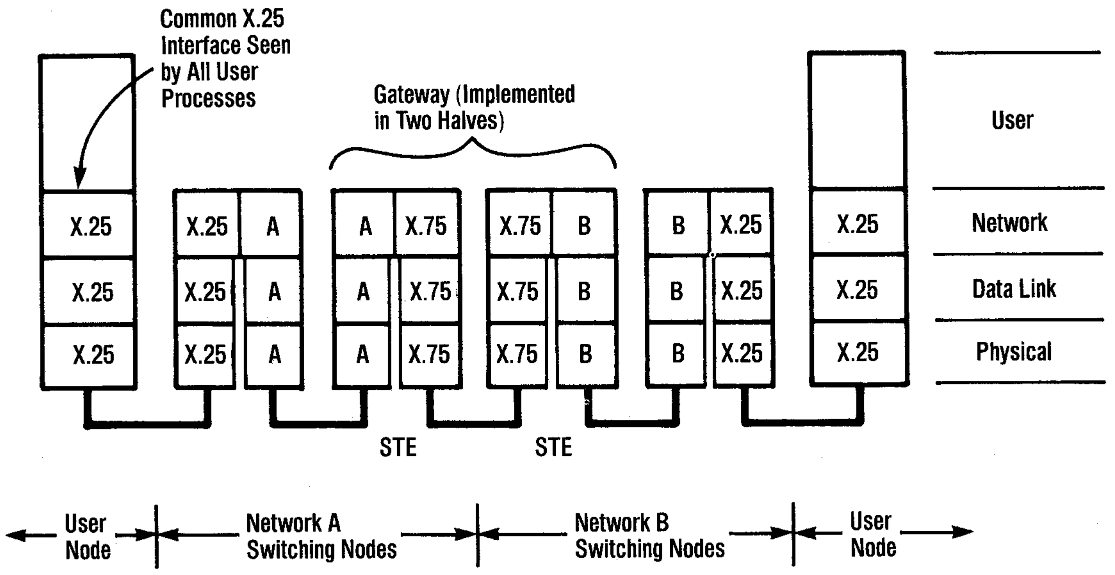


Figure 1-25. An internetworking model with a common network access interface.

the data link layer, and the physical layer. The network layer is referred to as the packet level in X.25. The initial X.25 protocol defined only virtual circuit services. Subsequent revisions have added a datagram service and a fast-select facility for virtual calls [RYBC 80, FOLT 80].

In addition to assuming a common network access interface that offers the same service(s), the model in Figure 1-25 also assumes a common physical address space for all user processes in all the networks interconnected. The interconnection of networks in this model may be accomplished with gateways located in user nodes or gateway halves located in switching nodes. The latter scenario is shown in Figure 1-25. The CCITT Recommendation X.75 specifies the protocols at the physical, data link, and network levels between two switching nodes implementing gateway halves. These switching nodes are called Signalling Terminal Equipments (STEs). The X.75 protocol is compatible with the X.25 protocol for virtual circuit services, but they have additional functions, called network utilities, that handle the signalling of internetwork administrative information. With this approach, an end-to-end virtual circuit is provided as a concatenation of virtual circuits in the networks being interconnected.

CCITT Recommendation X.121 defines an international numbering plan for public network addresses. Each public packet-switching network is assigned a data network identification code (DNIC). A DNIC is a four digit number; the first three digits identify the country and the fourth digit identifies the particular public packet-switching network in that country. For example, Telenet's DNIC is 3110 while Datapac's is 3020. Following the DNIC, an international address contains a network terminal number (NTN) which specifies the address of the terminal within the public data network. The NTN is a one- to ten-digit number and is assigned to a particular terminal by the public data network's administration. Generally, the routing of calls between public data networks is based on DNICs, while the routing of calls within a public data network is based on NTNs.

In summary, referring back to Figure 1-25, we see that protocol layers in user nodes are oblivious to the internals of the networks and gateways (with the exception of a common physical address space for all users in all networks). User processes in different networks can communicate because they see the same X.25 interface and service. The X.75 protocols provide the necessary interconnection functions. An application of this model is described in [UNSO 81] which is reprinted in Chapter 6.

Figure 1-26 shows an internetworking model that employs the method of encapsulation [CERF 78, POUZ 78]. In this model, a gateway intercon-

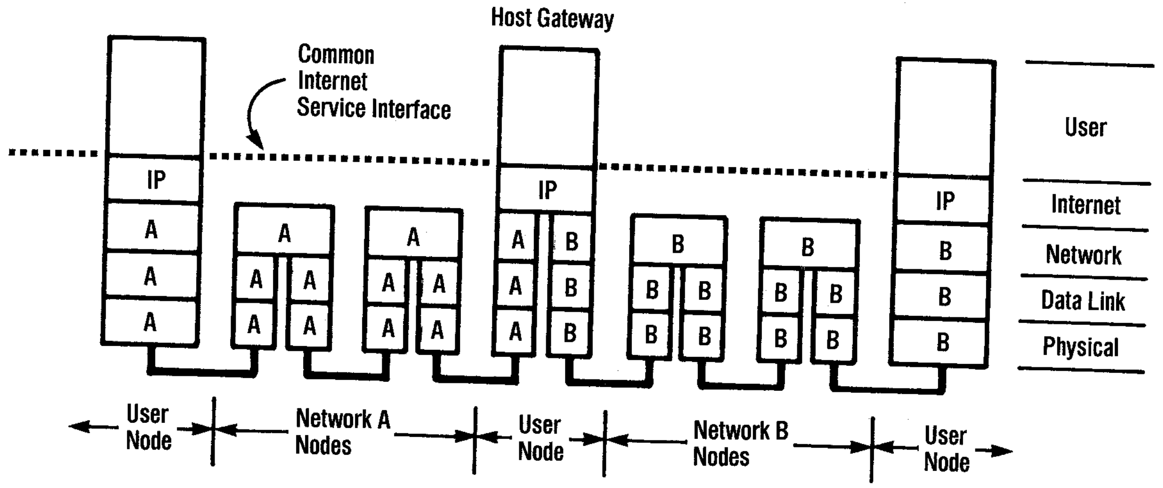


Figure 1-26. An internetworking model based upon encapsulation.

necting two networks resides in a user node that is indistinguishable from any other user node. Such a gateway is called a *host gateway*. The host gateway implements the unique network/data link/physical layer protocols for each network that it connects to. An extra layer is inserted into the process hierarchy (between the network layer and the transport layer in the OSI model). This layer is the same in all user nodes including the host gateways. What we have just described is the internetworking approach of the Defense Advanced Research Projects Agency (DARPA), and the protocol in this extra layer in DARPA's model is called the internet protocol (IP). This layer provides a datagram service to the user processes shown in Figure 1-26. Datagrams given by user processes to the IP layer for delivery are wrapped or encapsulated within the data units of individual network layer protocols. Additional protocols in layers above IP can be implemented to provide communication services having a higher quality and with better characteristics than those of the datagram service of IP.

In summary, referring to Figure 1-26, we see that user processes can communicate because they see the same IP layer interface. The interconnection functions (encapsulation, address translation, routing, segmentation, and reassembly) are provided by the IP layer and its interfaces with the different network layers. The basic advantage of this approach is that almost any sort of network can participate, which is especially useful for connecting networks with new architectures to older networks.

1.6. PROTOCOL STANDARDS

There are some boundaries in an architectural model where standardization of protocols between processes across those boundaries is extremely useful. A most natural boundary is the one that surrounds a packet-switching network. We have already encountered in Section 1.5.3 the X.25 protocol standard at the boundary between a user node, called a DTE (data terminal equipment), and a network switching node, called a DCE (data circuit-terminating equipment). We have also encountered the X.75 protocol standard at the boundary between switching nodes in two different networks, called STEs. DARPA's IP protocol is one that is situated at a different kind of boundary i.e. one that separates the network layer and user layer in Figure 1-26. It is not a protocol standard but it can conceivably become a de facto standard if its popularity grows.

Organizations involved in the development of standards for use in public data networks include CCITT, ISO, ANSI and EIA among others. These are briefly introduced below. The center of public data network standardization activities is Study Group VII of the International Telegraph and Telephone Consultative Committee (CCITT). CCITT is one of two committees of the International Telecommunications Union (ITU), an agency of the United Nations headquartered in Geneva, Switzerland. Over 100 countries make up the membership of the ITU. CCITT deals with technical issues and standards development associated with international telecommunications. Study Group VII is responsible for data communications over public data networks. Its work is contained in the X-series recommendations. Study Group XVII is responsible for data communications over telephone facilities. Its work is contained in the V-series recommendations.

The International Organization for Standardization (ISO) exists to develop standards to facilitate the international exchange of goods and services. It is a voluntary, nontreaty group. The members of ISO are representative standards organizations from more than 80 countries. The member organization from the United States is the American National Standards Institute (ANSI). The ISO work on data communications is focused in two subcommittees of Technical Committee 97 (computers and information processing): subcommittee SC6 is responsible for data communications, and subcommittee SC16 is responsible for open systems interconnection. The OSI model described earlier is the work of subcommittee SC16.

EIA stands for Electronics Industries Association, a national body that represents manufacturers in the U.S. electronics industry. The EIA work on data communications is carried out by Technical Committee TR30. EIA standards on data communication are published in their RS-series.

1.6.1. CCITT and Related Protocol Standards

Figure 1-27 illustrates the CCITT terminology for various user nodes and network switching nodes. It also illustrates the boundaries across which protocol standards are defined. Figure 1-28 gives an overview of the prominent CCITT protocol standards and their relationships to each other. A line joining two groupings in Figure 1-28 means that the protocol standards in the two groups are closely related. An arrow from one group to another means that the protocol standards at the arrow's head are incorporated as part of the protocol standards at its tail. CCITT protocol standards illustrated in Figure 1-27 are described below:

X.25	defines protocols across the boundary between a user node (a DTE), and a network switching node (a DCE);
X.75	defines protocols across the boundary between switching nodes (STEs) in two different networks;
X.3	defines a packet assembly/disassembly facility (PAD) in a public data network;
X.28	defines the protocols between a start-stop mode DTE and a PAD in a public data network within the same country; and
X.29	defines the protocols between a PAD and a packet-mode DTE.

In the initial version of X.25 approved in 1976, the network layer protocol (called the X.25 packet level) was defined to provide virtual circuit services. With a virtual call (VC) service, call establishment and clearing procedures are necessary to provide and to release a logical connection between two DTEs. With a permanent virtual circuit (PVC) service, the public data network provides an end-to-end logical connection between two DTEs for an indefinite period. (There are no call establishment and clearing procedures.) A DTE can have up to 4,096 logical connections across an X.25 interface (due to the use of a 12-bit logical channel number).

Various applications such as telemetry and inquiry-response systems (reservation systems, electronic funds transfer, etc.) require the sending or exchange of just a few small data units each time a logical connection is made. To accommodate such applications, a later version of X.25 adopted in 1980 contains two additional services:

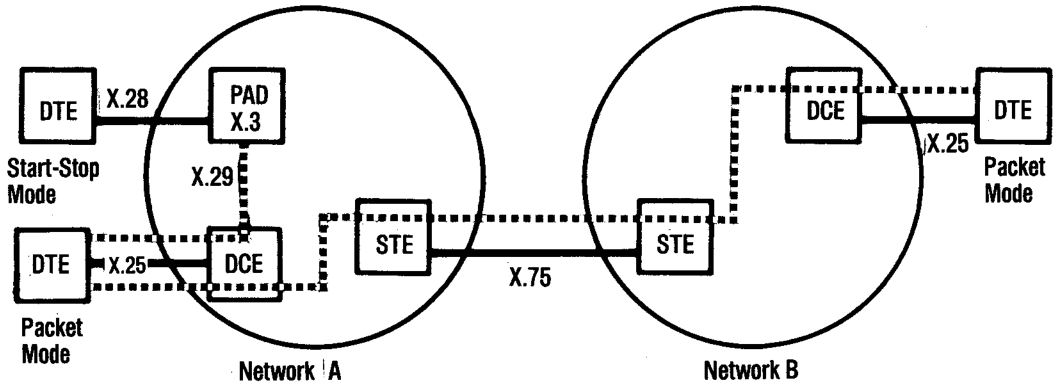


Figure 1-27. An illustration of CCITT terminology.

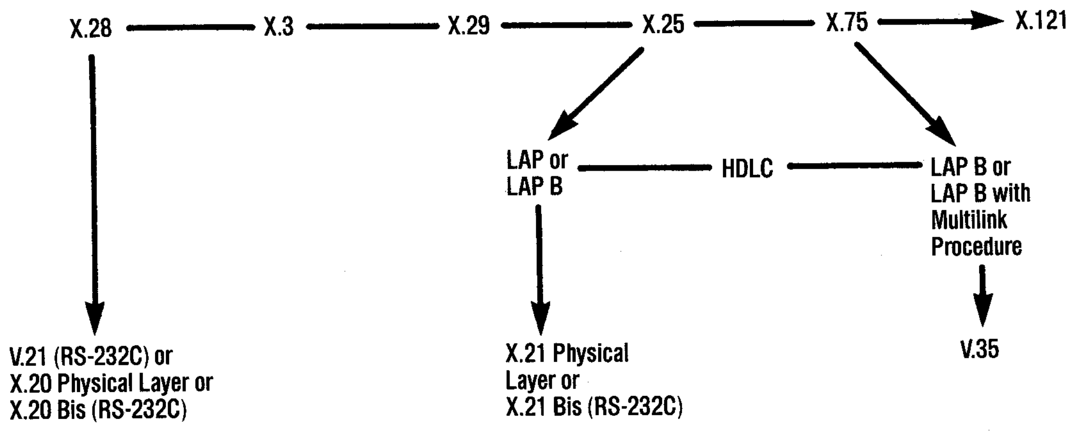

X.28
X.3
X.29
X.25
X.75
X.121
LAP or LAP B
HDLC
LAP B or LAP B with Multilink Procedure
V.21 (RS-232C) or X.20 Physical Layer or X.20 Bis (RS-232C)
X.21 Physical Layer or X.21 Bis (RS-232C)
V.35

Figure 1-28. Relationships between various protocol standards.

a datagram service and a fast-select service. The fast-select service is an extension of the virtual call service such that the Call Request packet can contain up to 128 octets of user data and the destination DTE's response, in the form of a Call Accepted packet or a Clear Request packet, can contain up to 128 octets of user data.

The data link layer protocol of X.25 has two versions: LAP and LAP B. Both were based on the ISO data link protocol standard, HDLC. LAP was developed first and is not quite compatible with HDLC. LAP B was subsequently specified and is compatible with the Asynchronous Balanced Mode (ABM) of HDLC. Both LAP and LAP B are specified, but LAP B is the preferred specification.

The physical layer specification of X.25 is defined to be the same as the physical layer specification of X.21 (or X.21 bis). Note: physical layer interfaces are summarized in Table 1-1 and will be introduced below.

STEs implementing X.75 provide a gateway function between two different networks. Like X.25, X.75 specifies protocols for the network/data link/physical layers. The network layer protocol (called the X.75 packet level) establishes, maintains, and clears virtual calls across the interface between STEs. Various network utilities are defined so that network administrative information may be signalled in the Call Request, Call Connected, and Clear Request packets. An X.75 interface provides a virtual circuit that joins together virtual circuits in the interconnected networks. As in X.25, up to 4,096 logical connections can be multiplexed across an X.75 interface. The data link layer protocol for X.75 is LAP B in X.25. (LAP is not specified.) In addition, a multilink protocol is also specified that allows the interface to operate over multiple lines, thus achieving greater throughput and reliability. The multilink protocol is unique to X.75 at this time.

The X.75 physical layer is defined to be the CCITT V.35 interface to accommodate modems for higher speed circuits than the speeds of modems intended for use with X.25 (e.g., 48 Kbps). There is no corresponding EIA standard, but this interface is used in the United States.

Note that a DTE and a DCE across an X.25 interface are likely to be located at the same place (possibly even in the same room), whereas two STEs across an X.75 interface are likely to be geographically apart; modems are required, and a reliable connection would require the use of multiple links.

The X.3/X.28/X.29 protocols were defined to enable start-stop character-mode terminals to communicate with remote packet-mode DTEs that implement X.25 (i.e., host computers). The functions performed by a PAD and its operational

Table 1-1. Functional, electrical, and mechanical characteristics of four physical layer interface standards.

	Public Data Network Interfaces		Modem Interfaces	
	X.20 (physical layer)	X.21 (physical layer)	RS-449	RS-232C
Functional (interchange circuits)	X.24		compatible with V.24	
Electrical	X.26/V.10 compatible with RS-423A (unbalanced) X.27/V.11 compatible with RS-422A (balanced)			compatible with V.28
Mechanical	ISO DIS 4903 15-pin connector		compatible with ISO DIS 4902 37-pin connector	compatible with ISO DIS 2110 25-pin connector

characteristics depend upon the values of some internal variables called PAD parameters. A set of PAD parameters exists for each start-stop mode DTE independently. (The use of different parameter values gives rise to different interfaces for a variety of terminals.) Either the start-stop mode DTE or the packet-mode DTE or both can configure the PAD (by using the X.28 and X.29 protocols to set its parameters) so that the PAD's operation is adapted to the start-stop mode DTE's characteristics and possibly to the application program in the host computer as well.

Start-stop terminals and their PADs are assumed to be physically apart, and X.28 includes the specification of a physical layer protocol. For a start-stop terminal accessing the PAD via a public switched telephone network or a leased line, the physical layer specification is in accordance with CCITT Recommendation V.21 for an asynchronous modem at rates of up to 300 bps; this is compatible with the EIA RS-232C interface. For a start-stop terminal accessing the PAD via a public switched data network or leased line, the X.20 (or X.20 bis) interface is specified. In addition to a physical layer protocol, X.28 also defines device control functions as well as data link and network layer functions. The character code set of International Alphabet No. 5 (Recommendation V.3) is specified for X.28 data link protocols.

X.29 specifies a protocol between a PAD and a packet-mode DTE to perform network layer functions (establishing, clearing, and resetting of calls), as well as to perform flow control and data transfer functions. X.29 employs the user data field of the X.25 protocol data units to exchange control information and data between a PAD and a packet-mode DTE.

In addition to participating in X.28 and X.29 protocols to handle interactions with start-stop mode and packet-mode DTEs, respectively, a PAD assembles characters into packets destined for packet-mode DTEs and disassembles packets into characters going to start-stop mode DTEs. Like terminal interface processors (TIPs) in the ARPANET, a PAD provides the function of data concentration and the function of terminal handling. Architecturally, however, PADs and TIPs are very different. TIPs are host nodes in the ARPANET; PADs are part of the public data network [DAY 79]. Figure 1-29 illustrates how one would place the X.3/X.28/X.29 functions in relation to the bottom three layers of the OSI model. We have shown them to be straddling the boundary between the network layer and the upper layers. Processes identified by "NET" in Figure 1-29 implement protocols that are internal to the public data network for the physical, data link and network layers.

1.6.2. Data Link Protocol Standards

Data link protocols provide important functions for computer equipment connected by a physical link to communicate with each other (see Section 1.5.2). Their development started prior to the current activities on standards for public data networks. We have already encountered ISO's HDLC protocol standard in the above discussions. HDLC is closely related to ANSI's ADCCP protocol standard and IBM's SDLC protocol. All are essentially compatible. These data link protocols are said to be bit-oriented. An earlier generation of data link protocols, including the ANSI X3.28 protocol standard and IBM's BSC protocol, are character-oriented. (These character-oriented protocols are still widely used in practice.) The HDLC and BSC protocols will be described in Chapter 2 as prominent examples of the classes of bit-oriented and character-oriented protocols.

1.6.3. Physical Layer Interfaces

To define a physical layer interface requires specification of the following characteristics:

1. mechanical
2. electrical
3. functional (interchange circuits and their functions)
4. procedural (the subset of interchange circuits and the protocol using them for connection establishment and clearing)

Four interface standards are shown in Table 1-1: X.21, X.20, RS-232C, and RS-449. All four are serial-data DTE/DCE interfaces. X.21 and X.20 are for accessing a (digital) public data network. RS-232C and RS-449 are for the use of modems. Table 1-1 shows their functional, electrical, and mechanical specifications; these are given in terms

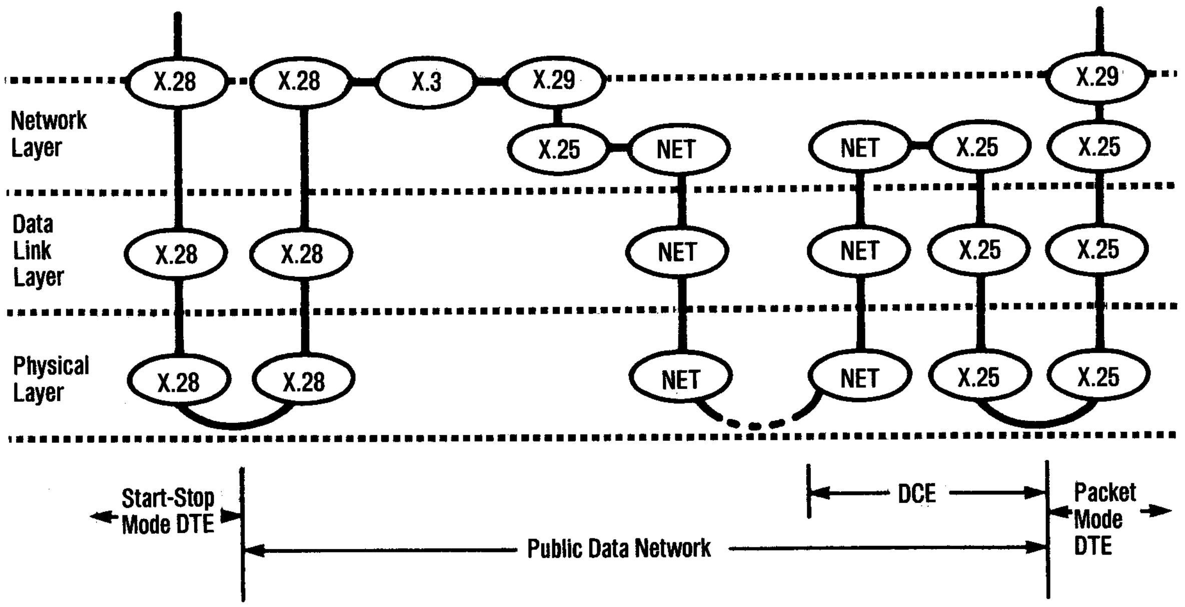


Figure 1-29. Placement of X.3, X.28, and X.29 functions in a structured architectural model.

of CCITT or ISO standards either adopted for the specific characteristics or compatible with the characteristics of these interfaces. (The X.21 and X.20 standards also define data link and network level functions. In this subsection, we are referring to their physical layer specifications only.)

First we look at the modem interfaces. The RS-232C interface is probably the most well known. It was developed by EIA in cooperation with the Bell System, various independent modem manufacturers, and computer manufacturers [MCNA 77]. It can be used with both asynchronous modems and synchronous modems (up to 19.2 Kbps). RS-449 was developed to replace RS-232C for improved performance, such as longer interface cables and a higher maximum data rate (to be achieved with the new electrical characteristics), additional interface functions, and a more precise and better mechanical specification.

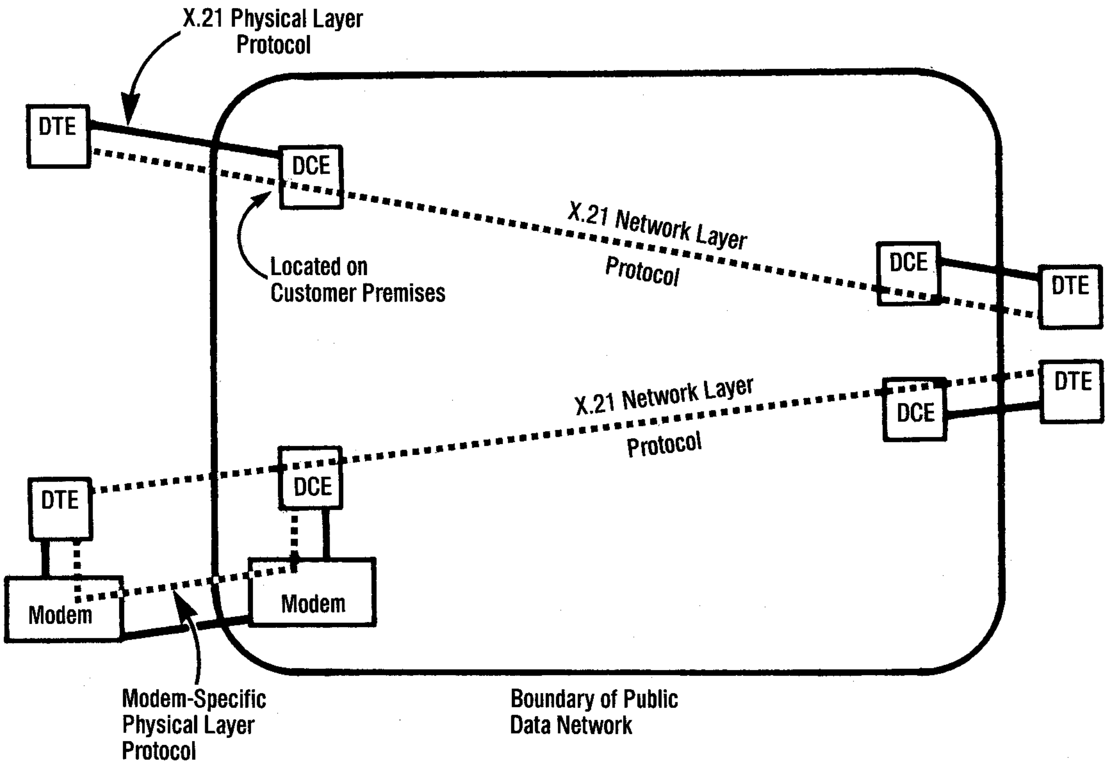
Note that RS-449 has the same electrical characteristics as X.21. Why wasn't the X.21 interface adopted instead of developing a new interface to replace RS-232C? One reason is that the X.21 interface was designed for accessing a public data network, and it therefore defines only a small number of interchange circuits. There are both technical and performance problems in trying to adopt X.21 for use with a wideband modem [BERT 80]. Furthermore, interoperability of RS-449 with RS-232C may be accomplished by means of a simple passive adapter and some additional design criteria for the RS-449 interchange circuits; no modifications are needed for the RS-232C equipment.

In addition to the specifications of functional, electrical and mechanical characteristics shown in Table 1-1, a physical layer interface also requires the specification of a protocol for the establishment/clearing of a DTE/DCE connection, using a subset of interchange circuits. When modems are used, the protocols for using these interchange circuits are specific to different types of modems. Additional interchange circuits are required for a switched-network connection compared to a leased-line connection. CCITT defines various modem-specific procedures for the use of interchange circuits in V-series recommendations. For example, the V.21 recommendation is a procedure for low-speed, asynchronous switched network modems compatible with the functional specification of RS-232C. Obviously, different modem-specific procedures can be defined for the R-232C (or RS-449) characteristics in Table 1-1. Two pieces of equipment are often said to be "RS-232C compatible" if their mechanical, electrical, and functional characteristics are compatible. In fact, for them to be able to communicate they need to implement the same procedural specification as well.

We next consider the public data network interfaces. Recommendation X.20 defines a DTE/DCE interface for start-stop (asynchronous) transmission up to 300 bps on public data networks. (Bit synchronism is actually maintained during the transmission of each character but the time between characters may be arbitrary.) Recommendation X.21 defines a DTE/DCE interface for synchronous transmission on public data networks for a data rate of 600, 2400, 4800, 9600, or 48,000 bps. Both X.21 and X.20 specify circuit-switched and leased-circuit services. The provisions to acquire circuits on demand from a public data network correspond to network layer functions. Thus in their entirety, X.21 and X.20 are more than just a physical layer protocol.

We next briefly describe the X.21 interface. In addition to the specifications in Table 1-1, the procedural specification within the physical layer is given by a state transition diagram of quiescent states; each such state is defined by the conditions on four interchange circuits. One of the states corresponds to both DCE and DTE being ready. X.21 also defines a protocol for the use of a circuit-switched public data network [FOLT 80]. This is a protocol between DTE processes and DCE processes residing in the network layer of the OSI model. Addresses of DTEs are assigned in accordance with recommendation X.121. The protocol is said to be character-oriented because control information for call-establishment is exchanged using control characters from the International Alphabet No. 5 (on the R circuit from DCE to DTE and on the T circuit from DTE to DCE). Two data link layer functions are also defined. The first is byte timing (the determination of octet boundaries). This is accomplished by placing two or more contiguous SYN characters before each control sequence of characters. In addition, byte timing may also be provided by a circuit from DCE to DTE. Simple odd parity is applied to each character for error detection (also a data link layer function).

Let us now return to Figure 1-28. Note that X.25 and X.28 incorporate the physical layer specifications of X.21 and X.20, respectively (but without the data link and network layer functions). Also, X.21 bis and X.20 bis were specified as interim protocols that describe how a public data network should interface with existing DTEs designed for using modems. They simply mean the use of CCITT V-series modem-specific procedures in the physical layer that are compatible with RS-232C (or RS-449). Figure 1-30 illustrates the difference between X.21 and X.21 bis, as well as the fact that X.21 has network layer functions. Note that X.21 specifies the interface between a DTE and a DCE which are physically close. The DCE is considered as part of the public data network although it is located on the premises of the DTE. For a switched, public data network, the network layer protocol of X.21 acquires from the network a circuit to a distant DTE/DCE pair. An X.21 bis interface is intended to be an interim measure for DTEs that access the network via modems. In this case, the network layer functions of X.21 are implemented on top of a modem-specific physical layer interface. (See lower part of diagram in Figure 1-30.)



X.21 Physical Layer Protocol
DTE
DCE
Located on Customer Premises
X.21 Network Layer Protocol
DCE
DTE
X.21 Network Layer Protocol
DTE
DCE
DTE
DCE
Modem
Modem
Modem-Specific Physical Layer Protocol
Boundary of Public Data Network

Figure 1-30. An illustration of X.21 and X.21 bis.

1.6.4. IEEE Local Area Network Standards

IEEE Standard 802 is a family of standards for local area networks. Protocols corresponding to the physical and data link layers of the OSI model are specified. In particular the data link layer is split into two sublayers: a logical link control layer, which implements functions of content and dialog synchronizations, and a multiple access control sublayer, which implements the functions of multiple access and frame synchronizations. As before, the physical layer implements the function of bit synchronization.

A peer protocol is specified for the logical link control sublayer. Two types of link operation are defined: a connectionless service and a connection-oriented service similar to that of HDLC. The interface between this sublayer and the network layer above and the interface between this sublayer and the multiple access control sublayer below are also defined. Several standards are specified for the multiple access control sublayer for different communication media: CSMA/CD for a bus, token-passing for a bus, and token-passing for a ring. The reader is referred to the standards documents for details [IEEE 82] and to Chapters 3 and 4 for discussions on multiple access protocols and local area networks respectively.

1.7. TRENDS AND APPLICATIONS

1.7.1. A Historical Perspective

In a 1978 article, Roberts gave an excellent historical perspective of the development of packet switching up to that time [ROBE 78]. He talked about pioneers and pioneering networks, the development of X.25 and public data networks, and the application of packet switching to satellite and radio communication networks. He observed that packet switching was not an invention. It was simply an application of the dynamic resource allocation principle and was introduced when digital electronics became inexpensive enough to render packet switching cost effective. He said, "In 1978 virtually all new data networks being built throughout the world are based upon packet switching." He also conjectured, "An open question at this time is how long will it take for voice communications to be revolutionized as well by the packet switching technology?" In fact, several researchers have studied the performance of sending digital voice traffic through packet switching networks [FORG 77, GONS 83]. It remains to be seen whether packet switching will displace circuit switching as the communication technology for voice traffic.

In his keynote speech at the 7th Data Communications Symposium, Kleinrock reviewed the fundamental principles that had evolved from a decade of network development. He described some of the successes, some of the technical problems to be addressed, some of the exciting applications, and the telecommunications revolution that had already begun. The 1982 article in the *Journal of Telecommunication Networks*, based on his speech, is reprinted below [KLEI 82].

1.7.2. Future Directions

Nowadays, data and text can be easily and conveniently stored, edited, retrieved, and communicated. The ease and convenience of handling such coded information behooves us to strive for networks and systems that will handle normally uncoded information like voice, image, and video in the same manner (after first digitizing and coding them). Currently, such uncoded information is handled separately, e.g., the use of telephone facilities and PBXs for voice and facsimile, and satellite and other wideband facilities for video. Filing, retrieving, and editing of voice messages, graphics displays etc. can be done but not conveniently. In fact it will be desirable to handle all information, data, text, voice, image, and perhaps also video, in the same communication networks (both local area and wide area networks) and in the same systems for office automation, home automation etc. Glimpses into such a scenario are given in Lewis Branscomb's keynote address at INFOCOM'83 [BRAN 83]. The scenario is also consistent with the CCITT activity on establishing the Integrated Services Digital Network (ISDN) concept for multiservice digital networks [DECI 82].

The biggest impact of office automation has been probably the ease with which documents are handled (created, filed, retrieved, perused, modified, deleted, distributed etc.). In his address, Branscomb envisioned the integration of voice, and data for creating compound documents with voice annotation and playback through a telephone headset. He conjectured that eventually, office systems will have to merge data, text, voice and image information into integrated electronic documents. The user will not be conscious of their form when stored. The transmission system will not care what form--words, voice, or pictures--the information was in originally. (This is of course consistent with the principle of data transparency as discussed earlier.)

Is such a scenario supported by the current PBX and LAN technologies? The basic function of a PBX is to support interactive human-to-human voice conversations. PBXs that support only 64-Kbps voice channels cannot meet the requirements of some future office applications. Hundreds of Kbps are needed for browsing through a document of coded information, and Mbps are needed for facsimile information. Packet-switched LANs do have such high bandwidths available on demand. On the other hand, it is doubtful that some of the current packet-switched LANs utilizing a baseband technology are suited to supporting large numbers of concurrent voice conversations between people in addition to data communications. Branscomb concluded that both a PBX and a LAN would be required to satisfy the needs of office applications in the foreseeable future. It seems that the "electronic paper clip," as referred to by Branscomb, that is needed to bind together documents integrating all kinds of information will have to be realized from one or more of the advanced technologies being developed for the LAN environment: broadband systems, frequency-agile modems, optical fibers, etc.

Let us now move away from the office automation environment to consider wide-area communication services. The ISDN concept is being established for an integration of telephone and data communication services in an evolutionary manner. CCITT Study Group XVIII is coordinating the ISDN activities within CCITT and the recommendation of standard interfaces. ISDNs are envisioned to evolve from the basic digital telephone networks founded on 64 Kbps PCM circuit switching. They will provide end-to-end connectivity (in digital form) to support a wide range of services including voice, data, sound and video transmissions. An ISDN provides a limited set of user interfaces and is recognized by the service characteristics of its interfaces rather than by its internal architecture, configuration, or technology. This concept provides the basis for permitting an evolutionary migration from the current generation of circuit-switched ISDNs. To accommodate services ranging from alarms and telemetry to voice, interactive and bulk data, electronic mail, videotex, and video, a variety of services are provided: leased lines, circuit-switched services, and virtual circuit and datagram services based upon packet switching etc. Internally, an ISDN may have separate communication facilities that constitute separate packet-switching and circuit-switching networks, or it may use a hybrid circuit-switching/packet-switching technology such as the one described earlier in Section 1.4.4. Networks that provide a wideband service may have to incorporate separate wideband facilities such as satellite channels.

1.7.3. Applications

Before we plunge into the technical details of communication and networking protocols in the following chapters, it is helpful to have in the back of our minds an overall view of the different kinds of computer networks together with their present and anticipated applications. We begin by mentioning a few of the pioneering networks.

Some early networks

The ARPANET was developed in the late 1960s with the original intention of sharing special resources, software and hardware, at different research sites. It turned out that this network demonstrated very convincingly the power of electronic mail and document creation and distribution. By linking various research groups together in the United States, it fostered the exchange of ideas and collaboration by researchers far away from each other. At about the same time, Tymnet was developed by Tymshare Corporation in 1970 to connect asynchronous terminals to its timesharing central computers. Airline reservation networks had been in operation since the early 1960s. But these networks, such as SABRE and PARS, were primarily centralized and terminal-oriented networks [SIWI 77]. An exception was the SITA network. SITA is a cooperative company of international air carriers organized to provide its members a world-wide data communication capability via a message switching network. In 1969 SITA began updating its design by replacing the major nodes of its message switching network with high-level network nodes organized to act like a packet-switching network. Incoming messages are subdivided into 240-byte packets and are stored and forwarded along predetermined routes to their destinations. Prestored distributed tables provide for alternate routes in the event of line failures [SCHW 77, ROBE 78].

Private networks

Many private networks exist to handle various general information processing functions of large corporations (information retrieval, file transfers, and document distribution) as well as to handle special applications (order processing, inventory control, etc.). Private networks are either homegrown or supplied by computer manufacturers, such as SNA networks by IBM and networks by DEC in accordance with its Digital Network Architecture (DNA). Private networks for a long time have been mostly centralized and terminal-oriented, meaning that they have a tree topology. The root node of the tree is made up of one or more host computers. The other nodes are either data concentrators or terminals. This approach is exemplified by the pre-SNA networks of IBM as well as SNA networks circa 1974. But the situation has been rapidly changing. The SNA multihost Advanced Communication Function (ACF) announced in 1976 permits "cross-domain" sessions. The SNA Network Interconnection technique was announced in 1983 [IBM 83]. DNA as described in [WECK 80] also permits multiple hosts as well as a mesh network topology.

Industry-wide networks

Various industry-wide networks have been in existence for some time. Aside from the airline reservations networks mentioned above, many financial services networks are in operation [WEIN 82]:

- networks for interbank transfers (electronic funds transfer systems),
- automatic tellermachine (ATM) networks,
- networks for securities trading, and
- networks for credit authorization, etc.

Some new developments are on the horizon. Extending credit authorization networks to include point-of-sale systems (electronic cash registers) is one example. Another development is the use of smart credit cards (containing microprocessors) for both automatic teller machines and point-of-sale systems. It is interesting to note that human operators are sometimes needed in credit authorization systems. In this respect the ISDN concept of integrating both voice and data traffic is quite appropriate here.

Local area networks

Local area networks have been developed and built within the local environments of college campuses and research laboratories. Their most important application, however, will probably be in office automation. The requirements of office automation are many and varied. We review some of them below:

- electronic mail, image mail, and voice mail
- document preparation, filing, perusal, retrieval, and printing
- data entry
- other personal computing functions
    - spread sheets
    - database access
    - graphics
    - calendar support

There are various other special applications for local area networks such as process control in manufacturing and energy management for a local environment.

Public data networks

The success of the packet-switching technology as demonstrated by the ARPANET and the extensive activities in the development of private networks in the early 1970s encouraged the development of public data networks. The most well known of them are Telenet in the U.S. (operational in 1975), Datapac in Canada (operational in 1977) and Transpac in France (operational in 1978). All of these are based upon the packet-switching technology and the use of the X.25 interface for customer access and the X.75 interface for internetworking. Numerous public data networks based upon X.25 have since become operational world-wide.

In the U.S., Tymnet was also approved as a carrier and began supplying public data services in the late 1970s.

These public data networks are often referred to as value added networks (VANs) because they are built on top of common carrier facilities.

Home and small business applications

Networking services to homes and small businesses have aroused a lot of interest. Such services constitute one of the main objectives of ISDNs. While ISDN is a solution that will slowly evolve from the current digital telephone networks, another possible solution is that of metropolitan area networks (MANs) based upon CATV systems. The IEEE 802 Standards documents contain a provision for such networks (IEEE 802.6 yet to be defined). If it is anything like the LAN network standards, the metropolitan area networks will be packet-switched systems that interconnect to packet-switched public data networks via gateways. Thus the ISDN and MAN concepts represent two solutions coming from different directions.

Various home applications that have been talked about include

- electronic mail,
- telebanking,
- teleshopping, and
- electronic publishing (database information services, news, etc.).

These home applications will involve personal computers. In the case of teleshopping that requires graphics and pictures with colors, videotex systems based upon consumer TV technology and the use of videodiscs are being investigated.

Various telemetry applications have also been mentioned in conjunction with ISDN. They include alarms, meter reading, energy management, and opinion polls, among others.

Some of the above-mentioned applications for homes and small businesses are already available, although the technologies of ISDN and MAN will not appear for some time.

CSNET provides an electronic mail service

Table 1-2. Some generic traffic models.

Communicating entities	Specific application examples	Traffic model	Special characteristics
terminal & terminal	electronic mail	data messages	short; not necessarily bursty
terminal & database	information retrieval	inquiry/response	interactive; asymmetric traffic
	reservation systems; financial services	transaction processing	interactive; typically bursty; database updates
terminal & program	time-sharing; A.I. applications such as medical diagnosis	remote job entry	may be interactive or batch
	data entry; telemetry	data messages	asymmetric traffic
program/database & program/database	file transfers	bulk data	could be bursty

to computer science researchers in the United States and Canada. It is a high-level logical network that spans several physical networks: Phonenet, ARPANET, and Telenet [DENN 83]. Plans are underway to develop both domestic and international gateways to connect to other networks as well.

Several systems--ITT, MCI Mail, Western Union's EasyLink and Graphnet's Freedom Network--provide a store-and-forward Telex service for people with personal computers. These vendors maintain store-and-forward computers and offer local or toll-free numbers for customers to call. To send a message, a customer dials up a store-and-forward computer using a modem and his telephone. He is assigned an electronic mailbox in the store-and-forward computer from which he can retrieve messages sent to him by others. Numerous information databases are also available to people with personal computers. Access is again via a modem and dial-up telephone service.

Sears, IBM, and CBS announced a videotex service in February 1984 for shopping, financial and information services. It is also based upon the use of personal computers and the existing telephone system.

Traffic models

The applications and potential applications of communication networks are only limited by one's imagination. From the point of view of the designers of communication and networking protocols, it will be convenient to classify the myriad of applications into a small number of generic traffic models. We make an attempt at such a classification below by considering the different ways of pairing up the data traffic sources and sinks: terminals, databases, and programs (Table 1-2).

An important decision to make in the design of communication networks is whether to allocate communication bandwidth in small units for a relatively long duration (allocation of channels) or in large units for a relatively short duration (allocation to waiting packets). Such a decision is dependent upon the burstiness of traffic sources. Packet switching is suitable for applications with bursty traffic sources whereas circuit switching is not. A measure defined in [LAM 78] suggests that the burstiness of a traffic source is determined by the ratio of its mean response time requirement and the mean interarrival time of its data units. In particular ***the smaller is this ratio for a particular traffic source, the more bursty it is.*** Thus, an electronic mail system that specifies a mean delivery time of one day would be non-bursty and can probably be efficiently handled by some form of circuit switching (e.g., Telex). On the other hand, file transfers for a distributed database system with short response time requirements of say a few seconds would be bursty, and a packet-switching network would be required. Thus, burstiness is not necessarily associated with short units of data.

Lastly, with the anticipated integration of data, voice, image and video traffic into the same networks, additional traffic models will have to be characterized and understood.

References

[BARN 77] Barnla, J. D. and F. R. Zitzmann, "Digital Communications Satellite System of SBS," IEEE Electron. and Aerosp. Syst. Conv., Sept. 1977; reprinted in *Satellite* Communications, H. L. Van Trees (Ed.), IEEE Press, 1979.

[BERT 80] Bertine, H. V., "Physical Level Protocols," *IEEE Trans. on Commun.*, Vol. COM-28, No. 4, April 1980, pp. 443-444.

*[BRAN 83] Branscomb, L. M., "Networks for the Nineties," Keynote address, IEEE INFOCOM'83; in *IEEE Communications Magazine*, October 1983, pp. 38-43.

[BRIN 73] Brinch Hansen, P., *Operating System Principles*, Prentice-Hall, Englewood Cliffs, N.J., 1973.

[CCIT 78] CCITT Grey Book, *Provisional Recommendations X.3, X.25, X.28 and X.29 on packet-switched data transmission services*, Geneva, 1978.

[CERF 78] Cerf, V. G. and P. T. Kirstein. "Issues in Packet-Network Interconnection," *Proc. IEEE* 66, 11, November 1978, pp. 1386-1408.

[CHU 69] Chu, W. W., "A Study of Asynchronous Time Division Multiplexing for Time-sharing Computers," *AFIPS Proceedings*, Vol. 35, FJCC, 1969, pp. 669-678.

[CHU 74] Chu, W. W., "Optimal Message Block Size for Computer Communications with Error Detection and Retransmission Strategies," *IEEE Trans. on Commun.*, Vol. COM-20, No. 10, October 1974.

[CYPS 78] Cypser, R. J., *Communications Architecture for Distributed Systems*, Addison-Wesley, Reading, Mass., 1978.

[DAVE 72] Davey, J. R., "Modems," *Proc. IEEE*, November 1972, pp. 1284-1292.

[DAY 79] Day, J. D., "Resource Sharing Protocols," *Computer*, Vol. 12, No. 9, September 1979, pp. 47-56.

[DECI 82] Decina, M., "Progress Towards User Access Arrangements in Integrated Services Digital Networks," *IEEE Trans. on Commun.*, Vol. COM-30, No. 9, Sept. 1982, pp. 2117-2130.

[DENN 83] Denning, P. J., A. Hearn, and C. W. Kern, "History and Overview of CSNET," *Proc. ACM SIGCOMM'83 Symp. on Comm. Architectures and Protocols*, University of Texas at Austin, March 1983, pp. 138-145.

[FOLT 80] Folts, H. C., "X.25 Transaction-Oriented Features--Datagram and Fast Select." *IEEE Trans. Commun.*, Vol. COM-28, April 1980, pp. 496-500.

[FORG 77] Forgie, J. W. and A. G. Nemeth, "An Efficient Packetized Voice/Data Network using Statistical Flow Control," *Conf. Rec. ICCC'77*, Chicago, pp. 38.2-44 to 38.2-48, 1977.

[GERL 80] Gerla, M. and L. Kleinrock, "Flow Control: A Comparative Survey," *IEEE Trans. Comm.*, COM-28 (1980), pp. 553-574.

[GONS 83] Gonsalves, T. A., "Packet-voice Communication on an Ethernet Local Computer Network: An Experimental Study," *Proc. SIGCOMM'83 Symp. on Comm. Architectures and Protocols*, University of Texas at Austin, pp. 178-185, March 1983.

[IBM 83] Systems Network Architecture Special Issue, *IBM Systems Journal*, Vol. 22, No. 4, 1983.

[IEEE 82] IEEE Project 802 Local Area Network Standards, Draft IEEE Standard 802.2, Logical Link Control, Draft D, Nov. 1982; Draft IEEE Standard 802.3, CSMA/CD Access Method and Physical Layer Specifications, Revision D, Dec. 1982; Draft IEEE Standard 802.4, Token-Passing Bus Access Method and Physical Layer Specifications, Draft D, Dec. 1982. (Available from IEEE Computer Society.)

[KERM 79] Kermani, P. and L. Kleinrock, "Virtual Cut-Through: A New Computer Communication Switching Technique," *Comput. Networks*, Vol. 3, 1979, pp. 267-286.

[KLEI 76] Kleinrock, L., *Queueing Systems, Vol. 2: Computer Applications*, Wiley-Interscience, New York, 1976.

*[KLEI 82] Kleinrock, L., "A Decade of Network Development," *Journal of Telecommunication Networks*, Spring 1982, pp. 1-11.

[LAM 78] Lam, S. S., "A New Measure for Characterizing Data Traffic," *IEEE Trans. Commun.*, Vol. COM-26, January 1978.

[LAM 80] Lam, S. S., "A Packet Network Architecture for Local Interconnection," *Conf. Rec. Int. Conf. Commun.*, Seattle, June 1980.

[LAM 83] Lam, S. S., "Data Link Control Procedures," Chapter 3 in *Computer Communications, Vol. 1: Principles*. W. Chou (Ed.), Prentice-Hall, Englewood Cliffs, N.J., 1983.

[MART 76] Martin, J., *Telecommunications and the Computer*, Second Edition, Prentice-Hall, Englewood Cliffs, N.J., 1976.

[MCNA 77] McNamara, J. E., *Technical Aspects of Data Communication*, Digital Equipment Corp., Maynard, Mass., 1977.

[MCQU 78] McQuillan, J. M. and V. G. Cerf, *A Practical View of Computer Communications Protocols*, IEEE Computer Society Press, Silver Springs, Md., 1978.

[OSI 81] "Data Processing--Open Systems Interconnection--Basic Reference Model." Document ISO/TC97/SC16, American National Standards Institute, 1430 Broadway, New York, N.Y.; *Computer Networks*, Vol. 5, No. 2, 1981, pp. 81-118.

[POUZ 78] Pouzin, L. and H. Zimmerman, "A Tutorial on Protocols," *Proc. IEEE*, 66, 11, November 1978, pp. 1346-1370.

*[ROBE 78] Roberts, L., "The Evolution of Packet Switching," *Proc. IEEE*, Vol. 66, November 1978.

[RUTL 82] Rutledge, J. H., "OSI and SNA: A Perspective," *Journal of Telecommunication Networks*, Vol. 1, No. 1, 1982, pp. 13-27.

[RYBC 80] Rybczynski, A., "X.25 Interface and End-to-End Virtual Circuit Service Characteristics," *IEEE Trans. Commun.*, Vol. COM-28, April 1980, pp. 500-510.

[SCHW 77] Schwartz, M., *Computer-Communication Network Design and Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1977.

[SCHW 80] Schwartz, M., *Information Transmission, Modulation, and Noise*, Third Edition, McGraw-Hill, New York, 1980.

[SHOC 80] Shoch, J. F., and J. A. Hupp, "Measured Performance of an Ethernet Local Network," *Communications of the ACM*, 23 (12), December 1980.

[SIWI 77] Siwiec, J. E., "A High-performance DB/DC System," IBM Systems Journal, Vol. 16, No. 2, 1977, pp. 169-195.

[UNSO 81] Unsoy, M. S. and T. Shanahan, "X.75 Internetworking of Datapac and Telenet." *Proc. 7th Data Communication Symp.*, Mexico City, October 1981, pp. 232-239.

[VITE 66] Viterbi, A. J., *Principles of Coherent Communication*, McGraw-Hill, New York, 1966.

[WECK 80] Wecker, S., "DNA: The Digital Network Architecture," *IEEE Trans. on Commun.*, Vol. COM-28, No. 4, April 1980.

[WEIN 82] Weinstein, S. B., "A Perspective on Financial Industry Networking," *Journal of Telecomm. Networks*, Vol. 1, No. 4, 1982, pp. 317-332.

(* article reprinted below.)