Communication Networks

Sharam Hekmat
PragSoft Corporation
www.pragsoft.com

Contents

Preface

This book is concerned with post-computer communication networks and two of its important streams: data communication and telecommunication. Data communication refers to the communication between digital computers, facilitated by computer networks. Telecommunication refers to the primarily human-to-human communication facilitated by the global telephone system. The differences between these two streams are mainly due to historical reasons. Telecommunication is increasingly relying on digital computer technology, and data communication is relying more than ever on telecommunication networks. The two streams are rapidly converging.

Newcomers to this field are often bewildered by the substantial wealth of information already published on the subject. This book is aimed at this group of people. It provides a broad coverage of the key concepts, techniques, and terminology, so as to prepare readers for more advanced discussions. In-depth discussions of technically-involved topics are intentionally avoided in favor of more general concepts. No previous knowledge of networks or programming is assumed.

The structure of the book is as follows. Chapter 1 introduces computer networks and explains some of their elementary concepts. It also introduces the OSI reference model, upon which later chapters are based. Each of Chapters 2-8 describes one of the seven layers of the OSI model in the context of wide area data networks. Chapter 9 looks at local area networks and their applications. Chapter 10 provides an introduction to telecommunication. Chapter 11 builds on earlier chapters by examining ISDN as the merging point of data and voice networks. Chapter 12 looks at the ATM technology and the potential applications that it can support.

1. Introduction

A **computer network** is the infrastructure that allows two or more computers (called **hosts**) to communicate with each other. The network achieves this by providing a set of rules for communication, called **protocols**, which should be observed by all participating hosts. The need for a protocol should be obvious: it allows different computers from different vendors and with different operating characteristics to 'speak the same language'.

This chapter introduces the fundamental concepts of computer networks. We will first look at constituent network components and various network types, and then describe a reference model for network protocol architectures which we will expand upon throughout the rest of this book. We will also discuss the role of international standards and major standards organizations.

After reading this chapter you should be able to:

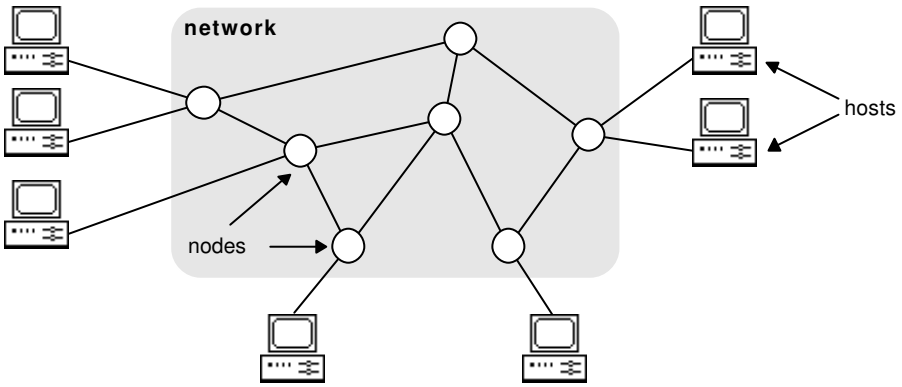
- Describe the general characteristics of a computer network.
- Understand the role of the major components of a computer network.
- Distinguish between different network types and understand their properties.
- Appreciate the relevance and importance of standards, in general, and the OSI model, in particular.
- Describe the role and functions of each of the OSI layers.
- Use sequence and state transition diagrams to interpret and describe protocols.
- Appreciate the wealth of knowledge available on communication networks.

1.1. Network Components

Figure 1.1 shows an abstract view of a network and its hosts. The network is made up of two types of components: **nodes** and communication **lines**. The nodes typically handle the network protocols and provide switching capabilities. A node is usually itself a computer (general or special) which runs specific network software. The communication lines may take many different shapes and forms, even in the same network. Examples include: copper wire cables, optical fiber, radio channels, and telephone lines.

A host is connected to the network by a separate communication line which connects it to one of the nodes. In most cases, more than one host may be connected to the same node. From a host's point of view, the entire network may be viewed as a black box, to which many other hosts are connected. Each host has a unique **address** allocated to it by the network. For a host to communicate with another host, it needs to know the latter's address. All communication between hosts passes through the nodes, which in turn determine how to route messages across the network, from one point to another.

Figure 1.1 An abstract network.



Throughout the rest of this book, there will be occasions when it is not necessary to distinguish between hosts and nodes. In such cases, we will use the term **station** to mean either.

1.2. Network Types

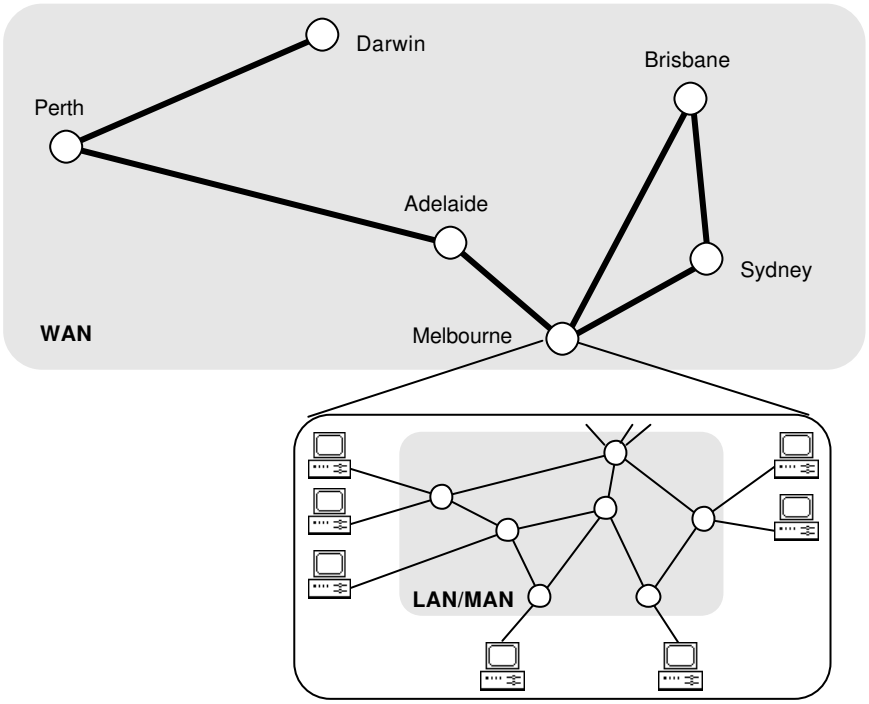
Networks may be divided into different types and categories according to four different criteria:

1. *Geographic spread of nodes and hosts.* When the physical distance between the hosts is within a few kilometers, the network is said to be a **Local Area**

Network (LAN). LANs are typically used to connect a set of hosts within the same building (e.g., an office environment) or a set of closely-located buildings (e.g., a university campus). For larger distances, the network is said to be a **Metropolitan Area Network** (MAN) or a **Wide Area Network** (WAN). MANs cover distances of up to a few hundred kilometers and are used for inteconnecting hosts spread across a city. WANs are used to connect hosts spread across a country, a continent, or the globe. LANs, MANs, and WANs usually coexist: closely-located hosts are connected by LANs which can access hosts in other remote LANs via MANs and WANs, as illustrated in Figure 1.2.

2. *Access restrictions.* Most networks are for the private use of the organizations to which they belong; these are called **private networks**. Networks maintained by banks, insurance companies, airlines, hospitals, and most other businesses are of this nature. **Public networks**, on the other hand, are generally accessible to the average user, but may require registration and payment of connection fees. Internet is the most-widely known example of a public network. Technically, both private and public networks may be of LAN, MAN, or WAN type, although public networks, by their size and nature, tend to WANs.

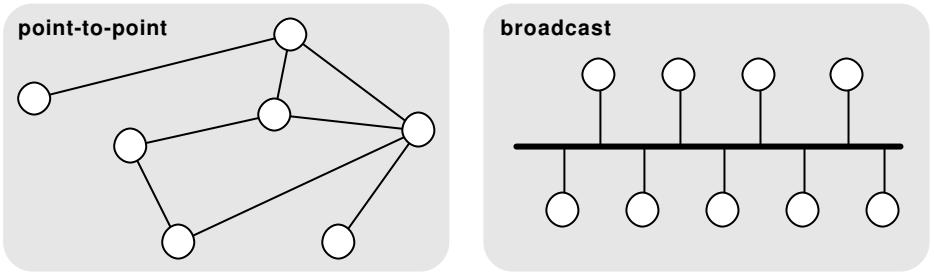
Figure 1.2 Example of a WAN between LANs.



3. *Communication model employed by the nodes.* The communication between the nodes is either based on a **point-to-point** model or a **broadcast** model (see Figure 1.3). In the point-to-point model, a message follows a specific route

across the network in order to get from one node to another. In the broadcast model, on the other hand, all nodes share the same communication medium and, as a result, a message transmitted by any node can be received by all other nodes. A part of the message (an address) indicates for which node the message is intended. All nodes look at this address and ignore the message if it does not match their own address.

Figure 1.3 Communication models.



4. *Switching model employed by the nodes.* In the point-to-point model, nodes either employ **circuit switching** or **packet switching**. Suppose that a host A wishes to communicate with another host B. In circuit switching, a dedicated communication path is allocated between A and B, via a set of intermediate nodes. The data is sent along the path as a continuous stream of bits. This path is maintained for the duration of communication between A and B, and is then released. In packet switching, data is divided into packets (chunks of specific length and characteristics) which are sent from A to B via intermediate nodes. Each intermediate node temporarily stores the packet and waits for the receiving node to become available to receive it. Because data is sent in packets, it is not necessary to reserve a path across the network for the duration of communication between A and B. Different packets can be routed differently in order to spread the load between the nodes and improve performance. However, this requires packets to carry additional addressing information.

1.3. The OSI Model

The International Standards Organization (ISO) has developed a reference model for network design called the **Open Systems Interconnection** (OSI). It proposes a seven-layer architecture for networks, as summarized by Figure 1.4. Each layer is characterized by a set of standard protocols which specify its behavior.

Figure 1.4 The OSI reference model.

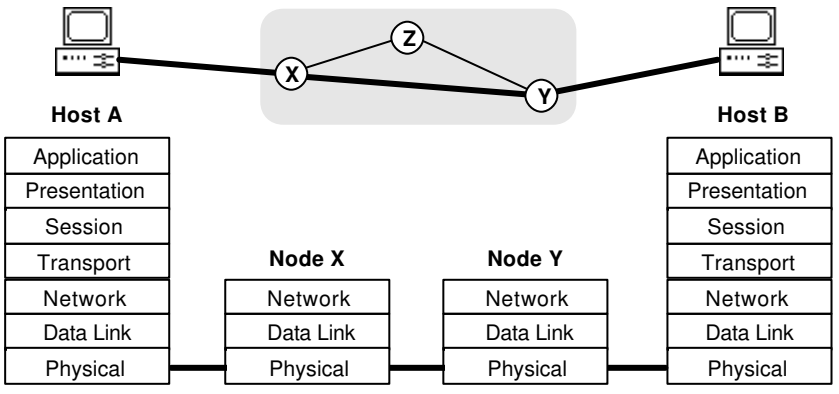
Layer	Name	Data Unit	Main Function

7	Application	Message	Mutually-agreeable meaning of application data (common semantics).
6	Presentation	Message	Mutually-agreeable binary representation of application data (common syntax).
5	Session	Message	Negotiation of the establishment and termination of connections (sessions).
4	Transport	Message	Efficient and cost-effective transportation of data across the network.
3	Network	Packet	Routing of data packets within the network and across multiple networks.
2	Data Link	Frame	Provision of a reliable communication line to the network layer.
1	Physical	Bit	Transmission of raw data bits over communication lines.

These seven layers represent the protocol architecture for the communications component of a host. The nodes in a network implement only the lower three layers, as illustrated in Figure 1.5. The reason for this is that the upper four layers are irrelevant to the task of communication between the nodes.

In Figure 1.5, when host A sends a message to host B, the message moves down the successive layers of host A, from the application layer to the presentation layer, to the session layer, etc., until it reaches the physical layer. It is then transmitted across the communication line between host A and node X, and moves up the three layers of node X and down again. Then it is transmitted to node Y where it goes through the same procedure, and finally is transmitted to host B, where it moves up its seven layers, until it arrives at the application layer of host B.

Figure 1.5 Nodes use only the lower 3 layers.

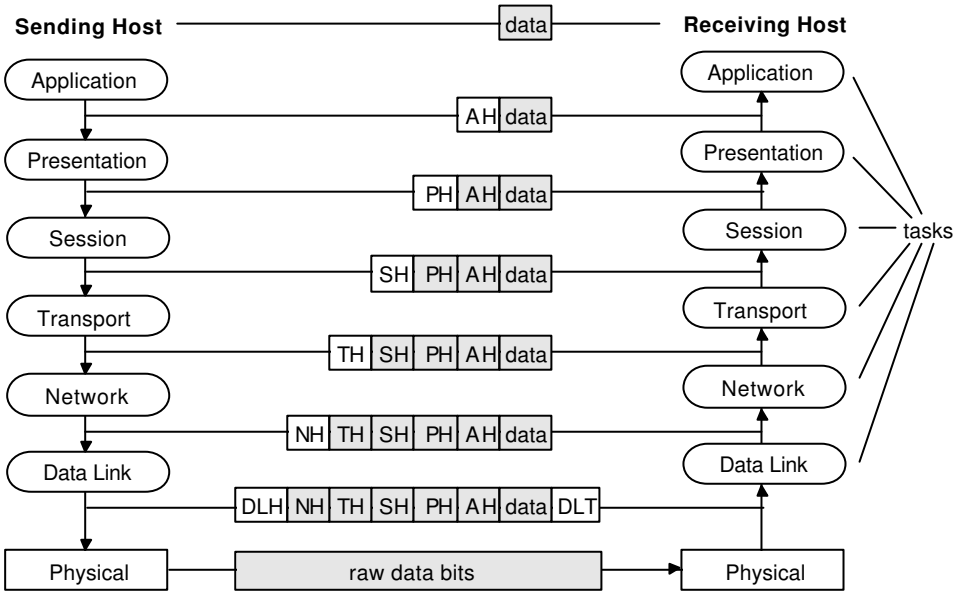


Although actual communication takes place only at the physical layer, it is often useful to think of virtual communication between corresponding layers. For example, we can use an imaginary line of communication between the presentation layer on host A and the same layer on host B. This would be characterized by the *presentation protocol*.

The terms *protocol* and *layer* are often used interchangeably. This is harmless but not entirely accurate. Strictly speaking, protocol refers to the rules and conventions that the functions of a layer should conform to. Layer refers to a set of services and functions and their realization in hardware or software. A layer is therefore characterized by its protocol. A set of network layers is also commonly referred to as a **protocol stack**.

Each of the seven layers of the OSI model hides the implementation details of the lower layers from the upper layers. Well-defined protocols and interfaces for each of the layers make it possible for the layer to be designed and implemented in isolation from the other layers. Except for the physical layer, which is implemented in hardware, all other layers are implemented in software.[1] For example, each of these layers may be implemented as a set of routines which communicate with the layer above and the layer below it via parameters passed in function calls. Alternatively, each layer may be implemented as a task (in a multi-tasking environment) which communicates with other tasks by message passing. Figure 1.6 illustrates the latter.

Figure 1.6 OSI layers as software tasks.



For house-keeping purposes, each layer adds an additional piece of information to the message it is transmitting. The same layer removes the additional piece of information on the receiving end. The additional information appears in form of a header (e.g., TH = Transport Header). The data link layer adds a header as well as a trailer to its data.

[1] The Data Link layer is also often implemented in hardware for efficiency reasons. Custom chips are typically used for this purpose.

Each of the seven layers of the OSI model is described below in more detail. Subsequent chapters examine the layers in greater depth and discuss their main protocols. It should be pointed out that the OSI model is not the only model in use. It is, however, the most-widely respected model and has become a standard benchmark for comparing other network architectures against.

1.3.1. The Physical Layer

The physical layer is concerned with the transmission of raw data bits over communication lines. Physical layer standards and protocols are concerned with issues such as the following:

- How a physical circuit is established between communicating devices.
- How the circuit is terminated when no longer needed.
- The physical form (e.g., voltages, frequencies, timing) in which data bits (binary values 0 and 1) are represented.
- Whether transmission of data can take place in one or both directions over the same physical connection.
- Characteristics of the physical media that carry the signals (e.g., copper wire, optical fiber, radio waves).
- Characteristics of the connectors used for connecting the physical media.
- How data from a number of sources should be multiplexed before transmission and demultiplexed upon arrival, and the type of multiplexing technique to be used.
- The type of modulation to be used for transmitting digital data over analog transmission lines.

The physical layer accounts for much of the tangible components of a network, including cables, satellites, earth stations, repeaters, multiplexers, concentrators, and modems. Physical layer protocols and standards are of mechanical, electrical, functional, and procedural nature.

The physical layer hides the above details from the higher layers. To the data link layer, it appears as a **logical** communication channel which can send a stream of bits from one point in the network to another (but not necessarily reliably).

1.3.2. The Data Link Layer

The data link layer is concerned with the reliable transfer of data over the communication channel provided by the physical layer. To do this, the data link layer breaks the data into **data frames**, transmits the frames sequentially over the channel,

and checks for transmission errors by requiring the receiving end to send back **acknowledgment frames**. Data link protocols are concerned with the following issues:

- How to divide the data into frames.
- How to delimit frames by adding special bit patterns to the beginning and end of each frame. This allows the receiving end to detect where each frame begins and where it ends.
- Error detection. Some form of error check is included in the frame header. This is constructed by the transmitting end based on the contents of the frame, and checked for integrity by the receiving end. A change in the frame bits can be detected in this way.
- Error correction. When a frame arrives corrupted or is for any reason lost in the network, it is retransmitted. Lost acknowledgment frames may result in duplicate frames, which need to be detected and corrected as well.
- Flow control. In general, not all communication devices in a network operate at the same speed. Flow control provides a means of avoiding a slow receiver from being swamped by data from a fast transmitter.

The data link layer hides the above details from the higher layers. To the network layer, it appears as a **reliable** communication channel which can send and receive data packets as frames.

1.3.3. The Network Layer

The network layer is concerned with the routing of data across the network from one end to another. To do this, the network layer converts the data into **packets** and ensures that the packets are delivered to their final destination, where they can be converted back into the original data. Network layer protocols are concerned with the following issues:

- The interface between a host and the network.
- The interface between two hosts across the network.
- Routing of packets across the network, including the allocation of a route and handling of congestion.
- Correct ordering of packets to reflect the original order of data.
- Collection of statistical information (e.g., number of transmitted packets) for performance measurement and accounting purposes.
- Internetworking: communication between two or more networks.

The network layer hides the above details from the higher layers. To the transport layer, it appears as a **uniform** data transfer service, regardless of the location of the communicating devices and how they are connected.

1.3.4. The Transport Layer

The aim of the transport layer is to isolate the upper three layers from the network, so that any changes to the network equipment technology will be confined to the lower three layers (i.e., at the node level). Transport layer protocols are concerned with the following issues:

- Establishment and termination of host-to-host connections.
- Efficient and cost-effective delivery of data across the network from one host to another.
- Multiplexing of data, if necessary, to improve use of network bandwidth, and demultiplexing at the other end.
- Splitting of data across multiple network connections, if necessary, to improve throughput, and recombining at the other end.
- Flow control between hosts.
- Addressing of messages to their corresponding connections. The address information appears as a part of the message header.
- Type of service to be provided to the session layer (e.g., error-free versus error-prone connections, whether messages should be delivered in the order received or not).

The transport layer hides the above details from the higher layers. To the session layer, it appears as a **customized** data transfer service between two hosts, isolating the underlying network technology from it.

1.3.5. The Session Layer

The session layer provides a structured means for data exchange between user processes on communicating hosts. Session layer protocols are concerned with the following issues:

- Negotiating the establishment of a connection (a **session**) between user processes on communicating hosts, and its subsequent termination. This includes the setting of various communication parameters for the session (e.g., synchronization and control).
- Correct ordering of messages when this function is not performed by the transport layer.

- Recovery from interrupted transport connections, if necessary.
- Grouping of messages into a larger message, if necessary, so that the larger message becomes available at the destination only when its constituent messages have all been delivered successfully.

The session layer hides the above details from the higher layers. To the presentation layer, it appears as an **organized** communication service between user processes.

1.3.6. The Presentation Layer

The presentation layer provides a mutually-agreeable binary representation of the application data communicated between two user processes. Since there are many ways of encoding application data (e.g., integers, text) into binary data, agreement on a common representation is necessary. Presentation layer protocols are concerned with issues such as the following:

- Abstract representation of application data.
- Binary representation of application data.
- Conversion between the binary representation of application data and a common format for transmission between peer applications.
- Data compression to better utilize network bandwidth.
- Data encryption as a security measure.

The presentation layer hides the above details from the higher layers. To the application layer, it appears as a **universal** communication service between user processes, regardless of their system-specific idiosyncrasies, allowing them to converse in a common **syntax**.

1.3.7. The Application Layer

The application layer is concerned with the **semantics** of data, i.e., what the data means to applications. The application layer provides standards for supporting a variety of application-independent services. Examples include:

- Virtual terminal standards to allow applications to communicate with different types of terminals in a device-independent manner.
- Message handling system standards used for electronic mail.
- File transfer, access, and management standards for exchanging files or parts thereof between different systems.

- Transaction processing standards to allow different companies with different systems to access each other's on-line databases (e.g., in banking and airline reservation).
- On-line directory standards for storing details of individuals, organizations, and network components.
- Standards for exchanging formatted documents.

Application layer standards have paved the way for open software systems, in which data can be communicated between incompatible base systems (i.e., different hardware and software architectures) without loss of meaning or usefulness.

1.4. Protocol Notations

OSI network protocols are specified in a variety of notations. This section describes two popular notations, **sequence diagrams** and **state transition diagrams**, which are extensively used in standards and the literature. Both rely on the notion of a **service primitive** which is described first.

1.4.1. Service Primitives

A service primitive is an abstract representation of the interaction between a **service provider** and a **service user**. Service primitives are concerned with *what* interactions take place rather than *how* such interactions are implemented. Service primitives may be of one of the following four types:

- **Request Primitive**. This is issued by a service user to the service provider to request the invocation of a procedure.
- **Indication Primitive**. This is issued by the service provider to a peer service user (usually in response to a request primitive) to indicate that a procedure has been requested.
- **Response Primitive**. This is issued by a peer service user to the service provider (usually in response to an indication primitive) to indicate that the requested procedure has been invoked.
- **Confirm Primitive**. This is issued by the service provider to a service user to indicate that an earlier request for the invocation of a procedure has been completed.

An actual service primitive consists of a command and, if appropriate, a set of associated parameters. A simple convention is used for naming primitives: a primitive name consists of the first letter of the layer to which it belongs, followed by its

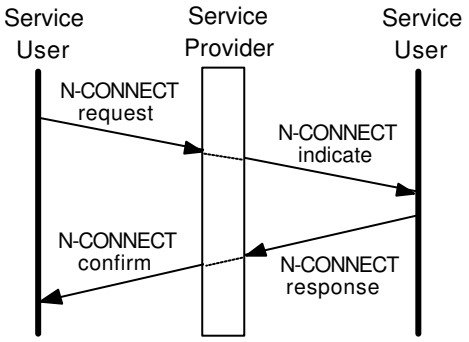
command name, followed by its type. For example, a request type primitive at the network layer for initiating a connection is named 'N-CONNECT request'.

1.4.2. Sequence Diagrams

A sequence diagram defines a service protocol by specifying the permissible sequence of service primitives that may be exchanged between service users and service providers. Service users and service providers are represented by vertical bars. Service primitives are represented by directed lines between the bars. For clarity, primitive parameters are not included.

Figure 1.7 shows a simplified example of requesting a connection at the network layer. According to the diagram, a service user can request, from the service provider, a connection to a peer service user. The service provider in turn issues a connection indication to the peer service user. The peer service user responds to the service provider which, in turn, confirms the cycle with the original service user.

Figure 1.7 A simple sequence diagram.



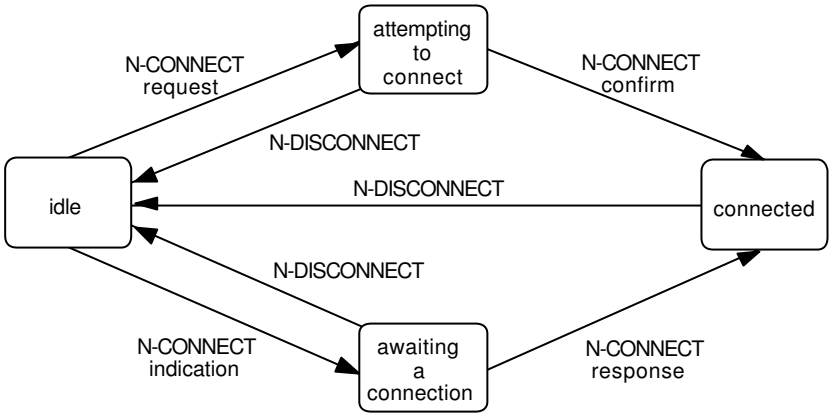
1.4.3. State Transition Diagrams

A state transition diagram describes the various execution states a station can assume and how service primitives cause it to transit from one state to another. States are represented by circles or boxes, and are labeled with a meaningful name that describes the state. A state transition is represented by a directed line from one state to another, and is labeled with the service primitive that triggers the transition.

Figure 1.8 shows an example which describes (in a simplified form) the states of a station at the network layer. According to the diagram, assuming that a station is in the *idle* state, if it issues a connection request to another station, it enters the *attempting to connect* state where it waits for a connection to be confirmed, in which case it moves to the *connected* state, or disconnected, in which case it returns to the *idle* state. A similar scenario applies to an incoming connection which starts with the station receiving a connection indication. Note that the N-DISCONNECT primitives can be either of request or confirmation type.

It is worth noting the complementary nature of sequence diagrams and state transition diagrams. The former specifies a service protocol from an outside observer's point of view, while the latter describes the same protocol from a station's point of view. The two notations, combined, provide a complete picture of how a protocol operates.

Figure 1.8 A simple state transition diagram.



1.5. Standards

The importance of standards in the field of communication cannot be overstressed. Standards enable equipment from different vendors and with different operating characteristics to become components of the same network. Standards also enable different networks in different geographical locations (e.g., different countries and continents) to be interconnected. From a customer's point of view, standards mean real cost savings: the same end-user device can be used for access to a variety of networks and services.

Standards are developed by national and international organizations established for this exact purpose. During the course of this book we will discuss a number of important standards developed by various organizations, including the following:

- The **International Standards Organization** (ISO) has already been mentioned. This is a voluntary organization with representations from national standards organizations of member countries (e.g., ANSI), major vendors, and end-users. ISO is active in many area of science and technology, including information technology. ISO standards are published as *ISO serial-no* (e.g., ISO 8632).
- The **Consultative Committee for International Telegraph and Telephone** (CCITT) is a standards organization devoted to data and telecommunication, with representations from governments, major vendors, telecommunication

carriers, and the scientific community. CCITT standards are published as *Recommendation L.serial-no*, where L is a letter of the alphabet (e.g., I.440). These are revised and republished every four years. CCITT standards are very influential in the field of telecommunications and are adhered to by most vendors and carriers.

- The **Institute of Electrical and Electronic Engineers** (IEEE) is a US standards organization with members throughout the world. IEEE is active in many electric and electronic-related areas. The IEEE standards for local area networks are widely adopted and will be discussed in Chapter 9. IEEE standards are published as *IEEE serial-no* (e.g., IEEE 908).
- The **Electronic Industries Association** (EIA) is a US trade association best known for its EIA-232 standard, which will be discussed in the next chapter.
- The **European Computer Manufacturers Association** (ECMA) is a standards organization involved in the area of computer engineering and related technologies. ECMA directly cooperates with ISO and CCITT.

In addition to these organizations, and because of their global market influence, large vendors occasionally succeed in establishing their products as *de facto* standards. We will also look at a few standards of this nature later in the book.

1.6. Further Reading

Communication is a vast subject area with many branches. Unfortunately the great majority of publications in this area are not at an introductory level. The most serious barriers a newcomer is faced with are layers of nomenclature and an inexhaustible supply of acronyms. One of the objectives of this book is to get the reader past these initial hurdles so that the publicly available literature becomes more accessible.

Overall, there are four good sources of reading to look into. These, roughly in increasing order of detail and complexity, are:

- **Books**. There are numerous text and reference books available on the subject. De Noia (1987), Martin and Leben (1988), Tanenbaum (1989), and Halsall (1992) are all useful readings. Hughes (1992) is an introductory text with emphasis on practice. Stamper (1991) is a well-illustrated introduction with excellent examples. Black (1989) and Stallings (1994) are examples of highly detailed texts, covering a wide range of protocols. Marshall (1990), Dickson and LLoyd (1992), and Jain and Agrawala (1993) present balanced accounts of the OSI model. Stallings (1990) serves as a good reference on communications standards. Brown *et al* (1993) provide a very useful compilation of OSI terms and acronyms.

- **Technical Magazines and Journals**. There are many technical communications magazines and journals in circulation throughout the world, and new ones appear every year. The following are primarily US-based, but have a global readership, and are available in most university libraries:
  - *Bell Systems Technical Journal*
  - *Computer Communication Review*
  - *Computer Networks*
  - *Data Communications*
  - *IBM Systems Journal*
  - *IEEE Communications Magazine*
  - *IEEE Computer*
  - *IEEE Journal on selected Areas in Communication*
  - *IEEE Transactions on Communications*
  - *Journal of Telecommunication Networks*
  - *Proceedings of the IEEE*
  - *Telecommunications*
- **Product information**. Also worth reading are publications by vendors on their network products. Product glossies, advertising literature, and user manuals are usually of introductory nature and easy to understand. Product handbooks, product specifications, and data books are typically at a much more technical level and therefore contain substantial detail. Unfortunately, it is not always easy to get hold of these publications, especially for larger products, such as telephone switches.
- **Published standards**. Communication standards are by no means easy reading material. These standards are often heavily cross-referenced and intended for very technically-minded readers. They are essential reading for those involved in the design and manufacturing of communication hardware and software, as they provide the necessary level of protocol specification detail required for these purposes. The CCITT standards as well as many others are available in most university libraries.

1.7. Summary

- A computer **network** consists of **nodes** and communication **links** which implement its **protocols**. It interconnects a set of **hosts** which conform to the network protocols.
- A network may be classified as a **LAN**, **MAN**, or **WAN**, depending on its geographic spread, and as **private** or **public**, depending on its access restrictions. It may employ a **point-to-point** or a **broadcast** communication

model. A point-to-point model may be based on **circuit switching** or **packet switching**.

- The **OSI model** proposes a seven-layer architecture for networks. Each layer is characterized by a set of protocols. The network nodes implement only the bottom three layers, while the hosts implement all the layers.
- The **physical layer** controls the transmission of raw data bits over communication lines. The **data link layer** facilitates the reliable transfer of data over communication channels. The **network layer** controls the end-to-end routing of data across the network. The **transport layer** manages the efficient and cost-effective transportation of data across the network. The **session layer** manages the negotiation of the establishment and termination of connections (sessions). The **presentation layer** provides a mutually-agreeable binary representation of application data (syntax). The **application layer** provides a mutually-agreeable meaning of application data (semantics).
- A **service primitive** is an abstract representation of the interaction between a **service provider** and a **service user**, and may be of one of four types: request, indication, response, and confirmation.
- A **sequence diagram** defines a service protocol by specifying the permissible sequence of service primitives that may be exchanged between service users and service providers.
- A **state transition diagram** describes the various execution states a station can assume and how service primitives cause it to transit from one state to another.
- Communication **standards** are essential in order to achieve interoperability between different equipment and networks.

1.8. Exercises

1.1 Provide three arguments in favor of the use of a computer network in a modern organization, and at least one argument against.

1.2 Classify the networks operated and/or utilized by your organization as LAN, MAN, WAN, private, public, point-to-point, broadcast, circuit-switched, or packet-switched.

1.3 Discuss and compare the advantages and disadvantages of circuit switching versus packet switching. Name at least one well-known network which is based on either type of switching.

1.4 Explain the rationale behind the OSI seven-layer model. Briefly describe the role of each layer and its main functions.

1.5 What is a service primitive? Describe the main four types of primitives used for defining protocols.

1.6 Explain how sequence and state transition diagrams can be used to specify protocols. What aspect of a protocol is better captured by either diagram?

1.7 Draw a sequence diagram for the following: A service user sends a SEND request to a service provider which in turn sends a SEND indication to the peer service user. The latter sends a DATA request to the service provider which in turn send a DATA indication to the original service user. The peer service user then sends a SEND response to the service provider which in turn sends a SEND confirmation to the original service user.

1.8 Draw a state transition diagram for the following: A station is originally in the *notsync* state. A SYNC request or indication will cause it to enter the *sync* state. While in this state, a RESET indication will cause it to return to the *notsync* state.

2. The Physical Layer

This chapter examines the physical layer of the OSI model in detail. We will first look at a categorization of networking equipment, and then discuss transmission-related issues, including various transmission media. Multiplexing methods will be described next, followed by a discussion of two important physical layer standards: RS-232 and X.21.

After completing this chapter you should be able to:

- Distinguish between different network equipment types and understand their roles.
- Distinguish between different device connection types.
- Understand how data is transmitted and the basic techniques that this process involves.
- Have a broad understanding of the different physical transmission media and their characteristics.
- Understand the basic multiplexing methods and their role in data transmission.
- Have a basic knowledge of physical layer standards RS-232 and X.21.

2.1. Equipment

This section briefly describes general networking equipment types and the types of connections that can be established between them.

2.1.1. Equipment Types

Network equipment may be classified into three broad categories:

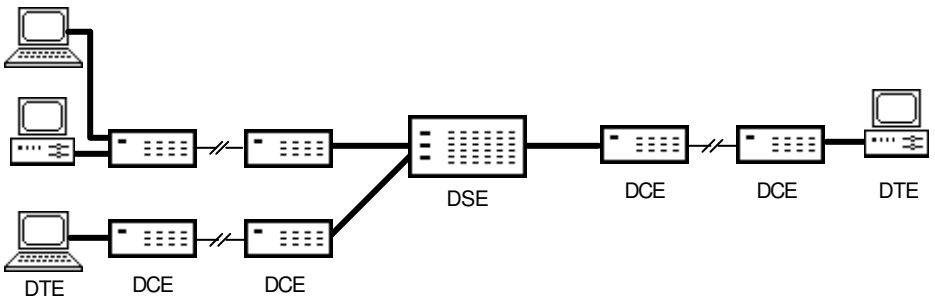
1. **Data Terminal Equipment (DTE)** refers to user equipment that convert outgoing user data into a transmission signal, and convert back the incoming signal into user data. DTEs may take many different shapes and forms. Examples include: terminals, terminal adapters, personal computers, and mainframes. DTEs commonly reside at user sites.

2. **Data Circuit-terminating Equipment (DCE)** refers to the network equipment that connect DTEs to the network communication lines. In general, a DTE and a network line may use different types of signals (e.g., electrical versus optical). The necessary signal conversion between these two is performed by the DCE. A DCE may be a part of a DTE or be an entirely separate device. Modems and multiplexers are all examples of DCEs.

3. **Data Switching Equipment (DSE)** refers to network equipment used to connect DCEs together, thus providing switching capability to the network. DSEs correspond to the nodes in a network, and are responsible for routing data across the network. A DSE is commonly referred to as a **switch**. Digital telephone switches used in digital networks are examples.

Figure 2.9 illustrates the way DTEs, DCEs, and DSEs are connected in a network.

Figure 2.9 Network equipment types.



2.1.2. Connection Types

Connections between devices may be classified into three categories:

1. **Simplex.** This is a unidirectional connection, i.e., data can only travel in one direction. Simplex connections are useful in situations where a device only receives or only sends data (e.g., a printer).

2. **Half-duplex.** This is a bidirectional connection, with the restriction that data can travel in one direction at a time.

3. **Full-duplex.** This is a bidirectional connection in which data can travel in both directions at once. A full-duplex connection is equivalent to two simplex connections in opposite directions.

2.2. Transmission

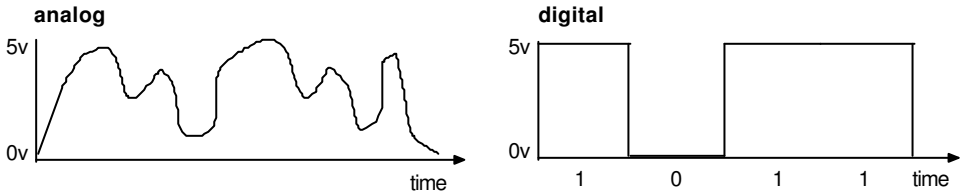
Transmission is the act of transporting information from one location to another via a signal. The signal may be analog or digital, and may travel in different media.

2.2.1. Signal Types

All signals are either analog or digital. An **analog signal** is one in which information appears as a continuous variation of some property. Human speech is an example: it produces a continuous variation of air pressure. A **digital signal**, on the other hand, is one in which information appears as a sequence of binary values 0 and 1. To represent these two values, a signal is used in which only two wave shapes are allowed, one representing the binary value 0 and the other representing the binary value 1. By definition, therefore, a digital signal is a restricted form of an analog signal. A human speaker who only utters the two words *zero* and *one* is a crude example of a digital signal.

In electrical terms, signals appear as variation of some electrical property (e.g., voltage). Figure 2.10 illustrates. In the analog signal example, the voltage freely varies between 0 and 5 Volts. In the digital signal, the voltage may assume only two values: 0 Volts to represent digital value 0 and 5 Volts to represent digital value 1.

Figure 2.10 Analog and digital signals.

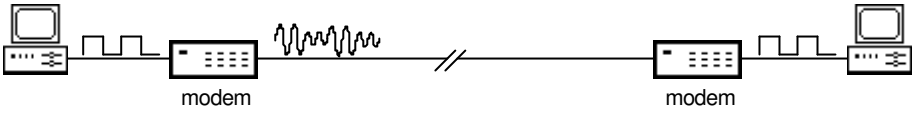


Since digital computers play a central role in data communication, in nearly all cases, digital signals are used. Analog signals are used in cases of equipment which date back to before the advent of digital technology. Existing analog telephone networks are a good example of the latter.

2.2.2. Modulation

Transmission of digital data over an analog line is achieved using a technique called **modulation**, where the digital bit stream is modulated over an analog carrier signal. A **modem** (modulator and demodulator) is a commonly used device which employs this technique. As illustrated in Figure 2.11, a modem converts the outgoing digital bit stream from a device into an analog signal and converts the incoming analog signal into a digital bit stream.

Figure 2.11 Role of modems.



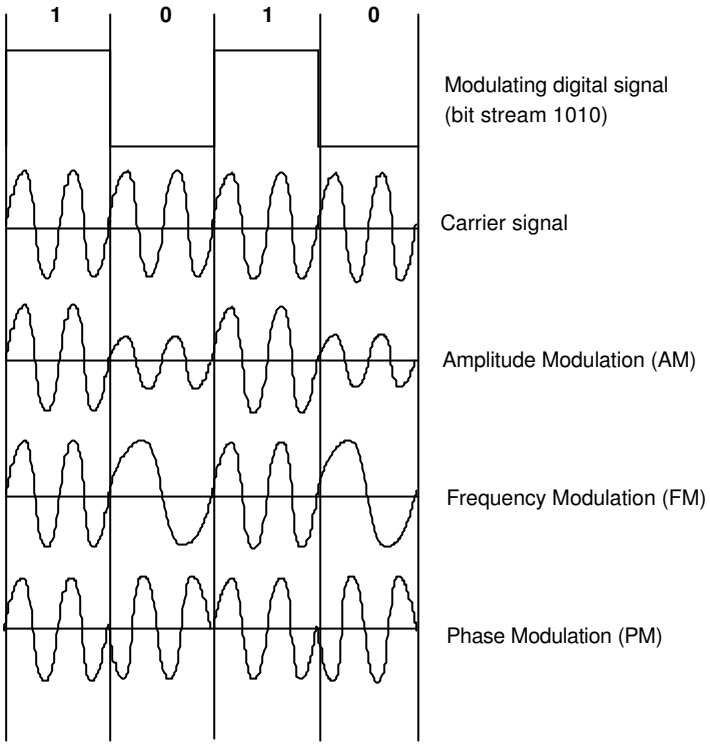
Three basic types of modulation are possible (see Figure 2.12 for a visual comparison):

1. **Amplitude Modulation (AM).** In AM, the carrier signal's *amplitude* is changed according to the modulating digital signal's bit value. For example, two amplitude sizes (a small and a large one) may be used to, respectively, represent bit values 0 and 1. AM's main weakness is its susceptibility to distortion.

2. **Frequency Modulation (FM).** In FM, the carrier signal's *frequency* is changed according to the modulating digital signal's bit value. For example, two frequency values (a low and a high one) may be used to, respectively, represent bit values 0 and 1. FM is more resistant to distortion than AM.

3. **Phase Modulation (PM)**. In PM, the carrier signal's *phase* is changed according to the modulating digital signal's bit value. A change in the carrier signal's phase indicates a change in the modulating digital signal's bit value from 0 to 1 or from 1 to 0.

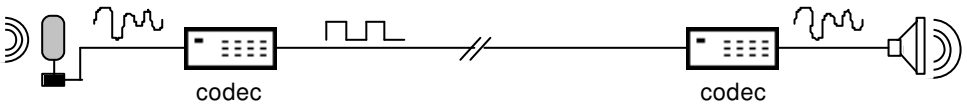
Figure 2.12 Three basic modulation methods.



2.2.3. Digitization

Digitization is essentially the opposite of modulation. Whereas in modulation a digital signal is modulated over an analog signal for transmission, in digitization an analog signal is converted into digital format through a process of sampling. For example, the analog signal resulting from human speech can be sampled and converted into digital data, transmitted over digital lines, and converted back to analog signal at the other end. These two functions are performed by a device called **codec** (coder/decoder). Figure 2.13 illustrates the concept.

Figure 2.13 Role of codecs.



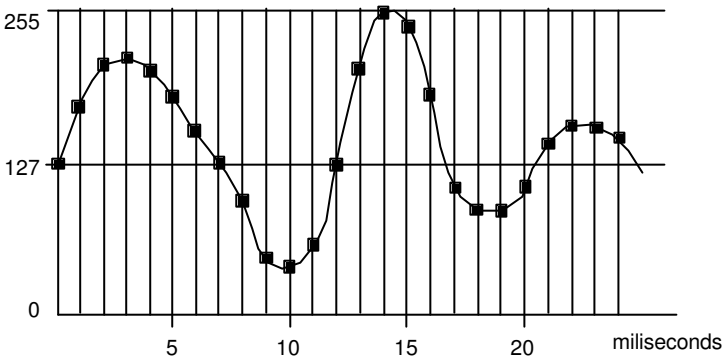
It is worth noting that, unlike modulation (which is an exact process since the digital signal at the source and the digital signal received at the destination are identical), digitization is only an approximate process because of sampling. Figure 2.14 illustrates how an analog signal is sampled. Here the time interval for each sample is one millisecond. Each sample (denoted by a small black box) is a real

value which is in turn represented by an integer in the range 0-255 so that it can be represented in one byte of data. This process (of representing a continuous value with a discrete value) is called **quantization**. The relatively small loss of information inherent in the process is called **quantization error**.

The coding process generates the sample data from the analog signal. The decoding process regenerates an approximation of the original signal by fitting a smooth curve to the sampled points. The quality of the regenerated signal can be improved by increasing the sampling rate (i.e., reducing the sampling interval), but up to a limit dictated by the Nyquist's theorem. This limit is exercised by a popular digitization technique called **Pulse Code Modulation** (PCM) which uses a sampling rate twice that of the original signal frequency. For example, a 4 kHz speech signal is sampled at a rate of 8000 samples per second.

The main advantage of digitization is that, due to its resistance to distortion, it is much easier to reliably transmit a digital signal over a long distance than an analog signal.

Figure 2.14 Sampling an analog signal.



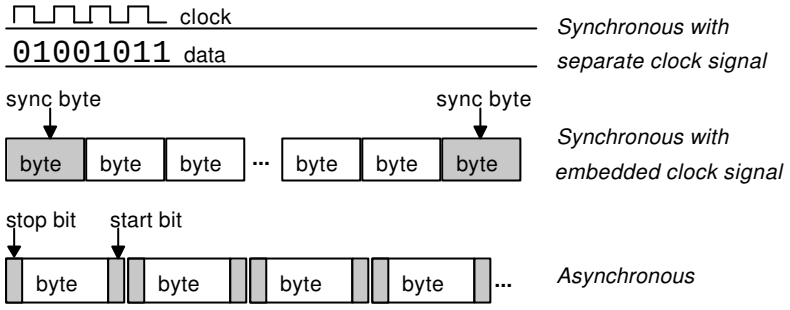
2.2.4. Synchronization

When two devices are about to communicate, the transmitter should somehow notify the receiver as to when to expect to receive data. This allows the receiver to prepare itself for receiving the data. Furthermore, such notifications should occur frequently enough so that both devices maintain an agreement about the exact distribution of data over time. This process is called **synchronization**.

There are two basic methods of synchronization: synchronous transmission and asynchronous transmission. In **synchronous transmission**, a clock signal is used as a common source of reference by both the transmitter and the receiver. By tying the data signal to the clock signal, either device can look at the clock signal to know where data bits may begin or end. The clock signal may be provided on a separate line, or be embedded in the data signal itself (see Figure 2.15). Because having a

separate clock line increases the costs, it is only used for covering very short distances (e.g., for connecting personal computers).

Figure 2.15 Synchronous and asynchronous transmission methods.



In **asynchronous transmission**, the beginning and end of each byte of data is marked by start and stop bits. This enables the receiver to work out the byte boundaries (see Figure 2.15). Because of its simplicity, asynchronous transmission is cheaper to implement and is therefore more widely used.

2.2.5. Transmission Media

Digital data can be transmitted over many different types of media. Selecting a transmission medium is guided by comparing transmission requirements against the medium's characteristics. Four important criteria influence the choice:

1. **Bandwidth.** Bandwidth is the maximum frequency range that can be practically supported by a medium. This is usually expressed in kilo Hz (kHz) or mega Hz (MHz). For example, analog transmission of human speech typically requires a bandwidth of 4 kHz. Also related, is the notion of **data rate**, which denotes the maximum number of bits per second (bps) that can be transmitted. For example, a data rate of 10 mbps means that 10 million bits of data can be transmitted in each second. Because of their obvious relationship, the terms bandwidth and data rate are sometimes used interchangeably. Because of distortion factors, bandwidth and data rate are usually inversely proportional to the communication distance.

2. **Cost.** Two types of cost are relevant: (i) the cost of installing the medium, including the medium-specific equipment that may be needed, and (ii) the cost of running and maintaining the medium and its equipment. There is usually a need for tradeoff between cost, bandwidth, and distance.

3. **Reliability.** Some media, by their physical nature, transmit data more reliably than others. Low reliability translates into a higher number of errors, which needs to be balanced against the potential cost of recovering from the errors (e.g., retransmission, more complex hardware and software).

4. **Coverage.** The physical characteristics of a medium dictate how long a signal can travel in it before it is distorted beyond recognition. To cover larger areas, repeaters are needed to restore the signal, and this increases the costs.

Transmission media may be classified into the following categories:

- **Copper Wire**. This is the oldest form of electronic transmission medium. Its use dates back to the development of telegraph in the 1800s and earliest telephone systems. Early installations used open wires, but these were superseded by twisted pairs, which consist of a pair of insulated and twisted wires (see Figure 2.16). Twisted pairs are superior because of reduced crosstalk.[2] They are very effective for relatively short distances (a few hundred feet), but can be used for up to a few kilometers. A twisted pair has a bandwidth to distance ratio of about 1 MHz per kilometer. The performance of the twisted pair can be substantially improved by adding a metallic shield around the wires. Shielded wires are much more resistant to thermal noise and crosstalk effects. Twisted pairs used for long distance connections (e.g., telephone lines) are usually organized as a much larger cable containing numerous twisted pairs.
- **Coaxial Cable**. A coaxial cable consists of four concentric cylinders: an inner conductor, surrounded by an insulating cylinder, surrounded by an outer conductor, surrounded by a final protective cover. This combination is called a coax (see Figure 2.16). Coaxial cables are superior to twisted pairs both in terms of bandwidth and communication distance, and can provide bandwidth to distance ratios in order of 10s of MHz per kilometer. Like twisted pairs, multiple coaxes are usually housed within one cable, which may also contain twisted pairs. Coaxial cables are extensively used in LANs and long distance telephone trunk lines.
- **Optical Fiber**. An optical fiber consists of two concentric cylinders: an inner core surrounded by a cladding. Both the core and the cladding are made of transparent plastic or glass material (see Figure 2.16). The core is used for guiding a light beam, whereas the cladding (which has a different refractive index) acts as a reflector to prevent the light from escaping from the core. Because optical fiber uses a light signal instead of electrons, it does not suffer from the various noise problems associated with electromagnetic signals. The signal is usually generated by a laser or Light Emitting Diode (LED). Optical fibers can provide bandwidth to distance ratios in order of 100s of MHz per kilometer. Like other cables, hundreds of optical fibers are usually housed within one cable. They are being increasingly used by telecommunication carriers for long distance

[2] Crosstalk is the unwanted coupling effect between two or more signal paths, which causes signal distortion.

digital trunk lines. Current trends promise that they will replace twisted pair residential loops in the near future.

- **Radio**. Radio signals have been used for a long time to transmit analog information. They are particularly attractive for long distance communication over difficult terrain or across the oceans, where the cost of installing cables can be too prohibitive. A minimum radio system consists of a transmitter and a receiver. It may operate at a variety of frequency bands, ranging from hundreds of Hz to hundreds of giga Hz (GHz). A huge range of transmission bandwidths are therefore possible. Microwave is by far the most widely used form of radio transmission. It operates in the GHz range with data rates in order of 100s of mbps per channel. Telecommunication carriers and TV stations are the primary users of microwave transmission.

  An important form of microwave system is a **satellite** system, which is essentially a microwave system plus a large repeater in the sky (see Figure 2.16). The signals transmitted by earth stations are received, amplified, and retransmitted to other earth stations by the satellite. Like other microwave systems, the bandwidth is subdivided into channels of 10s of MHz each, providing data rates in order of 100s of mbps. Because of their high bandwidths, satellites are capable of supporting an enormous number and variety of channels, including TV, telephone, and data. The satellite itself, however, represents a major investment and typically has a limited lifetime (at most a few decades).

  Another increasingly-popular form of radio is **cellular radio**, which is currently being used by carriers for providing mobile telephone networks. These operate in the VHF band and subdivide their coverage area into conceptual cells, where each cell represents a limited area which is served by a low-power transmitter and receiver station. As the mobile user moves from one cell area to another, its communication is handed over from one station to another.

- **Infra-red**. Infra-red signals are suitable for transmission over relatively short distances (the signal is easily reflected by hard objects). The signal is generated and received using optical transceivers. Infra-red systems represent a cheap alternative to most other methods, because there is no cabling involved and the necessary equipment is relatively cheap. Data rates similar to those of twisted pairs are easily possible. However, applications are limited because of distance limitations (of about one kilometer). One recent use of infra-red has been for interfacing hand-held and portable computing devices to LANs (see Figure 2.16).

Figure 2.16 Transmission media.

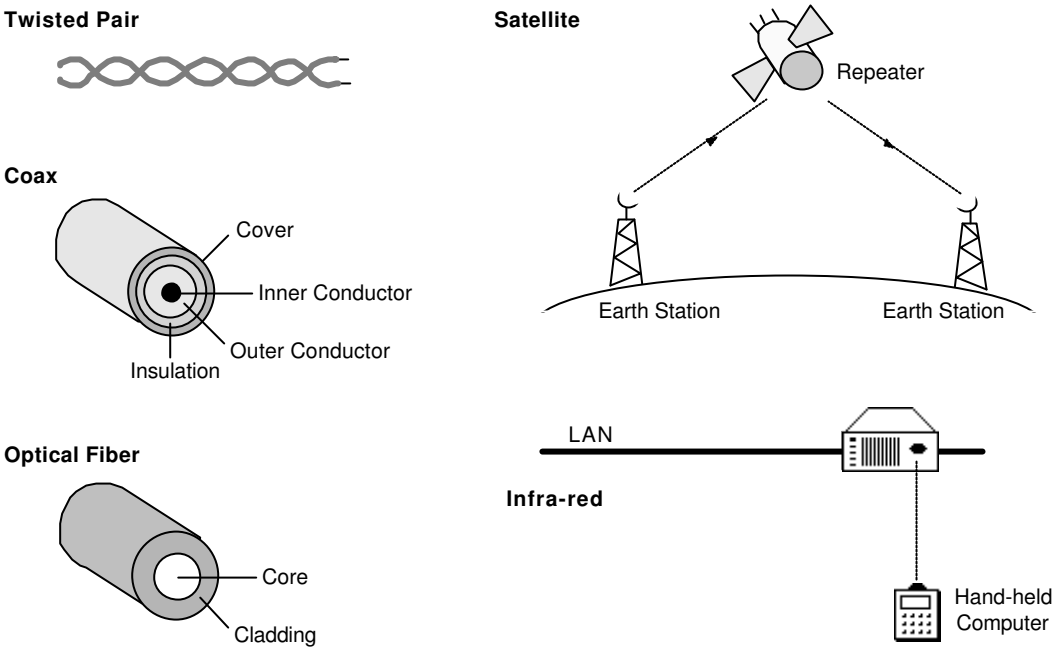


Figure 2.17 compares the characteristics of these media using the criteria mentioned earlier. It is important to note that the figures provided are approximate and continually improve as the technology moves forward.

Figure 2.17 Relative comparison of transmission media.

Medium	Bandwidth	Data Rates	Cost	Reliability	Coverage
Copper Cable	1 MHz	1-10 mbps	Medium/km	Low-Medium	Kilometers
Coaxial Cable	10s of MHz	10-100 mbps	High/km	Medium-High	10s of Kilometers
Optical Fiber	100s of MHz	100s of mbps	High/km	Very High	10s of Kilometers
Radio	100s of MHz	100s of mbps	Very High	Very High	1000s of Kilometers
Infra-red	1 MHz	1-10 mbps	Low	Low-Medium	Kilometer

2.3. Multiplexing

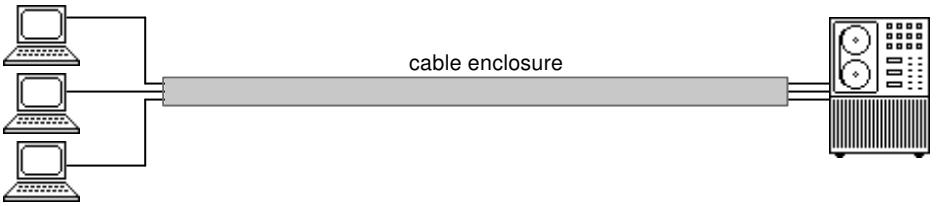
Multiplexing is a technique which makes it possible to cram a number of logical channels (each capable of supporting an independent connection) into the same physical channel or line. The objective of multiplexing should be obvious: to reduce costs by better utilizing the capacity of a line. There are three basic multiplexing methods; these are separately described below.

2.3.1. Space Division Multiplexing (SDM)

SDM is the simplest (and crudest) form of multiplexing. It involves grouping many separate wires into a common cable enclosure. A cable that has, for example, 50 twisted pairs inside it can support 50 channels. There is therefore a one-to-one correspondence between physical and logical channels (see Figure 2.18).

SDM has the unique advantage of not requiring any multiplexing equipment. It is usually combined with other multiplexing techniques to better utilize the individual physical channels.

Figure 2.18 Space division multiplexing.

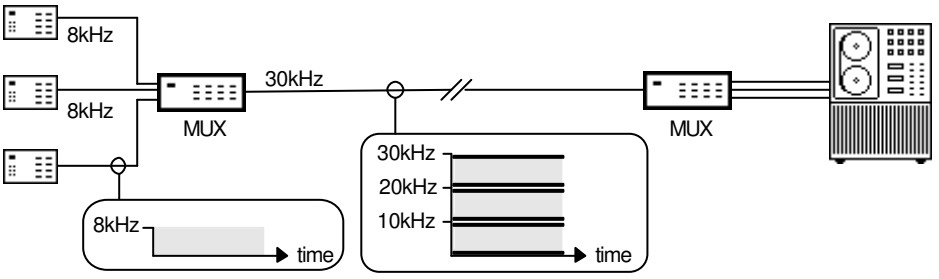


2.3.2. Frequency Division Multiplexing (FDM)

In FDM, the frequency bandwidth of the line is divided into a number of partitions, each of which is used as a separate logical channel. Radio and TV broadcasting represent the oldest examples of FDM. To avoid neighboring channels from interfering with one another, the extreme ends of the channel frequencies are left unused to provide a gap. For example, a line that has a bandwidth of 30 kHz can be divided into 3 times 10 kHz channels, each of which consists of 8 kHz of bandwidth for data and two gaps of 1 kHz on either side.

FDM requires special multiplexing/demultiplexing hardware (MUX) at either end of the line (see Figure 2.19).

Figure 2.19 Frequency division multiplexing.

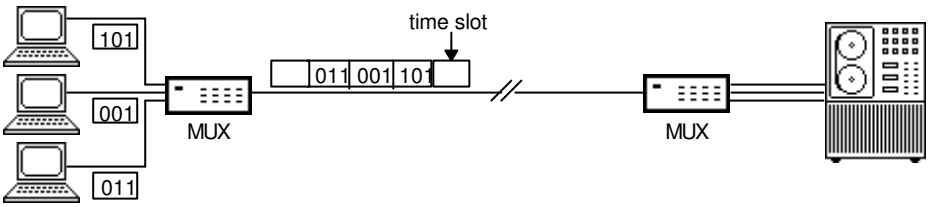


2.3.3. Time Division Multiplexing (TDM)

In TDM, each logical channel is allocated a time slot to transmit over a shared physical channel. For example, each logical channel may be given a 5 millisecond time slot to transmit, during which time it will have the entire bandwidth of the line to itself.

Like FDM, TDM requires special multiplexing/demultiplexing hardware (MUX) at either end of the line (see Figure 2.20). Because the channels are spread across time, some means of initial synchronization is also needed. Basically, the receiving end needs to know which time slot belongs to the first channel when the connection is established, and can work everything else out from this reference point.

Figure 2.20 Time division multiplexing.

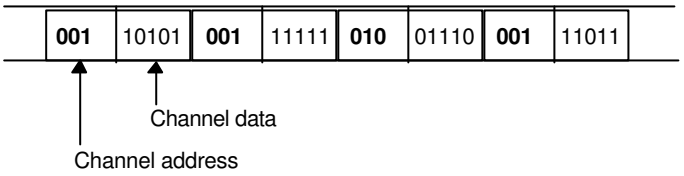


2.3.4. Concentration

In multiplexing, a predetermined bandwidth is reserved for each of the logical channels, the sum of which for all the logical channels equates the bandwidth of the line. In practice, none of the logical channels is fully utilized at all times by the equipment attached to them. Consequently, if the bandwidth of each of the channels could be dynamically adjusted according to its traffic, then some cost savings could be achieved by using a lower capacity line. For example, a 9600 bps line could be used to serve 10 times 2400 bps channels, assuming that no more than 4 channels are used at any one time.

This is made possible by a variation of TDM called **concentration**, where each channel is allocated a time slot only when it has data to transmit. Since, in this case time slots do not occur in a predetermined order, some means of indicating to which channel a time slot belongs is needed. This is easily achieved by having each time slot to contain two fields: the address of the channel to which it belongs, and the channel data (see Figure 2.21).

Figure 2.21 Time slots in concentration.



Concentration is a popular method for connecting a set of character-based terminals to a central computer. Line capacity requirements are greatly reduced due to the fact that terminals tend to be idle for most of their operating period.

2.4. Physical Layer Standards

The most commonly-used physical layer standards are those published by ISO, CCITT, IEEE, and EIA, many of which are inter-related. A large number of the existing standards deal with transmission over telephone lines. The CCITT V series of standards fall into this category and are by far the most-widely adopted.

Below we will look at two very popular standards for connecting DTEs and DCEs: the analog standard RS-232 and the digital standard X.21.

2.4.1. RS-232

RS-232 has dominated the computer industry as the most-widely used standard for physically connecting devices. It is an analog standard, defining the physical layer interface between a DTE and a DCE, and can support simplex, half-duplex, and full-duplex connections in synchronous as well as asynchronous mode. It originated in the late 1950s, and has been revised a number of times over the years. The latest revision, EIA-232-D, is based on CCITT's V.24 and V.28 standards and ISO's 2110 standard.

ISO 2110 defines the mechanical appearance of the RS-232 connectors (see Figure 2.22). The connector provides 25 pins for connecting the circuits derived from the V.24 standard, as summarized in Figure 2.23. The circuits are used for data transfer, conveying of control signals, and conveying of clocking signals for synchronization.

V.28 defines the electrical characteristics of RS-232. V.28 uses 5 to 15 Volts to represent binary value 0, and -5 to -15 Volts to represent binary value 1. It allows for connection distances of up to 20 meters and data rates of up to 20 kbps.

Figure 2.22 RS-232 connector (based on ISO 2110).

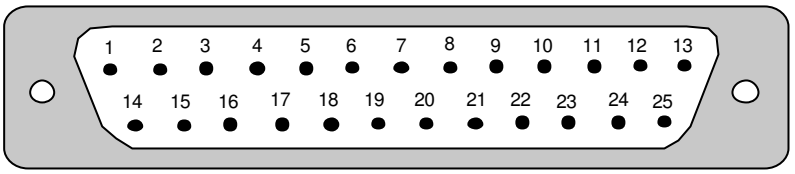
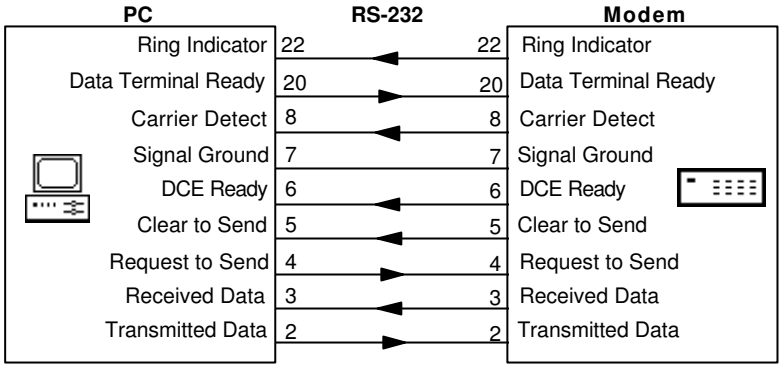


Figure 2.23 V.24 circuits.

Pin	Circuit	Direction	Description
1	AA	--	Protective ground shield / common return
2	BA	DTE to DCE	Transmitted Data
3	BB	DCE to DTE	Received Data
4	CA	DTE to DCE	Request to Send
5	CB	DCE to DTE	Clear to Send
6	CC	DCE to DTE	DCE Ready
7	AB	--	Signal Ground
8	CF	DCE to DTE	Carrier Detect
9	--	--	Reserved for testing
10	--	--	Reserved for testing
11	--	--	Unassigned
12	SCF	DCE to DTE	Secondary Carrier Detect
13	SCB	DCE to DTE	Secondary Clear to Send
14	SBA	DTE to DCE	Secondary Transmission Data
15	DB	DCE to DTE	Transmitter Signal Element Timing (transmitter clock)
16	SBB	DCE to DTE	Secondary Received Data
17	D	DCE to DTE	Receiver Signal Element Timing (receiver clock)
18	LL	--	Local Loopback
19	SCA	DTE to DCE	Secondary Request to Send
20	CD	DTE to DCE	Data Terminal Ready
21	RL/CG	DCE to DTE	Signal Quality Detector / Remote Loopback
22	CE	DTE to DCE	Ring Indicator
23	CH	DTE to DCE	Data Signal Rate Selector
23	CI	DCE to DTE	Data Signal Rate Selector
24	DA	DTE to DCE	Transmitter Signal Element Timing (transmitter clock)
25	TM	--	Test Mode

In most applications, only a few of the circuits specified by RS-232 are actually used. Figure 2.24 serves as an example. It illustrates how a PC may be connected to a modem.

Figure 2.24 Typical half-duplex connection using RS-232.



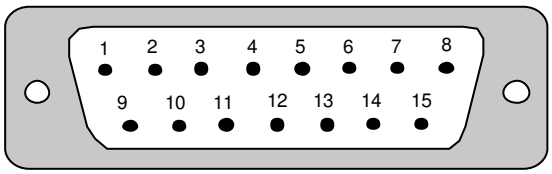
RS-232 has two important limitations which reduce its usefulness: it is not suitable for distances of more than about 50 meters, and it has a maximum bandwidth of 20 kbps. Other similar standards have been devised to overcome these limitations. For example, RS-449 and EIA-530 can both support data rates of up to 2 mbps over longer distances.

2.4.2. CCITT X.21

X.21 is a widely-accepted standard for interfacing a DTE to a DCE of a digital network. It can be used for connections of up to 1 km in length and data rates of up to 10 mbps (for distances less than 10 m). X.21 uses a connector based on the ISO 4903 standard (see Figure 2.25).

The connector provides 15 pins for connecting the circuits derived from the X.24 standard, as summarized in Figure 2.26. Unlike RS-232, the same transmit and receive circuits (T and R) are used for the exchange of control as well as data signals.

Figure 2.25 X.21 connector (based on ISO 4903).

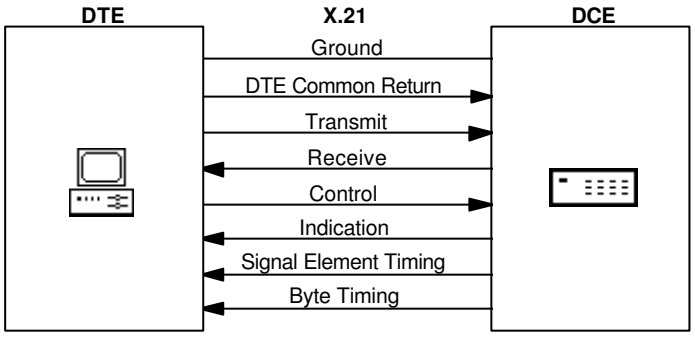


The electrical characteristics of X.21 are defined by V.10/X.26 or V11/X.27. V.10 uses 4 to 6 Volts to represent binary value 0, and -4 to -6 Volts to represent binary value 1. It allows for connection distances of up to 1 km. Figure 2.27 illustrates how a DTE and a DCE may be connected using X.21.

Figure 2.26 X.24 circuits.

Circuit	Direction	Description
G	--	Protective ground shield / Common Return
Ga	DTE to DCE	DTE Common Return
Gb	DCE to DTE	DCE Common Return
T	DTE to DCE	Transmit
R	DCE to DTE	Receive
C	DTE to DCE	Control
I	DCE to DTE	Indication
S	DCE to DTE	Signal Element Timing
B	DCE to DTE	Byte Timing
F	DCE to DTE	Frame Start Identification
X	DTE to DCE	DTE Signal Element Timing

Figure 2.27 Typical full-duplex connection using X.21.



X.21 *bis* is a variation of the X.21 standard with similarities to RS-232: it uses the V.24 circuits and is usually used with the 25-pin connector of ISO 2110.

2.5. Further Reading

Black (1988), Blahut (1990), Bic *et al* (1991), and Gitlin *et al* (1992) provide detailed descriptions of physical layer topics such as interfaces, coding, modulation, transmission, synchronization, error-handling, and standards. McClimans (1992) describes different transmission media and their properties. Stone (1982) describes detailed examples of (mainly RS-232) physical layer interfaces for microcomputers.

2.6. Summary

- Network equipment are classified into **DTE** (user equipment), **DCE** (connect DTE to network), and **DSE** (perform switching between DCEs).
- A connection may be of type **simplex**, **half-duplex**, or **full-duplex**.

- A signal may be **analog** (continuous variation of some property) or **digital** (sequence of binary values 0 and 1).
- Digital data is transmitted over analog lines using **modulation** and converted back to digital format using **demodulation**. These two functions are performed by a **modem**. Modulation method are classified into **AM**, **FM**, and **PM**.
- Converting an analog signal into digital is called **digitization** and is performed by a **codec**. PCM is a popular digitization method for voice signals.
- Transmission methods are classified into **synchronous** (clock-based) and **asynchronous**.
- Popular transmission media include: **copper wire**, **coaxial cable**, **optical fiber**, **radio**, and **infra-red**.
- Multiplexing methods are divided into **SDM** (multiple wires in a common enclosure), **FDM** (subdivision of the frequency bandwidth into logical channels), and **TDM** (allocation of time slots to each logical channel).
- **Concentration** is a variation of TDM where time slots are allocated on demand.
- **RS-232** is a popular analog standard for the physical interface between a DTE and a DCE.
- **X.21** is a popular digital standard for the physical interface between a DTE and a DCE.

2.7. Exercises

2.9 Describe the role and functions of DTEs, DCEs, DSEs, and name an example of each device type.

2.10 State the differences between an analog signal and a digital signal. Provide an example of either signal type.

2.11 Describe the differences between modulation and digitization. Name the devices that perform these functions.

2.12 Using a sample bit stream and a diagram, illustrate the difference between synchronous and asynchronous transmission.

2.13 Consider the problem of providing a 2 mbps physical connection between two LAN sites which are 10 kms apart and are located in the same city. Discuss the merits of using different types of transmission media for this purpose.

2.14 What is the purpose of multiplexing? Compare the strengths and weaknesses of FDM and TDM.

2.15 Describe how a 100 MHz line with a data rate of 200 mbps can be divided into 20 channels using FDM and TDM.

2.16 Describe the differences between RS-232 and X.21 standards. Provide an application example of either standard.

3. The Data Link Layer

This chapter looks at the data link layer of the OSI model. The data link layer transforms the logical communication channel provided by the physical layer into a reliable channel by splitting the data into frames which are subjected to error control and flow control procedures.

We will first look at various link protocol types, and then describe the constituent functions of link protocols, such as acknowledgment of frames, error checking, and flow control. These functions are embodied by a general technique called the sliding window protocol, which will be described next. Finally, we will discuss two popular data link standards: BSC and HDLC.

After completing this chapter you should be able to:

- Distinguish between different data link protocol types and know the characteristics of each type.
- Have a general understanding of the various data link protocol functions.
- Explain how the CRC error checking method works and how a CRC code is calculated.
- Understand the sliding window protocol and explain how it can be used for flow control.
- Describe the BSC character-oriented data link protocol, including its block format and functions.
- Describe the HDLC bit-oriented protocol, including its modes, frame format, different frame types, and subsets.

3.1 Link Protocol Types

Data link protocols are divided into two basic categories: synchronous and asynchronous. These are described below.

It is important not to confuse synchronization at the data link layer with synchronization at the physical layer. These two are distinct and essential in their own right: physical layer synchronization ensures that the transmitter and the receiver share a common clock signal so that bit boundaries can be detected; data link layer synchronization ensures that user data is not confused with control data.

3.1.1. Synchronous Protocols

Synchronous protocols operate by delimiting user data with unique bit patterns which the receiver uses to detect where the user data begins and where it ends. Synchronous protocols may be character-oriented or bit-oriented.

In a **character-oriented** protocol, user data consists of a sequence of characters and is delimited by two unique control characters (SYN and EOT). The biggest disadvantage of character-oriented protocols is that they are based on specific character sets (e.g., ASCII or EBSDIC) and are therefore character-set dependent.

In a **bit-oriented protocol**, no specific character set is assumed. The unit of transmission is a **frame**, which consists of user data (an arbitrary bit sequence), control data, address data, error checksum, and two delimiting bit patterns at either end of the frame. Figure 3.28 illustrates the frame structure for HDLC protocols (discussed later in this chapter).

Figure 3.28 HDLC frame structure.

Field	Description
01111110	Start flag: marks the beginning of the frame.
Address	Address of the host for which the frame is intended.
Control	Record frame type, frame sequence, flow control, etc.
Data	Contains the actual user data and is a bit sequence of arbitrary length.
Checksum	A checksum field for error detection.
01111110	End flag: marks the end of the frame.

The delimiting bit pattern used is 01111110 and is called a **flag**. To avoid this bit pattern occurring in user data, the transmitter inserts a 0 bit after every five

consecutive 1 bits it finds. This is called **bit stuffing**; its effect is canceled by the receiver, which removes every 0 bit that occurs after every five consecutive 1 bits.

Bit-oriented protocols are by comparison more recent than other protocols and have dominated the market. To acknowledge their importance, most of this chapter is devoted to the description of this class of protocols.

3.1.2. Asynchronous Protocols

Asynchronous protocols are character-oriented and operate by having the transmitter surround each character with a start and a stop bit. These two bits indicate to the receiver where the character starts and where it ends. The receiver extracts the user data by removing the start and stop bits.

3.1.3. Master-Slave Protocols

Master-slave protocols are used in situations where a primary station (such as a host) controls one or more secondary stations (such as peripheral devices). Communication between the master and its slaves is governed by a general technique called **polling**. Polling is typically used in a multidrop line configuration where, for example, a set of terminals are connected to a host via a shared communication line.

In its simplest form, polling works as follows. The master can send data in form of a message to any of the slaves. The message contains an address which uniquely identifies the slave for which it is intended. To receive data from the slaves, the master 'polls' each slave in turn by sending a message to it and asking if it has any data to send. The slave responds by either sending data or by sending a rejection message to indicate it has nothing to send.

3.1.4. Peer-to-Peer Protocols

In peer-to-peer protocols, all stations have similar status and communicate in a democratic fashion. The majority of network link protocols fall into this category. The **carrier-sense** protocol serves as an example, where all stations share a common communication channel. To transmit, a station monitors the channel for transmission by other stations and awaits its becoming free. Once the channel is free, the station starts transmitting. Collisions (when two stations attempt to transmit simultaneously) are resolved by requiring a transmitting station to monitor the channel at the same time and stop the transmission if it detects another station also transmitting. The collided stations then wait for some random period of time before attempting to retransmit.

3.2. Link Protocol Functions

This section describes various functions performed by link layer protocols.

3.2.1. Acknowledgments

Data link layer does not provide end-to-end accountability; this is a responsibility of higher layers. Instead, each station is responsible for ensuring that the data received by it is correctly passed to the next station. This is facilitated by the use of acknowledgments, of which there are two types:

- An acknowledgment (**ACK**) is sent by a receiver to a transmitter to indicate that a frame has been received seemingly intact.
- A negative acknowledgment (**NAK**) is sent by a receiver to a transmitter to indicate that a corrupt frame has been received, or that a frame is missing.

In character-based protocols, acknowledgments are implemented as control characters. In bit-oriented protocols, they are implemented as **acknowledgment frames**.

The control field of a frame can contain two sequence numbers: a **send sequence number** and a **receive sequence number**. The frames dispatched by a transmitter are sequentially numbered using the send sequence number. The receiver compares the send sequence number of a frame against the send sequence numbers of earlier frames to check for lost frames. A NAK frame uses the receive sequence number to identify a corrupted frame, while an ACK frame uses the receive sequence number to identify the next frame it expects to receive, implying that it has correctly received all frames with sequence numbers less than that number.

To improve the use of network bandwidth, an acknowledgment method known as **piggybacking** is often used. In piggybacking, instead of sending a separate acknowledgment frame, the receiver waits until it has data to send to the receiver and embeds the acknowledgment in that frame.

3.2.2. Timers

Timers provide a mechanism for placing a time limit on certain operations to be completed. A timer is set by a process and has a predefined expiry period. When the timer expires, the corresponding process is notified to take appropriate action.

Link protocols make use of a number of different timers, the exact number and function of which is protocol-dependent. However, two important timers (called T1 and T2) are used by most protocols.

T1 is set by a transmitter upon transmitting a frame. The transmitter expects to receive an acknowledgment frame before T1 expires. Otherwise, the transmitter resets T1 and retransmits the frame. The protocol allows a certain number of retries before the problem is propagated to a higher layer.

T2 is set by a receiver to ensure that an acknowledgment frame is sent to the transmitter before its T1 timer expires. This is intended to avoid the transmitter having

to retransmit frames because the receiver has been slow in acknowledging their receipt. When piggybacking is used, the receiver will send a *separate* acknowledgment frame if T2 expires.

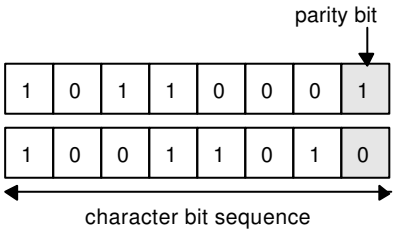
3.2.3. Error Checking

No physical medium can guarantee error-free transmission of data. Errors occur because of interference from the environment surrounding the transmission medium or because of artifacts introduced by the equipment or the medium itself. In case of copper wires, for example, thermal noise, electrical noise, and impulse noise are some of the most common causes.

The most important role of the data link layer is to provide error-free transmission. To do this, it uses an error checking scheme to detect errors, and overcomes them by retransmitting corrupted frames. Many different error detection methods have been devised. We will discuss two popular methods in this section: parity checking and cyclic redundancy code.

Parity checking is a simple error detection method used with character-oriented protocols. One bit in every character bit sequence is reserved for parity. This bit is set by the transmitter to 0 or 1 so that the total number of 1 bits in the character is always even (in case of even parity checking) or always odd (in case of odd parity checking). The receiver checks the parity bit of each character to see if it is as expected. Figure 3.29 illustrates the method for an even parity checking scheme.

Figure 3.29 Even parity error checking.



Parity checking is effective in detecting an odd number of bit errors in a character (single bit errors represent the most common case). If an even number of bits in a character are corrupted then this will go unnoticed, because the parity bit will seem correct.

Cyclic Redundancy Check (CRC) is a more sophisticated method used with bit-oriented protocols. It is also called the **polynomial code** — so named because it treats the single bits in a bit sequence as the coefficients of an imaginary polynomial. For example, the bit sequence 101001 represents the polynomial:

$$\mathbf{1}x^5 + \mathbf{0}x^4 + \mathbf{1}x^3 + \mathbf{0}x^2 + \mathbf{0}x^1 + \mathbf{1}x^0$$

$$= x^5 + x^3 + x^0$$

This polynomial is said to be of *order* 5, because its highest-order term is x^5.

CRC relies on the use of a **generator polynomial**, the choice of which is fairly arbitrary, except that the high and low order bits of its bit sequence must be 1 (e.g., 10111 is acceptable, but 0101 and 10110 are not) and that the transmitter and the receiver must use exactly the same generator polynomial.

CRC works as follows. Given a message (bit sequence) m represented by the polynomial $m(x)$, and a generator polynomial $g(x)$ of order k, m is appended with exactly k zero bits. The result is equivalent to the polynomial $p(x) = x^k m(x)$. $p(x)$ is divided by $g(x)$ and the remainder is added to $p(x)$ to produce the final polynomial $q(x)$:

$$q(x) = p(x) + (p(x) \textbf{ rem } g(x))$$

It follows that $q(x)$ is divisible by $g(x)$. The transmitter calculates q using m and g and then transmits q instead of m. If any of the bits in q are changed due to transmission errors, $q(x)$ has a very high probability of no longer being divisible by $g(x)$. This is checked by the receiver to detect transmission errors. If there are no errors, the receiver extracts m from q (by discarding the k low-order bits) and this will be identical to the original message.

The polynomial calculation procedures described above serve only as a conceptual model. In practice, equivalent binary operations are used instead. The binary operations are surprisingly simple, to the point that the whole process is usually implemented in hardware. Calculating q involves the following steps:

1. Given a bit sequence m and a generator polynomial bit sequence g of order k, append m with exactly k zero bits to produce bit sequence p. For example:

   m = 100101110101
   g = 10111 (k = 4)
   p = 1001011101010000

2. Divide p by g, using modulo 2 arithmetic, to produce a remainder r. Modulo 2 arithmetic operations are similar to binary operations, except that there are no borrows or carries. Addition and subtraction in modulo 2 are identical to the binary *exclusive or* operation. Continuing with our example:

```
                 11001000110   (quotient)
100101 | 1101011110010000
         100101
         0100001
          100101
          000100110
              100101
              0000110100
```

```
100101
0100010
 100101
    1110     (remainder)
```

$r = 1110$

3. Add r to p to produce q:

$$q = 1001011101010000 + 1110$$
$$= 1001011101011110$$

The effectiveness of CRC depends to a large extent on the choice of the generator polynomial. In practice, polynomials of order 16 are widely used. Hence 16 bits (two octets[3]) are needed for storing the remainder, which appears as the checksum field in the frame (see Figure 3.1). Such polynomials are more than 99% effective in catching a variety of errors.

3.2.4. Retransmission

Data link layer's approach to dealing with corrupted or lost frames is to retransmit them. Two different methods exist for this purpose. In the first method, called **Selective Reject**, upon encountering a faulty frame, the receiver requests the retransmission of that specific frame. Since additional frames may have followed the faulty frame, the receiver needs to be able to temporarily store these frames until it has received a corrected version of the faulty frame, so that frame order is maintained.

A simpler method, **Go-Back-N**, involves the transmitter requesting the retransmission of the faulty frame as well as all succeeding frames (i.e., all frames transmitted after the faulty frame). In other words, the receiver will not accept frames out of order and hence requires no additional means of storing them.

The advantage of Selective Reject over Go-Back-N is that it leads to better throughput, because only the erroneous frames are retransmitted. Go-Back-N, however, has the advantage of being simpler to implement and requiring less memory. To obtain the best of both worlds, sometimes a hybrid method is used. It behaves the same as Selective Reject, unless two consecutive frames are in error, in which case it behaves as in Go-Back-N.

3.2.5. Flow Control

The various stations in a network may operate at different speeds. One of the tasks of the data link layer is to ensure that slow devices are not swamped with data from fast devices. Flow control refers to the regulating of the rate of data flow from one

[3] An *octet* consists of eight consecutive bits. In most systems this is the same as a *byte*, but there are also systems, where a byte has seven, or nine, or some other number of bits.

device to another so that the receiver has enough time to consume the data in its receive buffer, before it overflows.

In character-oriented protocols, flow control is usually based on two control characters: XON and XOFF. When the receiver senses that it can no longer accept incoming data, it sends an XOFF character to the transmitter, which causes the latter to stop transmitting. Once the receiver has consumed enough of the data in its receive buffer so that it can receive more, it sends an XON character to the transmitter, causing it to resume transmission.

In bit-oriented protocols, flow control is handled through the use of ACK frames. Since the transmitter needs to keep a copy of its transmitted but yet unacknowledged frames in a buffer (in case they are corrupted and need to be retransmitted), the size of the buffer imposes an upper limit on the number of such frames. When necessary, the receiver can use this fact to slow down the transmitter by withholding ACK frames. The protocol described in the next section uses exactly such a strategy.

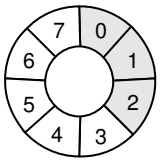
3.3. Sliding Window Protocol

Sliding window is a general protocol used with bit-oriented protocols. In this protocol, the transmitter maintains a variable, *S*, which denotes the sequence number of the next frame to be transmitted. Similarly, the receiver maintains a variable, *R*, which denotes the sequence number of the next frame it expects to receive. Both variables are restricted to a limited range of values (e.g., 0 through 7) by using modulo arithmetic (e.g., modulo 8).

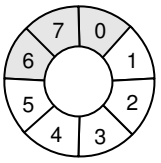
A **window** denotes a continuous subrange within the permitted range of values for the sequence numbers. For example, the ranges 0-3 and 6-1 both represent windows of size 3 (see Figure 3.30). Both the transmitter and the receiver have their own window:

- The **transmitter window** denotes the frames that have been transmitted but remain unacknowledged. This window can vary in size, from empty to the entire range. The transmitter must have enough buffer space to store the maximum possible number of unacknowledged frames.
- The **receiver window** denotes frames that are expected to be received. The receiver window size is fixed. A receiver window size of 1 means that frames must be received in transmission order. Larger window sizes allow the receiver to receive as many frames out of order. The receiver must have enough buffer space to store the maximum possible number of frames that can be received out of order.

Figure 3.30 Two windows of size 3.



Range 0-3



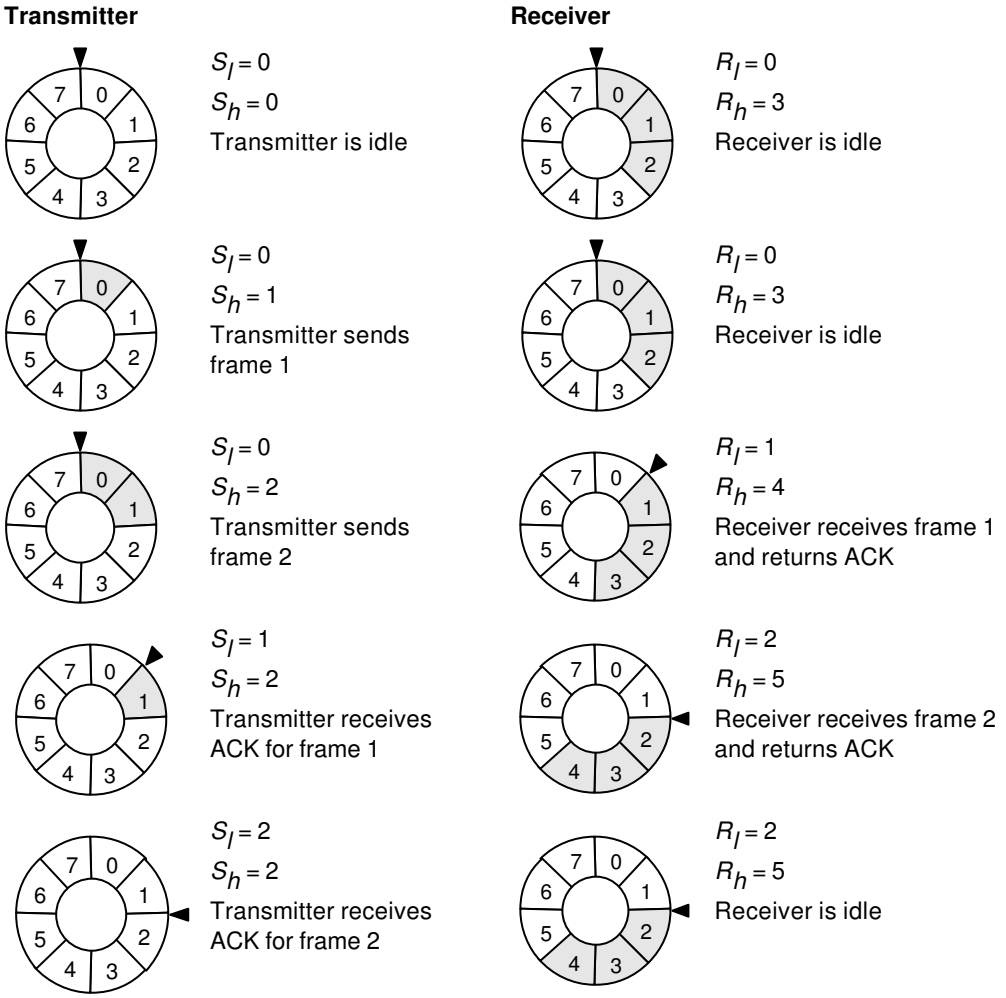
Range 6-1

The sliding window protocol operates as follows. When the transmitter sends a frame, it increments *S* (and the upper bound of its window). When the receiver receives a frame whose sequence number falls within its window, it increments *R* and sends an ACK to the transmitter. If the frame's sequence number matches any position other than the lower bound of the window, it notes the fact that the corresponding frame has now been received. If the frame's sequence number matches the lower bound of the window, the window is rotated clockwise by one position (or more positions if succeeding frames within the window have already been received).

When the transmitter receives an ACK for a transmitted frame, it increments the lower bound of its window. Figure 3.31 illustrates the protocol for a sliding window of size 3, with the sequence number range 0-7.

It should be now clear that flow control is straightforward in the sliding window protocol. If the receiver withholds ACK frames, the transmitter soon reaches its maximum window size and has to stop transmitting. Once it receives further ACK frames, it can reduce its window size and transmit more frames.

Figure 3.31 Sliding window of size 3, with sequence numbers 0-7.



3.4. Data Link Layer Standards

The most commonly-used data link layer standards are a number of *de facto* standards established by major computer manufacturers (e.g., BSC by IBM and DDCMP by DEC) and others published by ISO, CCITT, and IEEE. Below we will look at two popular standards; one character-oriented and one bit-oriented.

3.4.1. BSC

Binary Synchronous Control (BSC; also known as BISYNC) is a widely-used synchronous, character-oriented protocol devised by IBM in the 1960s for half-duplex communication.

To synchronize, the transmitter repeatedly sends a SYN character which the receiver looks for. Once the receiver has detected the SYN character, the two stations handshake to confirm that they are synchronized and can start exchanging data. Information is exchanged in character blocks. Figure 3.32 shows a sample block.

Figure 3.32 Sample BSC block.

Field	Description
SYN	Synchronization character.
SOH	Start-of-Header character.
Header	Used for control purposes.
STX	Start-of-Text character.
Data	Actual user data (of variable length and arbitrary content).
ETX	End-of-Text character.
Check	Error checking char(s).

A block starts with a SYN character. SOH marks the beginning of a header which contains additional control information, such as the block sequence number, the address of the transmitter, the address of the receiver, etc. STX and ETX mark the beginning and end of user data, which is an arbitrary sequence of characters. A redundancy check concludes the block.

Since control characters may also occur in user data, another control character, DLE (Data Link Escape), is used to avoid their being interpreted as control codes. If a control character occurs in user data, it is simply preceded by a DLE character. A literal DLE itself appears as two consecutive DLEs. The receiver treats a DLE as meaning 'accept the next character as literal'.

Error handling in BSC is fairly simple. If the receiver receives a corrupted block, it returns a NAK block which contains the sequence number of the offending block. The transmitter then retransmits that block. Parity checking is also used on individual characters.

3.4.2. HDLC

The High-level Data Link Control (HDLC) is a more recent bit-oriented protocol which enjoys widespread acceptance throughout the world. It is specified by ISO 3309, ISO 4335, and ISO 7809 standards, and supports half- as well as full-duplex

communication. Most vendors now tend to support this protocol (or one of its derivatives) in their networking products.

HDLC offers a master-slave arrangement in which the master station (in charge of the link) issues command frames and the slave stations reply with response frames. It is also possible for a station to assume a hybrid identity (master and slave) so that it can both issue commands and send responses. HDLC offers three modes of operation:

- **Normal Response Mode** (NRM). In this mode, a slave station is unable to initiate a transmission; it can only transmit in response to a command from the master station. The master station is responsible for managing the transmission. This mode is typically used for multipoint lines, where a central station (e.g., a host computer) polls a set of other stations (e.g., PCs, terminals, printers, etc.).
- **Asynchronous Response Mode** (ARM). In this mode, a slave station can initiate a transmission on its own accord. However, the master station is still responsible for managing the transmission. This mode is now largely obsolete.
- **Asynchronous Balanced Mode** (ABM). In this mode, all stations are of the hybrid form with equal status. Each station may transmit on its own accord. This mode is best-suited to point-to-point configurations.

The HDLC frame structure is shown in Figure 3.28. The *Address* field and the *Control* field are one or two octets each. The *Checksum* field is two octets and uses the CRC method. The *Data* field is of arbitrary length, and may be zero for some messages.

The structure of the control field depends on the type of frame, and may be one of the three shown in Figure 3.33. **Information frames** are used for exchange of user data between two stations. They may also be used for acknowledging receipt of data in a piggyback fashion. **Supervisory frames** are used for control purposes such as ACK and NAK frames and flow control. **Unnumbered frames** are used for link control functions such as link connection and disconnection.

The role of *Send* and *Receive sequence numbers* were discussed earlier in the chapter. The Poll/Final (**P/F**) bit is used in a HDLC master-slave arrangement. When set by the master station, it acts as a polling request to the slave station. When set by the slave station, it marks the last transmitted frame in response to a poll. The *Supervisory code* consists of two bits and determines the type of supervisory commands (maximum of four commands or four responses). The *Unnumbered code* consists of five bits and determines the type of unnumbered commands (maximum of 32 commands or 32 responses).

Figure 3.33 Frame types and associated control field structures.

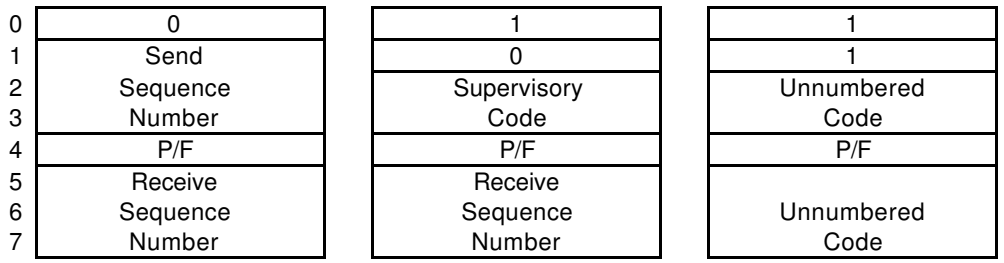


Figure 3.34 summarizes some of the supervisory and unnumbered commands and responses. The supervisory frames use the receive sequence number for acknowledgment and rejection, as discussed earlier.

There are a number of HDLC-related protocols, often referred to as HDLC subsets. These include:

- The **Link Access Procedure** (LAP) is based on the SARM command of HDLC. Under LAP, the transmitting station sends a SARM to the receiving station. The latter responds with a UA. At its discretion, the receiving station may interpret the receiving of a SARM command as a request for transmission in the opposite direction, in which case the roles are reversed.
- The **Link Access Protocol Balanced** (LAP-B) is an ABM subset of HDLC designed for use with X.25 (see Chapter 4). It is used for establishing a link between a DCE and a DTE. LAP-B does not support the SREJ command, and only supports a limited number of the unnumbered commands.
- The **Link Access Protocol, D channel** (LAP-D) is an HDLC subset designed for use with ISDN. It will be described in Chapter 11.
- The **Logical Link Control** (LLC) is an HDLC subset designed as a part of the IEEE 802 series of standards for use with LANs. It will be described in Chapter 10.

3.5. Further Reading

Martin and Leben (1988), Black (1989), Gitlin *et al* (1992), Halsall (1992), and Stallings (1994) provide comprehensive descriptions of the data link layer protocols and standards. LAN data link layers are discussed in Chapter 9. Chapter 11 describes the ISDN data link layer.

Figure 3.34 Sample HDLC commands and responses.

Mode	Command	Response	Description
Information	I	I	Used for exchange of user data.
Supervisory	RR	RR	Receive Ready. Station is ready to receive frames and acknowledged receipt of earlier frames.
	RNR	RNR	Receive Not Ready. Station is unable to receive frames, but acknowledged receipt of earlier frames.
	REJ	REJ	Reject. Rejects a frame according to the Go-Back-N scheme.
	SREJ	SREJ	Selective Reject. Selectively rejects a frame.
Unnumbered	SNRM		Set Normal Response Mode.
	SARM		Set Asynchronous Response Mode.
	SABM		Set Asynchronous Balanced Mode.
	SIM		Set Initialization Mode.
	UI	UI	Unnumbered Information. Used for exchange of data in unsequenced frames.
	UP		Unnumbered Poll. Unsequenced poll frame.
		UA	Unnumbered Acknowledgment. Unsequenced acknowledgment frame.
	DISC		Disconnect. Forces a slave station into disconnect mode.
		DM	Disconnect Mode. Used by a slave to indicate it is in disconnect mode.
	RSET		Reset. Enables stations to reset their send/receive sequence numbers.

3.6. Summary

- Data link protocols are divided into **synchronous** and **asynchronous**. A synchronous protocol is either **character-oriented** (user data is a sequence of delimited characters) or **bit-oriented** (user data is encapsulated by frames). Asynchronous protocols are all character-oriented.
- A data link protocol may use the **master-slave** model (based on the polling technique) or the **peer-to-peer** model (all stations have the same status).
- Link layer protocols comprise a number of functions: **acknowledgment** (to ensure that data is correctly passed on by stations), **timers** (to impose time limits on operations), **error checking** (to facilitate error-free transmission), **retransmission** (to overcome errors), and **flow control** (to manage the differences in device speeds).
- **Parity checking** is a simple error checking method used with character-oriented protocols. **CRC** is a more sophisticated error checking method used with bit-oriented protocols.

- The **sliding window** protocol is a widely-used flow control protocol which relies on the transmitter keeping track of transmitted but unacknowledged frames, and the receiver keeping track of frames that are expected to be received.
- The **BSC** (or **BISYNC**) protocol is a widely-used synchronous, character-oriented protocol devised by IBM.
- The **HDLC** is a widely-used bit-oriented protocol with a number of variations: LAP, LAP-B, LAP-D, and LLC. HDLC offers three modes of operation (NRM, ARM, and ABM). It transmits information in frames, of which there are three types: **information frames**, **supervisory frames**, and **unnumbered frames**.

3.7. Exercises

3.17 Explain the difference between synchronization at the physical layer and synchronization at the data link layer.

3.18 Describe what is meant by bit stuffing. Rewrite the following bit sequence after bit stuffing it: 01111100111011111111111010111.

3.19 Explain how the polling technique works. Describe an application for which polling may be appropriate.

3.20 Describe the role of frame sequence numbers and how they are used for acknowledgment purposes.

3.21 Assuming the generator polynomial $x^5 + x^3 + x^0$, what is the CRC code for the following bit sequence: 01101110011110111101111110101111?

3.22 Describe the role of T1 and T2 timers and how they can be used to handle a lost frame situation using the Selective Reject and G0-Back-N methods.

3.23 What problems are likely to occur in absence of a flow control scheme?

3.24 Using a diagram, illustrate how the sliding window protocol (with a maximum transmitter window size of 5 and receiver window size of 2) operates for the following scenario:

1. Transmitter sends frames 1 and 2

2. Transmitter receives ACK for frame 1
3. Transmitter sends frame 3
4. Transmitter receives ACK for frames 2 and 3

3.25 Describe the role of information, supervisory, and unnumbered frames in HDLC. Use a sequence diagram to illustrate the use of SREJ command in a data transfer situation.

4. The Network Layer

This chapter describes the network layer of the OSI model. The network layer handles the routing of data packets across the network, and defines the interface between a host and a network node.

We will first discuss the use of network primitives for defining network services. Then we will look at two switching methods and their use for routing. This is followed by a description of data packets and their handling by the network layer. We will then turn our attention to the problem of interconnecting two networks, and discuss the protocol sublayering provided for this purpose. Finally, we will look at four widely-accepted standards for networking as well as internetworking.

After completing this chapter you should be able to:

- Understand the nature of network services and use network primitives to describe network service scenarios.
- Describe how circuit switching works and appreciate its strengths and weaknesses.
- Describe how packet switching works and distinguish between the virtual circuit and datagram methods and their packet formats.
- Understand the three basic routing algorithms (flooding, static routing, and dynamic routing) and their characteristics.
- Appreciate the importance of congestion control.
- Appreciate the need for internetworking and the sublayers provided to support it.
- Have a basic knowledge of network layer standards X.25, X.75, IP, and ISO 8473.

4.1. Network Services

At the outset, the network layer provides a set of services to the transport layer. These services characterize the interface between the two layers and isolate the network details (lower three layers) from the network users (upper four layers). Network service content is therefore of considerable importance.

Network services are defined in terms of network service primitives. Figure 4.35 summarizes the primitives together with their possible types and parameters. Please note that these primitives only serve as a modeling tool and do not imply anything about how network services are implemented.

Addresses refer to Network Service Access Points (NSAP); these denote entities at the network layer that act as the interface to service users. *Quality Of Service* (QOS) denotes a set of parameters (such as error rate, delays, cost, failure likelihood, throughput) which collectively describe the quality of the network service. *User data* refers to actual user data provided by service users for transfer by the service provider. *Options* denote a variety of options that may be activated with some commands (e.g., whether data transfers can be confirmed).

Figure 4.35 Network service primitives.

Primitive	Types	Parameters	Purpose
N-CONNECT	request indicate response confirm	(addresses, options, quality of service, user data)	Used for initiating a connection (connections are always initiated by service users).
N-DATA	request indicate	(user data, confirm option)	Used for normal data transfer (with or without confirmation).
N-EXPEDITED-DATA	request indicate	(user data)	Used for high priority data transfer.
N-ACK-DATA	request indicate	()	Used for confirmation of data transfer.
N-RESET	request indicate response confirm	()	Used by the service user or provider for resetting the network service.
N-DISCONNECT	request indicate	(addresses, reason, user data)	Used by the service user or provider for disconnecting a connection.

Figure 4.36 Sample scenario of network services.

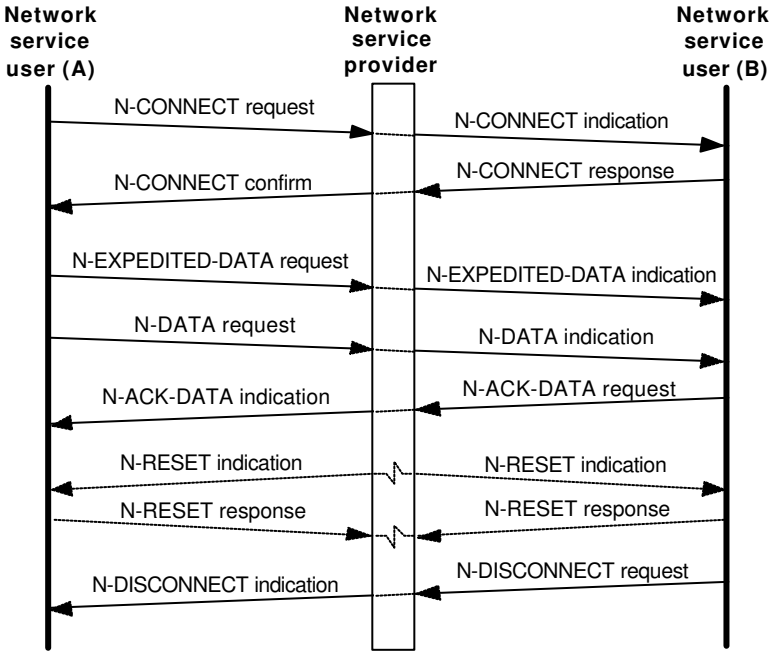


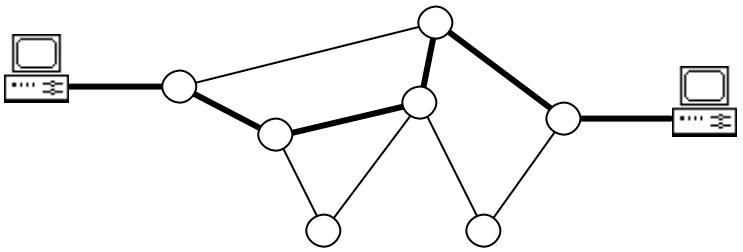
Figure 4.36 illustrates the use of the network services in a sample scenario. Network service user A first requests a connection, which is indicated to network service user B by the service provider. B responds to the request and the network confirms with A. Then A expedites some data which is indicated to B by the network. A normal data transfer from A to B follows which includes confirmation.

Then the network encounters an error and simultaneously sends reset indications to A and B, which both respond to. Finally, B requests a disconnection, which is indicated to A, and terminates the connection.

4.2. Switching Methods

Switching is the generic method for establishing a path for point-to-point communication in a network. It involves the nodes in the network utilizing their direct communication lines to other nodes so that a path is established in a piecewise fashion (see Figure 4.38). Each node has the capability to 'switch' to a neighboring node (i.e., a node to which it is directly connected) to further stretch the path until it is completed.

Figure 4.37 A 'switched' path.



One of the most important functions of the network layer is to employ the switching capability of the nodes in order to route messages across the network. There are two basic methods of switching: circuit switching and packet switching. These are separately discussed below.

4.2.1. Circuit Switching

In circuit switching, two communicating stations are connected by a *dedicated* communication path which consists of intermediate nodes in the network and the links that connect these nodes. What is significant about circuit switching is that the communication path remains intact for the duration of the connection, engaging the nodes and the links involved in the path for that period. (However, these nodes and links are typically capable of supporting many channels, so only a portion of their capacity is taken away by the circuit.)

Figure 4.38 shows a simple circuit switch which consists of a 3×3 matrix, capable of connecting any of its inlets (*a*, *b*, and *c*) to any of its outlets (*d*, *e*, and *f*). Each crosspoint appears as a circle. A hollow circle means that the crosspoint is *off* (i.e., the two crossing wires are not connected). A solid circles means that the crosspoint is *on* (i.e., the crossing wires are connected). The switch can support up to three simultaneous but independent connections. (Although we have used an

equal number of inlets and outlets here, in general, this need not be the case. Switches may also have more inlets than outlets, or more outlets than inlets.)

Figure 4.38 A simple circuit switch.

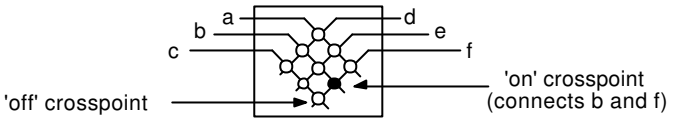
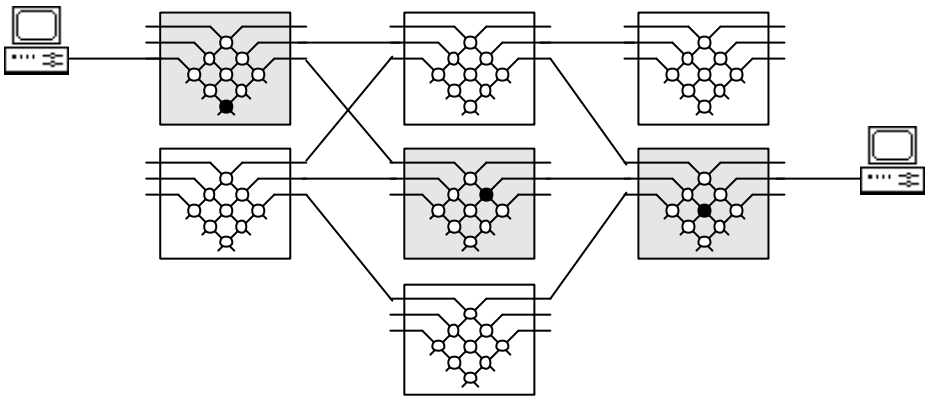


Figure 4.39 shows a simple circuit-switched network built using the switch in Figure 4.38. When the two hosts shown in the figure initiate a connection, the network determines a path through the intermediate switches and establishes a circuit which is maintained for the duration of the connection. When the hosts disconnect, the network releases the circuit.

Figure 4.39 Circuit switching.



Circuit switching relies on dedicated equipment especially built for the purpose, and is the dominant form of switching in telephone networks. Its main advantage lies in its predictable behavior: because it uses a dedicated circuit, it can offer a constant throughput with no noticeable delay in transfer of data. This property is important in telephone networks, where even a short delay in voice traffic can have disruptive effects.

Circuit switching's main weakness is its inflexibility in dealing with computer-oriented data. A circuit uses a fixed amount of bandwidth, regardless of whether it is used or not. In case of voice traffic, the bandwidth is usually well used because most of the time one of the two parties in a telephone conversation is speaking. However, computers behave differently; they tend to go through long silent periods followed by a sudden burst of data transfer. This leads to significant underutilization of circuit bandwidth.

Another disadvantage of circuit switching is that the network is only capable of supporting a limited number of simultaneous circuits. When this limit is reached, the

network blocks further attempts for connection until some of the existing circuits are released.

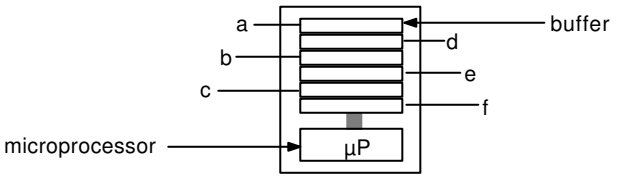
4.2.2. Packet Switching

Packet switching was designed to address the shortcomings of circuit switching in dealing with data communication. Unlike circuit switching where communication is continuous along a dedicated circuit, in packet switching, communication is discrete in form of packets. Each packet is of a limited size and can hold up to a certain number of octets of user data. Larger messages are broken into smaller chunks so that they can be fitted into packets. In addition to user data, each packet carries additional information (in form of a header) to enable the network to route it to its final destination.

A packet is handed over from node to node across the network. Each receiving node temporarily stores the packet, until the next node is ready to receive it, and then passes it onto the next node. This technique is called **store-and-forward** and overcomes one of the limitations of circuit switching. A packet-switched network has a much higher capacity for accepting further connections. Additional connections are usually not blocked but simply slow down existing connections, because they increase the overall number of packets in the network and hence increase the delivery time of each packet.

Figure 4.40 shows a simple packet switch with six I/O channels (*a* through *f*). Each channel has an associated buffer which it uses to store packets in transit. The operation of the switch is controlled by a microprocessor. A packet received on any of the channels can be passed onto any of the other channels by the microprocessor moving it to the corresponding buffer.

Figure 4.40 A simple packet switch.



Two variations of packet switching exist: virtual circuit and datagram.

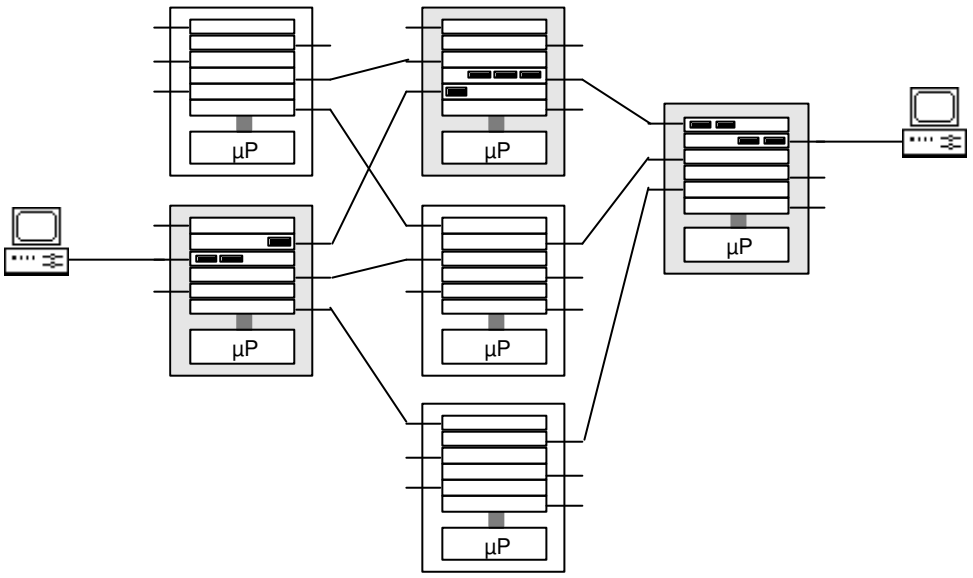
The **virtual circuit** method (also known as **connection-oriented**) is closer to circuit switching. Here a complete route is worked out prior to sending data packets. The route is established by sending a connection request packet along the route to the intended destination. This packet informs the intermediate nodes about the connection and the established route so that they will know how to route subsequent packets. The result is a circuit somewhat similar to those in circuit switching, except

that it uses packets as its basic unit of communication. Hence it is called a virtual circuit.

Each packet carries a virtual circuit identifier which enables a node to determine to which virtual circuit it belongs and hence how it should be handled. (The virtual circuit identifier is essential because multiple virtual circuits may pass through the same node at the same time.) Because the route is fixed for the duration of the call, the nodes spend no effort in determining how to route packets.

Figure 4.41 illustrates the virtual circuit method using the switch in Figure 4.40. When the two hosts initiate a connection, the network layer establishes a virtual circuit (denoted by shaded switches) which is maintained for the duration of the connection. When the hosts disconnect, the network layer releases the circuit. The packets in transit are displayed as dark boxes within the buffers. These packets travel only along the designated virtual circuit.

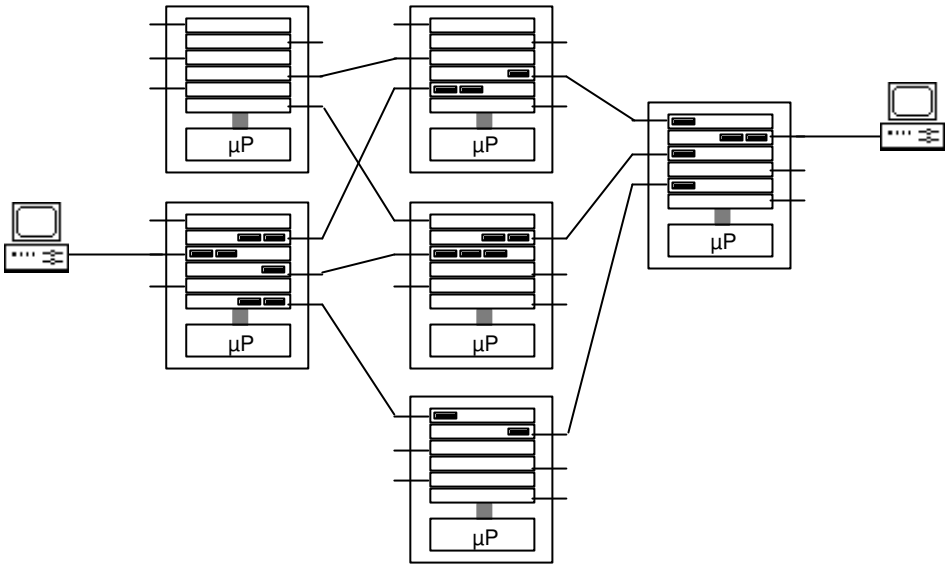
Figure 4.41 Packet switching with virtual circuits.



The **datagram** method (also known as **connectionless**) does not rely on a pre-established route, instead each packet is treated independently. Therefore, it is possible for different packets to travel along different routes in the network to reach the same final destination. As a result, packets may arrive out of order, or even never arrive (due to node failure). It is up to the network user to deal with lost packets, and to rearrange packets to their original order. Because of the absence of a pre-established circuit, each packet must carry enough information in its header to enable the nodes to route it correctly.

Figure 442 illustrates the datagram method. Note how the packets exercise different routes.

Figure 4.42 Packet switching with datagrams.



The advantage of the datagram approach is that because there is no circuit, congestion and faulty nodes can be avoided by choosing a different route. Also, connections can be established more quickly because of reduced overheads. This makes datagrams better suited than virtual circuits for brief connections. For example, database transactions in banking systems are of this nature, where each transaction involves only a few packets.

The advantage of the virtual circuit approach is that because no separate routing is required for each packet, they are likely to reach their destination more quickly; this leads to improved throughput. Furthermore, packets always arrive in order. Virtual circuits are better suited to long connections that involve the transfer of large amounts of data (e.g., transfer of large files).

Because packet switching is the more dominant form of switching for data communication, we will focus our attention on this form of switching from now on. Also, unless indicated otherwise, by *switching* we will mean *packet switching* in the rest of this book.

4.3. Packet Handling

In this section, we will look at packets, their structure, and how they are handled by the network layer.

4.3.1. Packet Structure

The general structure of a virtual circuit packet is shown in Figures 4.43. The contents of the fields with darker shading vary according to the *Packet Type*, and may be altogether absent from some packets.

Figure 4.43 General virtual circuit packet structure.

Field	Description
Packet Format	Determines the internal format of the packet.
Packet Type	Identifies the type of the packet.
Virtual Circuit ID	Identifies the virtual circuit for routing the packet.
Facilities	Parameters such as packet size, window size, etc.
Service Parameters	Parameters for the network service to which this packet belongs (see Section 4.1).
User Data	Actual network user data.

Figure 4.44 shows the general structure of a datagram packet. Because packet headers may vary in length, the *Header Length* field is needed to indicate where *User Data* starts. Each packet is assigned a limited lifetime denoted by the *Lifetime* field. This is an integer quantity and is decreased in value by the nodes that handle the packet. When this value reaches zero, the packet is discarded. This is intended as a measure against congestion by packets that aimlessly circulate the network.

Figure 4.44 General datagram packet structure.

Field	Description
Header Length	Length of the packet header in bits or octets.
Source Address	Network address of the source end.
Destination Address	Network address of the destination end.
Lifetime	Packet lifetime indicator (to avoid packets living forever).
Options	Various options such as priority, security, etc.
User Data	Actual network user data.

4.3.2. Routing

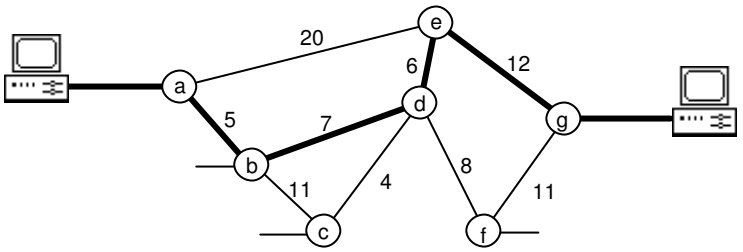
Routing is the task of selecting a path for the transport of packets across the network, and is one of the most important functions of the network layer. Routing is generally viewed as an optimization problem with the objective of choosing an optimal path according to certain criteria:

- Transmission cost (measured in terms of tied up network resources).

- Transmission delay (measured as the delay involved in delivering each packet).
- Throughput (measured as the total number of packets delivered per unit of time).

The overall cost depends on all these three, and an optimal route is one that minimizes the overall cost. This can be represented by a weighted network, where an abstract cost figure is associated with each link, as illustrated in Figure 4.45. The cost of a route is the sum of the cost of its links.

Figure 4.45 A weighted network.



There are three classes of routing algorithms:

- Flooding
- Static Routing
- Dynamic Routing

Of these, only dynamic routing makes any serious optimization attempts.

In **flooding**, every possible path between the source and the destination station are exercised. Each node, upon receiving a packet, forwards copies of it to all its neighboring nodes (except the one from which it received the packet). Flooding is a highly robust technique since it offers the best chance of at least one packet copy reaching the destination. Its major disadvantage, however, is that it quickly congests the network. To avoid packets being indefinitely copied, each packet is assigned a limited lifetime which when expired will cause it to be destroyed (see Section 4.3.1). Because of its limitations, use of flooding is confined to specialized applications that require very high levels of robustness (e.g., military networks). Flooding is only suited to the datagram approach.

In **static routing**, a fixed routing directory is used to guide the selection of a route which remains unchanged for the duration of the connection. The directory consists of a table which for each node pair (p,q) in the network suggests a partial path by nominating the first intermediate node, r, along the path. This should be interpreted as 'to get from p to q, first go to r'. The path can then be continued by looking at the entry for the pair (r, q), etc., until q is reached.

A sample route directory is shown in Figure 4.46. The route directory is tied to the network topology and remains unchanged unless the network topology changes. The route directory may be stored in a central location or distributed amongst the

nodes (each node requires its corresponding row only). The directory should remain accessible to network administrators for manual updates.

Figure 4.46 Static route directory for the network in Figure 4.45.

		To						
		a	**b**	**c**	**d**	**e**	**f**	**g**
From	**a**	-	b	b	b	b	b	b
	b	a	-	c	d	d	d	d
	c	b	b	-	d	d	d	d
	d	b	b	c	-	e	f	e
	e	d	d	d	d	-	d	g
	f	d	d	d	d	d	-	g
	g	e	e	e	e	e	f	-

The advantages of static routing are its simplicity and ease of implementation. Its disadvantage is lack of flexibility in face of possible variations within the network. Static routing can be used with both the datagram and the virtual circuit approach.

Although static routing takes some form of cost into account, these cost measures are fixed and predetermined. Variations in traffic load have no impact on route selection, which makes this method unsuitable for large networks with significant traffic fluctuations. For small networks with relatively constant traffic, however, static routing can be very effective.

Dynamic routing attempts to overcome the limitations of static routing by taking network variations into account when selecting a route. Each node maintains a route directory which describes the cost associated with reaching any other node via a neighboring node. A sample directory is shown in Figure 4.47.

Figure 4.47 Dynamic route directory for node 'd' in Figure 4.45.

	To						
	a	**b**	**c**	**d**	**e**	**f**	**g**
From d	b	b	c	-	e	f	e
Cost	7	7	4	0	6	8	6

The nodes periodically calculate estimates for the costs of the links to their neighboring nodes according to the statistical data that they have collected (queue lengths, packet delays, traffic load, etc.), and share these estimates with other nodes. This enables the nodes to update their route directories in an objective manner so that they reflect the current state of the network.

To select a route between two stations, dynamic routing employs a graph optimization algorithm to find an optimal (or close to optimal) path as described earlier in this section.

The advantage of dynamic routing is its potential to improve performance and reduce congestion by choosing more optimal routes. Its disadvantage is its inherent complexity. Nevertheless, dynamic routing algorithms enjoy widespread use because they have proved to be generally more effective than other algorithms. Like static routing, dynamic routing can be used with both the datagram and the virtual circuit approach.

4.3.3. Congestion Control

A network has a certain carrying capacity, denoted by the maximum number of packets that it can hold at any point in time. When this limit is approached, considerable delays are experienced in packet delivery, and the network is said to be **congested**. Congestion can occur in all types of networks. Uncontrolled congestion can lead to outright network failure.

At the node level, congestion manifests itself as packet buffers that have approached their full capacity. This happens because the node is receiving more packets than it is passing on to the next nodes, which in turn are presumably unable to receive more packets due to their buffers being full. When this situation occurs, the chances are that it will progressively get worse.

The best way to deal with congestion is to avoid it. This is facilitated by putting in place measures that prevent buffer overflow. These measures include the following:

- *Reducing the load on a node by disposing packets.* As mentioned in earlier sections, packet disposal can be guided by a *lifetime* indicator which is eroded by the nodes that handle the packet. More blatant ways of disposing packets may also be employed. For example, a node that receives a packet for which it has almost no buffer space may destroy it immediately.
- *Reducing the traffic destined for a heavily-utilized link*. Nodes can monitor the traffic on their outgoing links and ask the source host to reduce the transmission rate when they feel that a link is approaching its capacity. The request can be put to the source host using a special packet.
- *Imposing a limit on the total number of packets in the network*. This approach requires some means of keeping a count of the packets in the network. Furthermore, the nodes will have to communicate to ensure that the count is kept up-to-date. Although, this approach ensures that the network cannot be overloaded with too many packets, it does not prevent an individual node from being overloaded.

4.3.4. Error Handling

The extent to which the network layer attempts to deal with errors is largely dependent on the type of service being offered. The datagram service offers little in

this respect. If packets do not arrive due to being corrupted or lost, or if they arrive out of order, it is the responsibility of the network user to deal with these problems. The virtual circuit service, however, handles these problems transparently.

The usual approach to dealing with corrupt or lost packets is to request retransmission. This issue was covered in earlier discussions.

It is not uncommon for network protocols to have additional measures built into them to enable them to better deal with errors. For example, some protocols also use a CRC check on the packet header. To communicate the cause of detected errors, some protocols use diagnostic packets. The subject area remains largely protocol dependent.

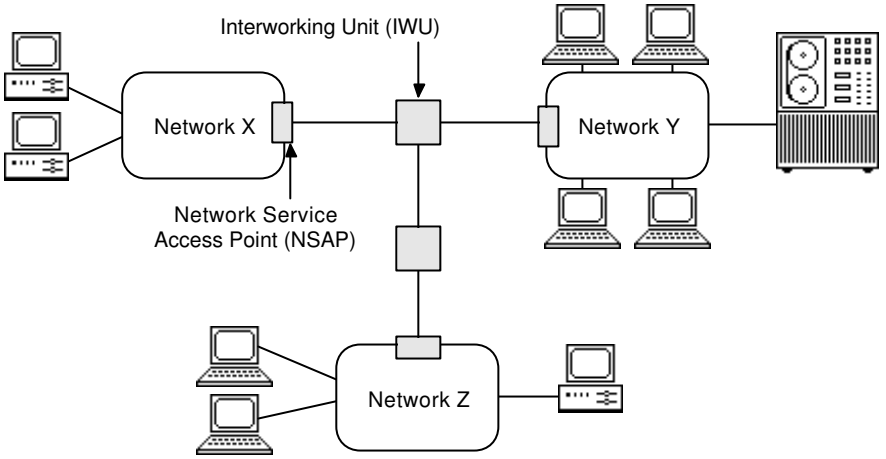
4.4. Internetworking

It is an inevitable aspect of communication networks that because of business, organizational, political, and historical reasons, there exist many different types of networks throughout the world. More and more, it is becoming desirable, if not necessary, to interconnect these networks. Take, for example, the LANs operated by many universities around the world. These usually differ in many respects, but there has been a strong need to interconnect them so that the international academic and research community can freely exchange information.

The problem of interconnecting a set of independent networks is called **internetworking**, and is illustrated by Figure 4.48. Each of the participating networks is referred to as a **subnetwork** (or **subnet**). The role of the **Interworking Units** (IWU) is to carry out protocol conversion between the subnets. IWUs are used for interconnecting networks that use the same architecture but employ different protocols. Because of this, they operate at the network layer and are commonly referred to as **routers**.

Another type of protocol converter is a **gateway**. Unlike a router, it is used to interconnect networks of different architectures. Gateways operate at layers above the network layer and usually encompass all the layers.

Figure 4.48 Internetworking.



4.4.1. Network Sublayers

To better address the internetworking problem, the network layer is further subdivided into three sublayers (see Figure 4.49). This subdivision is intended to isolate the functions of internetworking from the functions of intranetworking. The latter is handled by the bottom sublayer, and the former by the upper two sublayers. IWUs typically need to implement all three sublayers.

Figure 4.49 Network sublayers.

Layer	Name	Main Function
3.3	Subnet Independence Convergence Layer	Deals with internetworking services (i.e., routing and data transfer between the subnets).
3.2	Subnet Dependence Convergence Layer	Deals with service mapping between the subnets (i.e., protocol conversion, address mapping).
3.1	Subnet Access Layer	Deals with the services of a single subnet (i.e., subnet operation, access, addressing, routing, etc.).

Depending on the packet switching mode employed by the subnets, three situations are possible:

1. *All subnets use virtual circuits.* In this case a network level virtual circuit protocol is used for internetworking. The IWUs perform the protocol conversion between the subnets' network level protocols and the internetworking protocol.

2. *All subnets use datagrams.* In this case a network level datagram protocol is used for internetworking. The IWUs perform the protocol conversion between the subnets' network level protocols and the internetworking protocol.

3. *Some subnets use virtual circuits, some datagrams.* This situation has no easy solution. A practical way of addressing it is to use either a network level virtual circuit protocol or a network level datagram protocol for internetworking and require all participating subnets and IWUs that do not support that mode to implement both modes.

Given that the subnets may use different network architectures and different protocols, many incompatibilities need to be overcome by the IWUs and gateways, including the following:

- Different types of service and network user interfaces
- Different message formats
- Different addressing schemes
- Different packet switching modes
- Different routing methods
- Different error handling methods
- Different security measures

4.5. Network Layer Standards

Network services (discussed in Section 4.1) are defined by the ISO 8348 and CCITT X.213 standards. ISO 8880-2 and ISO 8880-3, respectively, provide standards for the provision and support of network services for the virtual circuit and datagram models. Internetworking and the internal breakdown of the network layer into sublayers is covered by the ISO 8648 standard. CCITT X.121 provides a numbering plan for data network addressing on an international basis.

There are many other standards pertaining to the network layer. Below we will look at three influential networking and internetworking standards: X.25, X.75, and ISO 8473.

4.5.1. CCITT X.25

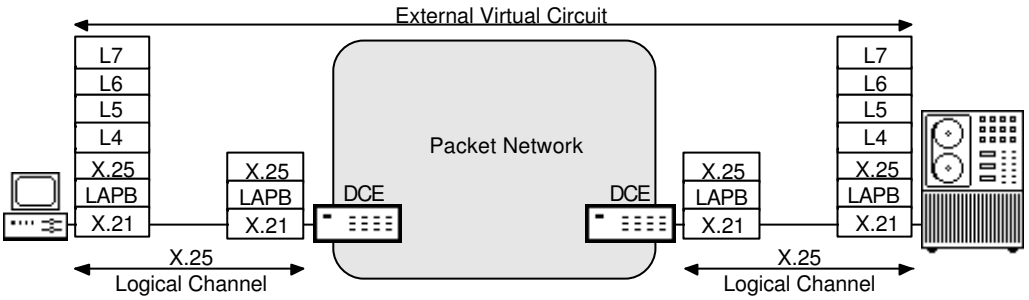
X.25 is probably the most widely-known network layer standard. It has been in existence since 1974 and has been revised a number of times. X.25 enjoys widespread use in the industry for connecting DTEs to packet networks, and has found its way to many network products.

X.25 encompasses the bottom three layers of the OSI model. For its bottom two layers, however, it simply refers to other standards. It recommends X.21 and X.21 *bis* (see Chapter 2) for its physical layer, and LAP-B (see Chapter 3) for its

data link layer. Many vendors have also used other standards instead of these in their products.

Strictly speaking, X.25 provides no routing or switching functions. It simply provides an interface between DTEs and network DCEs. As far as X.25 is concerned, the network is an abstract entity, capable of providing end-to-end connections. Figure 4.50 illustrates the role of X.25 in a typical end-to-end connection. The two logical channels between DTEs and DCEs at either end are joined through the packet network to complete an external virtual circuit. X.25 makes no assumptions about the internal mode of the packet network. This may be virtual circuit or datagram.

Figure 4.50 X.25 scope and layers.



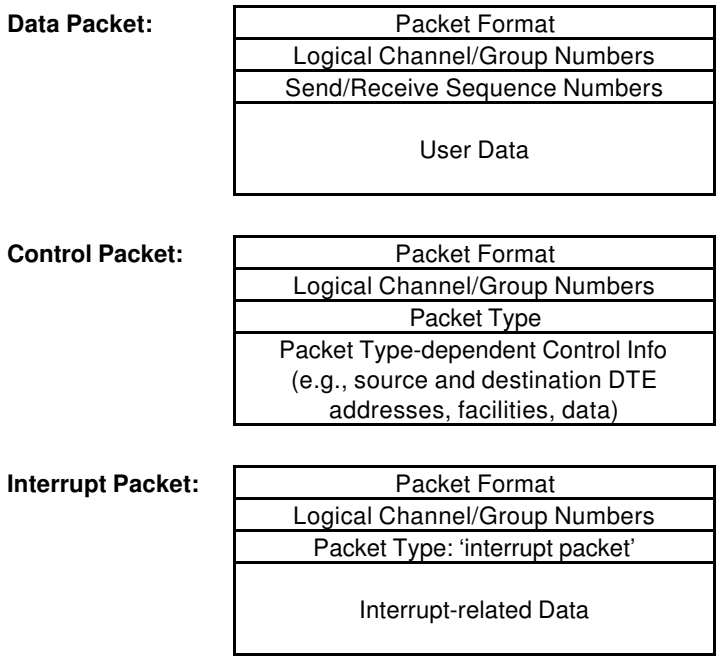
X.25 provides three types of external virtual circuits which, although similar in many respects, serve different applications:

- **Virtual Call** (VC). Virtual calls consist of three sequential phases: (i) *call setup* which involves the caller and the callee exchanging call request and call accept packets, (ii) *data transfer* which involves the two ends exchanging data packets, and (iii) *call clear* which involves the exchanging of call clear packets.
- **Permanent Virtual Circuit** (PVC). The virtual circuit is permanently assigned by the network, hence the DTEs are guaranteed that a connection will always be available. There is no need for call setup or call clearing, and the data transfer is as in VC.
- **Fast Select Call**. Fast select calls are similar to virtual calls, except that the call setup and the call clear packets may contain up to 128 octets of user data. A variation of this type of circuit provides for the immediate clearing of the call after the first phase. Fast select calls in effect emulate a datagram facility, which is useful for situations where the communication between the two DTEs is limited to a couple of short messages.

X.25 provides for sequencing of packets using the send and receive sequence numbers. Flow control is based on the sliding window protocol. Error control uses the Go-Back-N method. All these are used in a similar fashion to their use in HDLC.

There are three general categories of X.25 packet formats: **data** packets, **control** packets, and **interrupt** packets (see Figure 4.51). Most packet types fall into the control packet category. Interrupt packets are used for implementing the network layer's expedited data service. These are assigned a higher priority, can carry up to 32 octets of user data, and bypass the normal flow control procedures. To avoid flooding the network, only one unconfirmed interrupt packet is allowed at a time.

Figure 4.51 X.25 packet format categories.



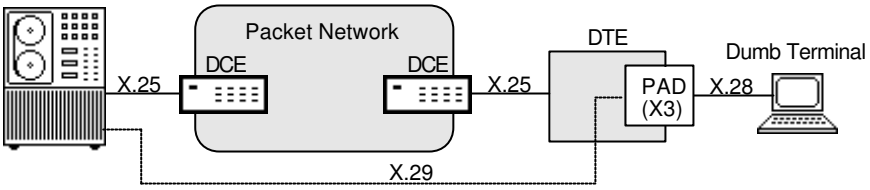
X.25 packets support all the network services described in section 4.1, as well as three other services: diagnostic, registration, and restart. *Diagnostic* packets are used to communicate additional error-related information between X.25 stations. *Registration* packets enable X.25 subscribers to optimize the network services assigned to them by conveying appropriate requests to the network. *Restart* packets reset all the logical channels and all the virtual circuits; they are useful for dealing with serious network failures.

Other protocols often associated with X.25 are: X.3, X.28, and X.29 (collectively referred to as triple-X). These define the functions and the interfaces for a Packet Assembler/Disassembler (PAD). The role of the PAD is to allow conventional character-based terminals to be connected to X.25. It assembles the characters from a terminal into a packet ready for transmission and disassembles an incoming packet into a character sequence for terminal consumption. X.3 defines the PAD parameters. X.28 defines the interface between the PAD and the terminal.

X.29 defines the interface between the PAD and the remote host. Figure 4.52 illustrates.

It is worth noting that the triple-X have no clear place in the OSI model. Also, their role will diminish over time as X.25 terminals become more widespread.

Figure 4.52 The triple-X protocols.



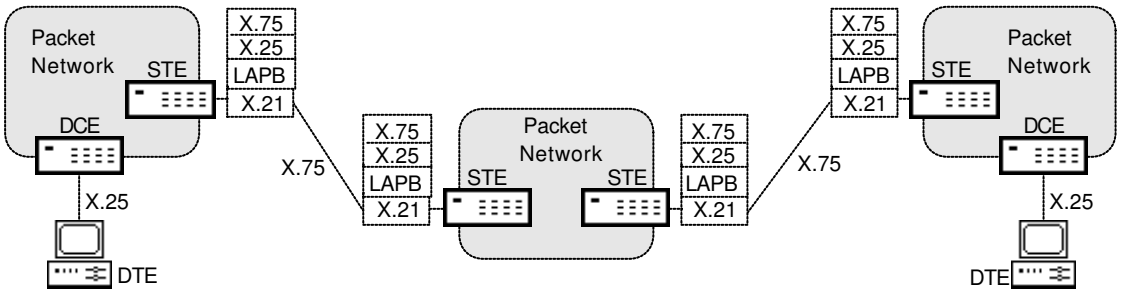
4.5.2. CCITT X.75

X.25 is a subnet standard; it operates within a single network. A complementary standard, X.75, has been developed to support the internetworking of X.25 subnets, although it can also be used to interconnect subnets based on other protocols.

Like X.25, X.75 consists of three layers: physical, data link, and network. For its physical and data link layers, it may use standards similar to those used by X.25 (e.g., X.21 and LAP-B). At the network level, X.75 accounts for the upper two sublayers, sitting on top of X.25 which accounts for the lower sublayer (see Figure 4.49).

Operationally, X.75 provides a virtual circuit end-to-end connection between two DTEs on separate subnets, by interconnecting the latter and any intermediate subnets (see Figure 4.53). The interconnections are facilitated by **Signaling Terminals** (STEs) which act as partial IWUs or routers, each implementing the X.75 protocol stack. The two DTEs are connected by a virtual circuit which is comprised of a number of 'smaller' virtual circuits within and in between the subnets.

Figure 4.53 Internetworking with X.75.



X.75 packets implement the international packet-switched services for internetworking defined by the X.75 standard. In many respects, they are similar to

X.25 packets, except that they are used for virtual circuits between STEs as opposed to virtual circuits between DTEs and DCEs.

4.5.3. IP

The Internet Protocol (IP) is a connectionless datagram protocol developed by the US Department of Defense Advanced Research Projects Agency (DARPA). It currently enjoys widespread use around the world. An IP network consists of a set of subnets and a set of intermediate stations which act as gateways. The subnets may have totally different characteristics. The gateways perform routing functions as well as handling protocol conversion between the subnets.

IP datagrams are referred to as **Internet Protocol Data Units**, or IPDUs (see Figure 4.54). This datagram is transported as user data within the packets of the subnets. When a packet is about to cross a subnet boundary, the intervening gateway extracts the datagram and encapsulates it into a new packet for use of the next subnet. Gateways and subnet stations maintain a (static or dynamic) routing table which depicts the next gateway the IPDU should be forwarded to.

The interface for IP service users is quite simple, and consists of two primitives: *send* and *deliver*. An IP service user transmits data by issuing a *send* command which contains various IPDU fields as parameters. When the IPDU is delivered to its final destination, the receiving station is issued with a *deliver* command which contains the original data.

IP provides status reporting, diagnostics, and flow control capabilities through its Internet Control Message PDUs (ICMPDUs). These are special PDUs used for exchanging status and diagnostics information between the stations.

Figure 4.54 IPDU structure.

Field	Description
Version	Version of the IP.
Header Length	Length of the header fields in octets.
Services	Denotes the type of IP services required.
Total Length	Total length of the IPDU.
Data Unit ID	Denotes the first segment of a segmented PDU.
Flags	For segmentation and error reporting.
Segment Offset	Denotes the relative position of IPDU within PDU.
Lifetime	IPDU lifetime.
Checksum	Checksum for the IPDU header.
Addresses	Source and destination NSAP addresses.
Options	For source routing, route recording, QOS, etc.
Data	Actual user data.

4.5.4. ISO 8473

ISO 8473 is a standardization and simplification of IP. Despite its many similarities to IP, the two protocols remain functionally incompatible. ISO 8473 is also referred to as the **ISO Internet Protocol** (IP) and the **Connectionless Network Protocol** (CLNP).

The CLNP network service is embodied by a single service primitive, N-UNITDATA, of which there are two types (see Figure 4.55). CLNP makes no assumptions about the underlying subnetwork architecture, except that it should be able to support point-to-point transfer of data units of at least 512 octets each. For packets larger than supported by the subnet, CLNP provides the necessary segmentation and re-assembly functions. Routing can be guided by the transmitter by including a routing option in the packet, but is otherwise determined by the internetworking nodes.

Figure 4.55 CLNP service primitive.

Primitive	Types	Parameters	Purpose
N-UNITDATA	request indicate	(source address, destination address, quality of service, user data)	Used for data transfer between two NSAPs. Each packet is fully addressed and routed independently.

The general structure of CLNP IPDUs is shown in Figure 4.56. These are used to represent user **Network Service Data Units** (NSDUs). It follows that, depending on its size, an NSDU may be represented by a single IPDU or segmented into multiple IPDUs.

Figure 4.56 ISO 8473 IPDU structure.

Field	Description
Protocol ID	Denotes the IPDU protocol (i.e., ISO 8473).
Header Length	Length of the header fields in octets.
Version Number	Version of the ISO 8473 protocol.
Lifetime	IPDU lifetime.
Flags	For segmentation and error reporting.
Type	Denotes the IPDU type.
Segment Length	Total length of the segment in octets.
Checksum	Checksum for the IPDU header.
Addresses	Source and destination NSAP addresses.
Data Unit ID	Denotes the first segment of a segmented NSDU.
Segment Offset	Denotes the relative position of IPDU within NSDU.
Total Length	Total length of the NSDU.
Options	For source routing, route recording, QOS, etc.

Data	Actual user data.

The *Protocol ID* field uniquely identifies the IPDU from other network packets. Segmentation is facilitated by the *Segment Length* and *Segment Offset* fields. The former is set to the length of the segment for this IPDU; the latter denotes the relative position of this segment from the beginning of the original NSDU. The *Total Length* field denotes the total length of the original NSDU.

Corrupt and expired IPDUs are discarded immediately. This causes an error report being generated by the respective station, which is then returned back to the originating station.

ISO 8473 can be used in a variety of arrangements. It can be used on top of X.25, or even directly on top of LAP-B.

4.6. Further Reading

General descriptions of the network layer, its sublayers, protocols, and standards are provided by Freeman (1989), Tanenbaum (1989), Stamper (1991), and Stallings (1994). Detailed accounts of internetworking, gateways, and routers can be found in White (1992) and Perlman (1992). Dhas and Konangi (1986) and Schlar (1990) describe X.25 and related standards. Black (1992), Feit (1992), Comer (1993), and Piscitello and Chapin (1993) describe IP in detail. Lynch and Rose (1993) is a handy reference on the Internet. Stallings (1990 and 1993b) are useful sources on networking standards.

4.7. Summary

- The generic method for establishing a path for point-to-point communication in a network is called **switching**. There are two general switching methods: circuit switching and packet switching.
- In **circuit switching** two communicating stations are connected by a dedicated communication path.
- In **packet switching** communication is discrete in form of packets. The packets are handled by the intermediate nodes in a store-and-forward fashion. Packet siwtching is either based on **virtual circuits** or on **datagrams**.
- The task of selecting a path for the transport of packets across the network is called **routing**. The three classes of routing algorithms are: **flooding**, **static routing**, and **dynamic routing**.

- **Congestion control** measures ensure that a network's performance is not eroded due to overload conditions.
- **Internetworking** facilitates the communication between two stations situated on different networks (subnets). This requires protocol conversions and is handled by **Interworking Units** (IWUs).
- A **router** is a protocol converter which interconnects two networks of the same architecture. A **gateway** is a protocol converter which interconnects two networks of different architectures.
- The network layer is subdivided into three sublayers to facilitate internetworking: the **subnet access** sublayer deals with the services of a single subnet, the **subnet dependence convergence** sublayer deals with service mapping between the subnets, the **subnet independence convergence** sublayer deal with internetworking services.
- **X.25** is a widely-known network layer standard. It provides no routing or switching functions, but rather an interface between DTEs and network DCEs. X.25 supports three types of packets: **data** packets, **control** packets, and **interrupt** packets.
- Conventional character-based terminal can be connected to X.25 using a **Packet Assembler/Disassembler** (PAD). **X.3** defines the PAD parameters. **X.28** defines the interface between the PAD and the terminal, and **X.29** defines the interface between the PAD and the remote host.
- **X.75** supports the internetworking of X.25 subnets. The subnets are interconnected by **Signaling Terminals** (STEs) which act as partial IWUs or routers.
- The **Internet Protocol** (IP) is a connectionless datagram protocol developed by the DARPA. **ISO 8473** is a standardization and simplification of IP.

5. The Transport Layer

This chapter describes the transport layer of the OSI model. The transport layer is concerned with the provision of host-to-host user connections for the reliable and cost-effective transfer of user data. Although it may use a variety of networks and network protocols to realize its service, it hides such details from its users by offering a transparent transport service.

The transport layer is similar to the network layer in that both attempt to offer a reliable data transfer service. Their main difference lies in that the transport layer looks at the problem from an end-user's point of view (denoted by a process running on a host), while the network layer's perspective encompasses lower level concerns (such as routing, connection modes, and internetworking). Also, the transport layer operates only on the end hosts, while the network layer also operates on intermediate network nodes.

The transport layer is the lowest user layer in the OSI model. Its scope of responsibility is a function of the quality of service required of it by the user and the quality of service provided by the network layer. The transport layer simply has to bridge this gap, which may be wide in some cases and negligible or nil in other cases.

We will first look at the transport service primitives and see how these define the transport service. Then we will describe the transport protocol and related issues, such as segmentation, multiplexing, addressing, error control, and flow control. Finally, we will discuss TCP as an example of a popular transport layer standard.

5.1. Transport Services

As with the network layer, the transport services are defined in terms of transport primitives. Figure 5.57 summarizes the primitives together with their possible types and parameters.

Figure 5.57 Transport service primitives.

Primitive	Types	Parameters	Purpose
T-CONNECT	request indicate response confirm	(addresses, option, QOS, user data)	Used for initiating a transport connection (connections are always initiated by service users).
T-DISCONNECT	request indicate	(reason, user data)	Used for disconnected a connection (may be initiated by service users or the service provider).
T-DATA	request indicate	(user data)	Used for normal data transfer.
T-EXPEDITED-DATA	request indicate	(user data)	Used for high priority data transfer.

Addresses refer to Transport Service Access Points (TSAP); these denote entities at the transport layer to which connections are made. *Quality Of Service* (QOS) denotes a set of parameters (such as connection/release/transfer delay, connection/release/transfer probability of failure, throughput, error rate) which collectively describe the quality of the transport service requested by the user. *User data* refers to actual user data provided by the service user for transfer by the service provider. *Option* refers to the expedited data function being available in the service.

Figure 5.58 Sample scenario of transport services.

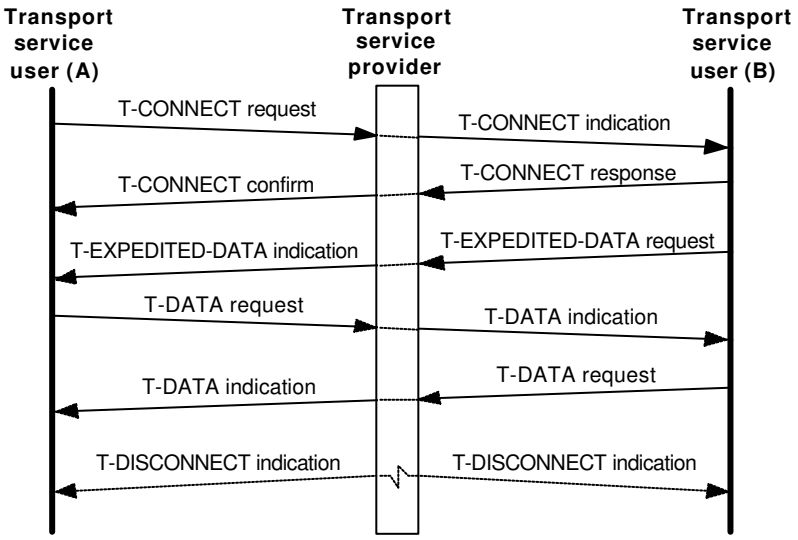


Figure 5.36 illustrates the use of the transport services in a sample scenario. Transport service user A first requests a connection, which is indicated to transport service user B by the service provider. B responds to the request and the service provider confirms with A. Then B expedites some data which is indicated to A by the service provider. Two normal data transfers from A to B, and from B to A then follow. Finally, the service provider simultaneously sends disconnect requests to both A and B and terminates the connection.

5.1.1. Network Types

When a transport service user requests a connection, the request includes a QOS parameter (see T-CONNECT in Figure 5.57). This parameter indicates the user's expected quality of service. The transport layer may have a number of different network services at its disposal, each offering a different quality of service. It needs to match the requested QOS against a network QOS and possibly also perform additional work to fulfill user needs.

For this reason, networks, depending on their QOS, are divided into three broad categories (see Figure 5.59). *Data errors* refer to discrepancies in the bit stream due to transmission errors that have not been detected by the lower-level error checking procedures (e.g., CRC checks). *Signaled errors* refer to failures in the switching mechanisms, resulting, for example, in lost packets.

Figure 5.59 Network types.

Network	Characteristics	Example
Type A	Exhibits acceptable data and signaled error rates. Hence messages arrive correctly and in order. The transport layer need not do much in this case.	Circuit-switched packet network operating over a reliable medium
Type B	Exhibits acceptable data error rates, but unacceptable signaled error rates. The transport layer is responsible for recovery from such errors.	X.25 network
Type C	Exhibits unacceptable data and signaled error rates. The transport layer is responsible for recovery from all such errors.	Datagram network

It should be clear from this table that much of the transport layer's complexity is due to it having to cope with type B and C networks. Compensating for their unacceptable error rates may be a formidable task. However, when the transport layer is unable to bridge the gap between the QOS of these networks and the requested QOS, it may simply refuse to establish the connection.

5.2. Transport Protocol

The transport protocol is essentially connection oriented, regardless of whether the underlying network layer uses virtual circuits or datagrams. After establishing a connection (always full-duplex), the two user processes may exchange data in the normal or expedited fashion. The sequence of exchanged messages is always maintained, except for expedited data, which do not obey the normal flow control. For messages larger than can be handled by the network layer, the transport layer performs the necessary segmentation and re-assembly.

5.2.1. TPDUs

Transport layer messages are exchanged by the network layer using **Transport Protocol Data Units** (TPDUs). In a manner similar to the network sublayers 2 and 3,

a user PDU may be represented by a single TPDU, or segmented into multiple TPDUs when it is too large. The general structure of a TPDU is shown in Figure 5.60.

Figure 5.60 General TPDU structure.

	Field	Description
	Header Length	Length of the TPDU header in octets.
H	TPDU Type	Identifies the type of the TPDU.
E	Credit	Used for flow control.
A	Destination Reference	Reference to the destination user process.
D	Source Reference	Reference to the source user process.
E	Class/Options/Reason	Class of protocol, various options, or disconnect reason.
R	Variable Part	Zero or more less-frequently used parameters, each of the general format <Type,Length, Value>.
	Data	Actual transport service user data.

A TPDU consists of a variable length header and user *Data*. The contents of the header is dependent on the *TPDU Type*.

5.2.2. Classes of Protocol

To facilitate the mapping of the user-requested QOS to an appropriate network QOS, five classes (0-4) of transport protocols are defined. The protocol class is selected during the establishment of a transport connection according to the requested QOS. The choice is made transparently by the transport layer without the user's knowledge or involvement.

Class 0 is the most basic transport service and is suitable for type A networks. Although Class 0 is capable of detecting data errors (e.g., corrupted, lost, duplicated data) and signaled errors (e.g., failure in switching nodes), it cannot recover from them. These errors therefore result in the connection being terminated. This is the simplest form of transport service and must be supported by all transport layer implementations. (Supporting other classes is optional.)

Class 1 protocol supports recovery from signaled errors. It also supports segmentation, expedited data transfer, and acknowledgment using sequence numbers. Class 2 is like class 0, except that it also supports multiplexing and flow control. Class 3 is like class 2, except that it also supports recovery from signaled errors. Finally, Class 4 provides the widest set of features, making its suitable for C type networks. It is the only class that supports the resequencing of TPDUs, and the splitting of the transport connection into multiple network connections to improve throughput.

Figure 5.61 summarizes the five protocol classes and some of their main features, together with the network type suitable for supporting each class.

Figure 5.61 Transport protocol classes.

Class No.	Class Name	Supported Features							Suitable Network
		MUX	DER	SER	FC	ACK	R/S	EXP	

0	Simple class	-	-	-	-	-	-	-	A
1	Basic error recovery class	-	-	✓	-	✓	-	✓	B
2	Multiplexing class	✓	-	-	✓	✓	-	✓	A
3	Error recovery multiplexing class	✓	-	✓	✓	✓	-	✓	B
4	Error detection and recovery class	✓	✓	✓	✓	✓	✓	✓	C

MUX = Multiplexing of multiple transport connections onto a single network connection
DER = Data Error Recovery
SER = Signaled Error Recovery
FC = Flow Control
ACK = Acknowledgments
SPL = Resequencing of TPDUs / Splitting and Recombining connections
EXP = Expedited data transfer

5.2.3. Segmentation

The Transport Service Data Units (TSDUs) which the transport service users work with may be larger than can be handled by the network layer packets. For this reason, all protocol classes support the segmenting of TSDUs into multiple TPDUs and re-assembling them at the receiver end. This facility is very similar to the segmentation facility supported by the internetworking sublayers.

5.2.4. Multiplexing

Given the limited number of ports available on a host and the potentially large number of transport connections usually required by the users, multiple transport connections are often multiplexed onto a single network connection. The price to be paid for this is additional complexity: transport connections need to carry identifiers for distinguishing between connections over the same circuit, and each connection needs to be separately flow controlled. Multiplexing is supported by Class 2, 3, and 4 protocols.

5.2.5. Splitting and Recombining

In situations where a transport connection is needed to have a bandwidth of size that cannot be provided by a single network connection, the connection may be split over multiple network connections. This is the opposite of multiplexing and introduces an additional complexity: with multiple network connections, the TPDUs may arrive out of order and need to be resequenced. Thus splitting can only be supported when the transport protocol can support resequencing of TPDUs. Class 4 is the only protocol that can provide this function.

5.2.6. Addressing

A transport layer source or destination address uniquely identifies a TSAP within a network-wide name space. The TSAP denotes a port on a host to which transport connections are made. Transport addresses are not directly used by TPDUs;

references are used instead. These are much shorter identifiers that are mapped to the transport addresses (in the variable part of the TPDU header) during connection establishment, allowing the two parties to agree on a common mapping. The source and destination references are used for the remainder of the connection.

Also, for user convenience, a higher level mapping is often provided by a name server, which maps meaningful service **names** for frequently-used transport addresses to the addresses themselves. When wishing to connect to one of these services, the user specifies the service name and the network software looks up the name using the name server to come up with the corresponding transport address, which is then used for initiating a transport connection.

5.2.7. Flow Control

As mentioned earlier, supporting multiplexing in Class 2, 3, and 4 protocols requires transport connections to be separately flow controlled. The flow control protocol used for this purpose is called **credit**. It is similar to the sliding window protocol of the data link and network layers in that a window is used. However, unlike the sliding window protocol where the receiver window size remains fixed, here the window size may vary. The sender is informed of the new window size through a data acknowledgment TPDU which includes a new credit value (the *Credit* field in the TPDU) which denotes the number of TPDUs that can be received.

A suitable credit value is negotiated during connection establishment. This value can be dynamically adjusted subsequently by the receiver. When the receiver is unable to receive further TPDUs, it can suspend flow of further TPDUs by adjusting the credit value to 0.

5.2.8. Error Checking

The transport layer may do additional error checking on top of the ones carried out by lower layers. This may serve one of two reasons: the lower layers might not be performing any error checking, or the network service may be very unreliable and deliver packets that have undetected errors.

To address these needs (especially for the Class 4 protocol) a checksum option is available in the TPDUs. This is a 16-bit checksum that covers the entire TPDU and is designed to be much simpler than CRC so that it can be efficiently computed in software.

5.3. Transport Layer Standards

Transport services (discussed in Section 5.1) are defined by the ISO 8072 and CCITT X.214 standards. ISO 8073 and CCITT X.224 define the transport protocol, and were the basis of the descriptions in Section 5.2. Below we will look at TCP— a *de facto* transport layer standard that has gained considerable acceptance in the industry.

5.3.1. TCP

The **Transmission Control Protocol** (TCP) originated in ARPANET, and was designed to cope with Type C networks. Consequently, the CCITT Class 4 protocol has borrowed many of its concepts. TCP sits directly on top of IP (see Chapter 4) and is well suited to internetworking. The combination is commonly referred to as TCP/IP.

TCP accepts arbitrary size messages from a service user and, if necessary, segments them into smaller blocks. Figure 5.62 depicts the general structure of a TCP block. The block is constructed in two stages. First, a TCP header is added to user data; this header is essentially a transport-layer-style header. Next, a datagram header is added to the result to prepare it for a datagram network service.

Given the unreliable network service expectations of TCP (Type C), blocks may be corrupted, lost, duplicated, or out of sequence. TCP uses sequence numbers and resequences blocks when necessary. Unlike other protocols, sequencing is performed on an octet rather than block basis. The sequence numbers for the octets in a block are determined by the *Sequence Number* and the *Total Length* fields. Flow control is also octet-oriented. It is based on a credit-style sliding window, where the window size determines how many octets may be transmitted.

Figure 5.62 General structure of a TCP block.

	Field	Description
Datagram Header	Version	Datagram protocol version.
	Header Length	Datagram header length.
	Service Type	Type of service expected of the network layer.
	Total Length	Total length of the block.
	Segment Owner	The sequence number of the owner datagram.
	Segment Flags	Segmentation-related flags.
	Segment Offset	Offset to where segment begins in owner datagram.
	Lifetime	Block lifetime in seconds.
	Checksum	Simple checksum for the datagram header.
	Addresses	Source and destination network+host addresses.
	Options	Various options for errors, routing, security, etc.
TCP Header	Ports	Source and destination ports.
	Sequence Number	Sequences every octet of data.
	Acknowledgment	Used for piggyback acknowledgment of data.
	Header Length	Length of the TCP header.
	Flags	Flags for SYN, ACK, segmentation, reset, etc.
	Window	Window size in octets.
	Checksum	Simple checksum for the TCP header.
	Expedited	Offset to where expedited data is located.
	Options	Various TCP-related options, e.g., buffer sizes.
	User Data	Actual transport service user data.

Corrupted and lost blocks are retransmitted (the T1 timer is used for detecting lost blocks). Delayed duplicate control blocks for establishing connections are dealt with

using the **three-way handshake** method. In this method, to establish a connection between A and B, when A requests a connection and receives a confirmation from B, it must acknowledge (and therefore finalize) the connection as intended. Should B receive a delayed duplicate connection request and confirm it, A will know that this is due to a duplicate message and reject the connection on that basis.

5.4. Further Reading

Schwartz (1987), Black (1989), and Stallings (1994) provide general descriptions of the transport layer protocols and standards. Black (1992), Feit (1992), Comer (1993), and Piscitello and Chapin (1993) describe TCP/IP in detail.

6. The Session Layer

This chapter describes the session layer of the OSI model. The session layer provides a *structured* means for data exchange between user processes (or applications) on communicating hosts. This layer uses the term **session** instead of connection to signify the point that communication is studied from an application rather than a host point of view. More specifically, a session imposes a set of rules on the way applications should communicate. Also of interest are: how a session is negotiated between two applications, the synchronization and control of message exchanges between applications (e.g., how they should take turns), the context of messages (e.g., whether they relate to records from a database or keystrokes on a terminal), dealing with transport failures, and the bracketing of messages as required by some applications.

We will first look at the session service primitives and their use by higher layers. Then we will describe the session protocol and related issues, such as the use of tokens in activities and dialog units, synchronization, error handling, and the structure of session messages. Finally, we will list a number of session layer standards.

6.1. Session Services

As with earlier layers, the session services are defined in terms of session primitives, of which there are many. Figure 6.63 summarizes the primitives together with their possible types and parameters.

Figure 6.63 Session service primitives.

Primitive	Types	Parameters	Purpose
S-CONNECT	request indicate response confirm	(addresses, QOS, result, requirements, serial no., token, user data)	Used for initiating a session connection (connections are always initiated by service users).
S-RELEASE	request indicate response confirm	(result, user data)	Used for orderly release of a connection.
S-U-ABORT	request indicate	(user data)	Used for service user-initiated abort.
S-P-ABORT	indicate	(reason)	Used for service provider-initiated abort.

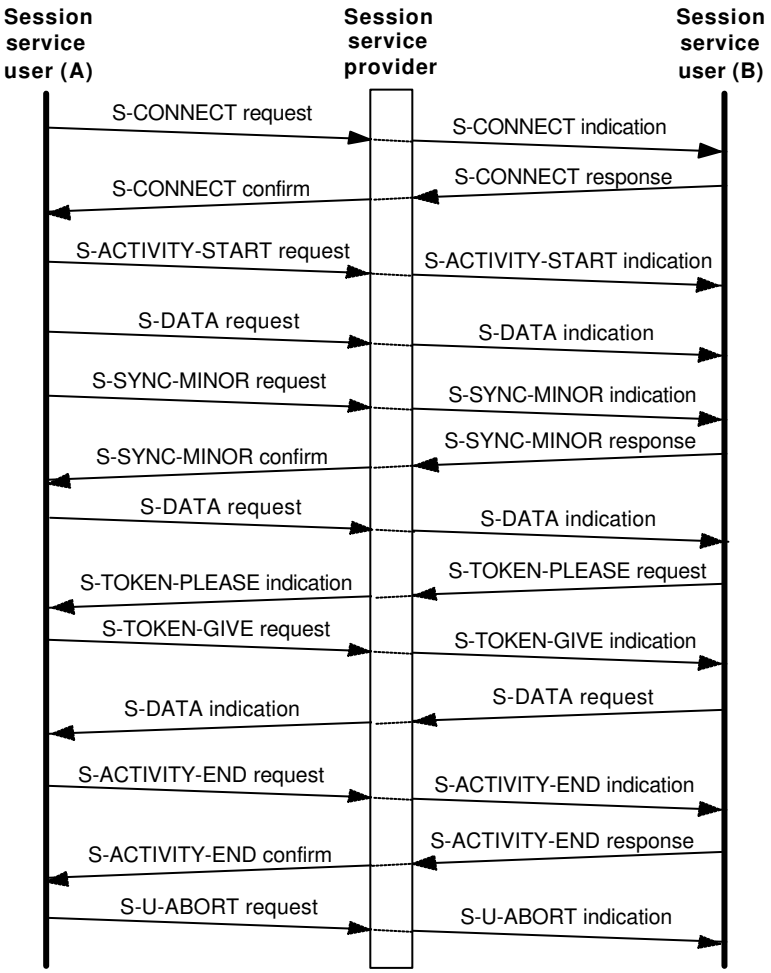
S-U-EXCEPTION-REPORT	request indicate	(reason, user data)	Used for exception reporting by a service user.
S-P-EXCEPTION-REPORT	indicate	(indication)	Used for exception reporting by a service provider.
S-DATA	request indicate	(user data)	Used for normal data transfer.
S-EXPEDITED-DATA	request indicate	(user data)	Used for high priority data transfer.
S-TYPED-DATA	request indicate	(user data)	Used for typed data transfer.
S-CAPABILITY-DATA	request indicate	(user data)	Used for data exchange when no activity is in progress.
S-ACTIVITY-START	request indicate	(activity ID, user data)	Used for starting an activity.
S-ACTIVITY-INTERRUPT	request indicate response confirm	(reason)	Used for temporarily interrupting an activity.
S-ACTIVITY-RESUME	request indicate	(new ID, old ID, serial no., user data)	Used for resuming an interrupted activity.
S-ACTIVITY-DISCARD	request indicate response confirm	(reason)	Used for discarding an activity.
S-ACTIVITY-END	request indicate response confirm	(serial no., user data)	Used for ending an activity.
S-TOKEN-PLEASE	request indicate	(tokens, user data)	Used by a service user for requesting a token.
S-TOKEN-GIVE	request indicate	(tokens)	Used by a service user for forwarding a token to the other user.
S-CONTROL-GIVE	request indicate	()	Used by a service user for forwarding all tokens to the other user.
S-SYNC-MINOR	request indicate response confirm	(type, serial no., user data)	Used for setting a minor synchronization point.
S-SYNC-MAJOR	request indicate response confirm	(serial no., user data)	Used for setting a major synchronization point.
S-RESYNCHRONIZE	request indicate response confirm	(type, serial no., tokens, user data)	Used for resynchronization.

Addresses refer to Session Service Access Points (SSAP); these typically are the same as their corresponding transport addresses. *Quality Of Service* (QOS) denotes a set of parameters which collectively describe the quality of the session service requested by the user. These are essentially the same as those used for the

transport layer, with the addition of a few new ones (e.g., concatenation of service requests). *Result* denotes the acceptance or rejection of the connection. *Requirements* is used for the negotiation and selection of functional units (described below) that are to be effective for the duration of the session. *Serial number* denotes the initial serial number used for synchronization purposes. *Token* refers to the side to which the tokens are initially assigned. *User data* denotes actual user data provided by the service user for transfer by the service provider.

Figure 6.36 illustrates the use of the session services in a sample scenario. Session service user A first requests a (half-duplex) connection, which is indicated to session service user B by the service provider. B responds to the request and the service provider confirms with A. Two normal data transfers from A to B follow, with a minor synchronization cycle in between. B then asks A for the data token, which A hands over to B. B sends some data to A, and then asks for the session to be aborted.

Figure 6.64 Sample scenario of session services.

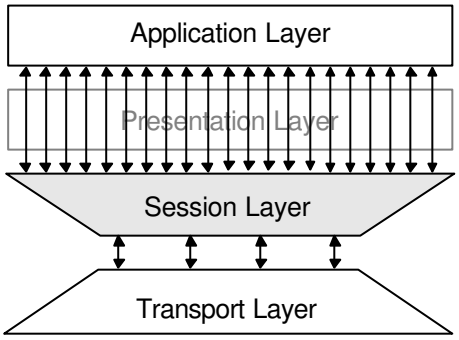


6.1.1. Session Layer Role

The exact role of the session layer in relation to other layers is worth some consideration. Although the session layer is positioned below and receives its requests from the presentation layer, its primary role is to serve the application layer by allowing applications to converse in a structured manner. The presentation layer is completely transparent to this process.

The interface between the session layer and the transport layer is very simple. It provides for basic opening and closing of transport connections (a session connection directly maps to a transport connection) and reliable transfer of data over the connections. This simple interface is enriched by the session layer into a diverse set of services for the ultimate use by the application layer. Figure 6.65 illustrates these points.

Figure 6.65 Role of the session layer.



6.1.2. Functional Units

It is generally rare for an application to require the use of all session services. Often a relatively small subset will suffice. To facilitate this, the session layer services are divided into 13 functional units (see Figure 6.66). Each functional unit is a logical grouping of related session services. Functional units are negotiated and selected during connection establishment using the *requirements* parameter in the connection request. The functional units selected determine what services will be available for the duration of the session.

Kernel represents the most basic set of services that applications could live with. All session layer implementations must provide this minimal subset.

Figure 6.66 Session service functional units.

Functional Unit	Service Primitives	Purpose
Kernel	S-CONNECT S-DATA S-RELEASE S-U-ABORT S-P-ABORT	Provides basic services for establishing a connection, data transfer, and connection termination.
Negotiated Release	S-RELEASE S-TOKEN-PLEASE S-TOKEN-GIVE	Provides the ability to negotiate the orderly release of connections.
Exceptions	S-U-EXCEPTION-REPORT S-P-EXCEPTION-REPORT	Provides error reporting capabilities.
Expedited data	S-EXPEDITED-DATA	Provides expedited data transfer capability.
Typed Data	S-TYPED-DATA	Provides typed data transfer capability independent of the data token.
Capability Data	S-CAPABILITY-DATA	Provides data transfer capability when no activity is in progress.
Activity Management	S-ACTIVITY-START S-ACTIVITY-INTERRUPT S-ACTIVITY-RESUME S-ACTIVITY-DISCARD S-ACTIVITY-END S-TOKEN-PLEASE S-TOKEN-GIVE S-CONTROL-GIVE	Provides activity management capability.
Half Duplex	S-TOKEN-PLEASE S-TOKEN-GIVE	Provides half-duplex data exchange capability using the data token.
Full Duplex	S-TOKEN-PLEASE S-TOKEN-GIVE	Provides half-duplex data exchange capability.
Minor Synchronize	S-SYNC-MINOR S-TOKEN-PLEASE S-TOKEN-GIVE	Provides minor synchronization capability.
Symmetric Synchronize	S-SYNC-MINOR S-TOKEN-PLEASE S-TOKEN-GIVE	Same as Minor Sync, except that it operates in both directions.
Major Synchronize	S-SYNC-MAJOR S-TOKEN-PLEASE S-TOKEN-GIVE	Provides major synchronization capability.
Resynchronize	S-RESYNCHRONIZE	Provides resynchronization capability.

6.2. Session Protocol

As with the transport protocol, the session protocol is essentially connection oriented. It consists of three phases: a connection establishment phase, a data transfer phase, and a connection release phase. The data transfer is by far the most complex of the three, accounting for most of the service primitives. Below, we will

look at various protocol-related concepts which primarily deal with the data transfer phase.

6.2.1. Tokens

Tokens provide a mechanism for limiting the use of certain session services to one of the two session users at a time. Four tokens are provided:

- **Data Token**. This is used for half-duplex connections. The service user possessing the token has the exclusive right to issue S-DATA requests. Data exchanges using S-EXPEDITED-DATA and S-TYPED-DATA requests are not affected by this token. This token is irrelevant to and unavailable in full-duplex connections.
- **Release Token**. This is used for connections which have successfully negotiated the use of the *Negotiated Release* functional unit. The service user possessing the token has the exclusive right to issue an S-RELEASE request. Disconnections using S-U-ABORT requests are not affected by this token.
- **Sync-Minor Token**. This is used for connections which have successfully negotiated the use of the *Minor Synchronize* functional unit. The service user possessing the token has the exclusive right to issue S-SYNC-MINOR requests. This token is irrelevant and unavailable when the *Symmetric Synchronize* functional unit is being used instead.
- **Sync-Major/Activity Token**. This is used for connections which have successfully negotiated the use of the *Major Synchronize* or the *Activity Management* functional unit. The service user possessing the token has the exclusive right to issue S-SYNC-MAJOR and S-ACTIVITY requests.

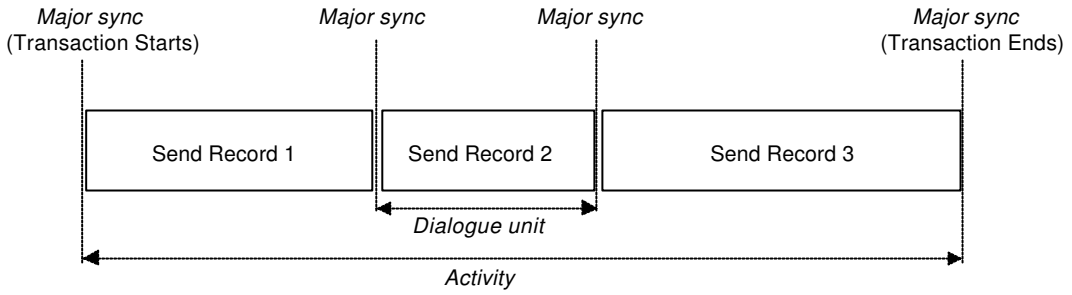
Token distribution is managed by three service primitives. S-TOKEN-PLEASE is used by a service user to request the possession of one or more tokens from the other user. S-TOKEN-GIVE is used by the possessor of the token(s) to forward them to the other user. Finally, S-CONTROL-GIVE enables a service user to forward all its tokens to the other user.

6.2.2. Activities and Dialogue Units

An **activity** represents a convenient way of referring to a set of related tasks as a single entity. It has a clear beginning and a clear end, both of which are marked by **major synchronization** points. An activity consists of one or more atomic tasks, called **dialogue units**. Like an activity, the beginning and end of each dialogue unit is marked by a major synchronization point. Dialogue units (and therefore activities) can be interrupted and resumed later.

A transaction issued by a banking application to an account database is an example of an activity. It may consist of one or more records, each of which represents a dialogue unit (see Figure 6.67).

Figure 6.67 A transaction 'activity'.



The activity service is based on seven service primitives for managing activities. An S-ACTIVITY-START request is issued by one service user to another to indicate the commencement of an activity. It causes the synchronization serial numbers to be set to 1. Each activity is identified by a user-specified identifier. An activity is completed by the service user issuing an S-ACTIVITY-END request, subject to confirmation. Both the activity start and end primitives involve implicit major synchronization points. A service user may interrupt an activity by issuing an S-INTERRUPT-ACTIVITY request, subject to confirmation. Data in transit may be lost, but the activity can be later resumed by issuing an S-ACTIVITY-RESUME request and specifying the identifier of the interrupted activity. A current activity can be discarded by issuing an A-ACTIVITY-DISCARD request, subject to confirmation.

The rest of an activity is occupied by data transfer (using mostly S-DATA) and synchronization, which is discussed next.

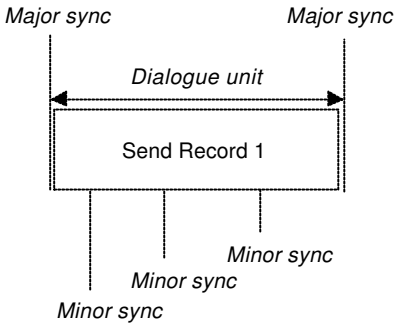
6.2.3. Synchronization

The synchronization service of the session layer is based upon the use of synchronization points. These are markers that are inserted in the data flow to coordinate the exchange of data between applications. There are two types of synchronization points:

- **Major Synchronization Points**. These are used to delimit dialogue units, and hence activities (see Figure 6.5). This service is supported by the S-SYNC-MAJOR primitive. When this request is used by a service user, no further data exchange can take place until it is confirmed by the receiving user. The primitive takes two parameters: a serial number and an arbitrary block of user data.

- **Minor Synchronization Points**. These can be issued at arbitrary points in time, and are used for coordinating data exchange within dialogue units (see Figure 6.68). This service is supported by the S-SYNC-MINOR primitive. The primitive takes one parameter (type) in addition to the two used by S-SYNC-MAJOR; it determines whether the request is to be confirmed.

Figure 6.68 Minor synchronization points.



All synchronization points are serially numbered for ease of reference. The serial numbers are managed by the service provider. For minor synchronization, when the functional unit *Symmetric Synchronize* is selected instead of *Minor Synchronize* at connection time, two serial numbers are employed, one for each data flow direction.

6.2.4. Error Reporting and Resynchronization

The error reporting service is based on two primitives. S-U-EXCEPTION is issued by a service user to report an error condition to the other user. Error conditions detected by the service provider are reported to service users using the S-P-EXCEPTION primitive. All data in transit is lost as a consequence. Also, unless the error is handled by the service users, no further activity is permitted. Depending on the nature of the error, it may be handled in a number of ways: issuing an S-U-ABORT, S-ACTIVITY-INTERRUPT, S-ACTIVITY-DISCARD, or resynchronizing.

Resynchronization is a service used for recovering from errors as well as for resolving disagreements between service users. It allows the service users to agree on a synchronization point, and start from that point. All data in transit is lost as a result. This service is supported by the S-RESYNCHRONIZE primitive, which may be issued by either user and requires confirmation. It takes four parameters: type (explained below), serial number (for a synchronization point), tokens (for distribution of tokens after resynchronization), and user data (an unlimited block). The type parameter may be one of:

- **Abandon**. Aborts the current dialogue by resynchronizing to a new synchronization point with a serial number greater than all previous ones.

- **Restart**. Re-attempts a dialogue by retreating to an earlier synchronization point, provided it is no earlier than the last confirmed major synchronization point.
- **Reset**. Aborts the current dialogue and sets the synchronization serial number to a new value subject to negotiation.

6.2.5. SPDUs

Session layer messages are exchanged by the transport layer using **Session Protocol Data Units** (SPDUs). Most primitives are implemented as one or two SPDUs (the additional SPDU is used where acknowledgment is required, i.e., in confirmed primitives). Some SPDUs are individually mapped onto TSDUs (e.g., Connect and Disconnect SPDUs). Others are always concatenated with Please-Token and Give-Token SPDUs and then mapped onto TSDUs (e.g., all Activity SPDUs).

The general structure of an SPDU is shown in Figure 6.69. The exact parameters depend on the type of SPDU. As indicated in the figure, parameters may be grouped together. Furthermore, parameter groups may contain subgroups.

Figure 6.69 General SPDU structure.

		Field	Description
		Service ID	Specifies the type of this SPDU.
		Length Indicator	Total length of the following parameters.
PARAMETERS	sample single par.	Parameter ID	Specifies the type of this parameter.
		Length Indicator	Length of the parameter.
		Parameter Value	Specifies the value of the parameter.
		...more parameters or parameter groups...	These can be in any order.
	sample par. group	Parameter Group ID	Specifies a group of parameters.
		Length Indicator	Total length of parameters in this group.
		Group	The group parameters appear one by one here
		Parameters	and may contain other groups as well.
		User Data	Actual session service user data.

6.3. Session Layer Standards

Session services (discussed in Section 6.1) are defined by the ISO 8326 and CCITT X.215 standards. ISO 8327 and CCITT X.225 define the session protocol, and were the basis of the discussions in Section 6.2.

6.4. Further Reading

General descriptions of the session layer and related protocols can be found in Tanenbaum (1989), Black (1989), Martin (1990), Stamper (1991), and Stallings (1994).

7. The Presentation Layer

Applications use a variety of forms of data, ranging in complexity from very simple (e.g., textual) to elaborate and complex (e.g., nested data structures). Communication between applications involves the exchange of such data. However, all forms of data have inherent programming language and machine dependencies, which means that unless application data is accurately converted to a format acceptable by the peer application, the meaning of the data will be lost during transmission.

The role of the presentation layer is to facilitate semantic-preserving data exchange between two peer applications. The presentation layer achieves this in two stages: (i) by having the two peer applications adopt a common high-level syntax for data definition, and (ii) by negotiating a transfer syntax for the sole purpose of transmission, which both applications can convert to and from.

We will first look at the notion of abstract data syntax, and then describe presentation service primitives and functional units. ASN.1 will be presented as a standard abstract syntax definition notation for the use of applications, together with BER for defining data at the binary level. Use of ASN.1 and BER will be illustrated using some simple examples. Next, we will discuss the presentation protocol, and conclude by listing a number of presentation standards.

7.1. Presentation Services

The notion of syntax is central to understanding the presentation services. This is discussed below first, followed by a description of presentation service primitives and service functional units.

7.1.1. Syntax

Data is structured according to a set of rules, called **syntax**. Depending on their level of abstraction, syntax rules may be classified into two categories: abstract and concrete. **Abstract syntax** is a high-level specification of data which makes no assumptions about its machine representation. Abstract syntax describes the essential characteristics of data in general terms. **Concrete syntax**, on the other hand, is a low-level (bit-level) specification of data according to some specific machine representation. In general, a concrete syntax is derived from an abstract syntax by applying a set of encoding rules to the latter. It follows that there is a one-to-many

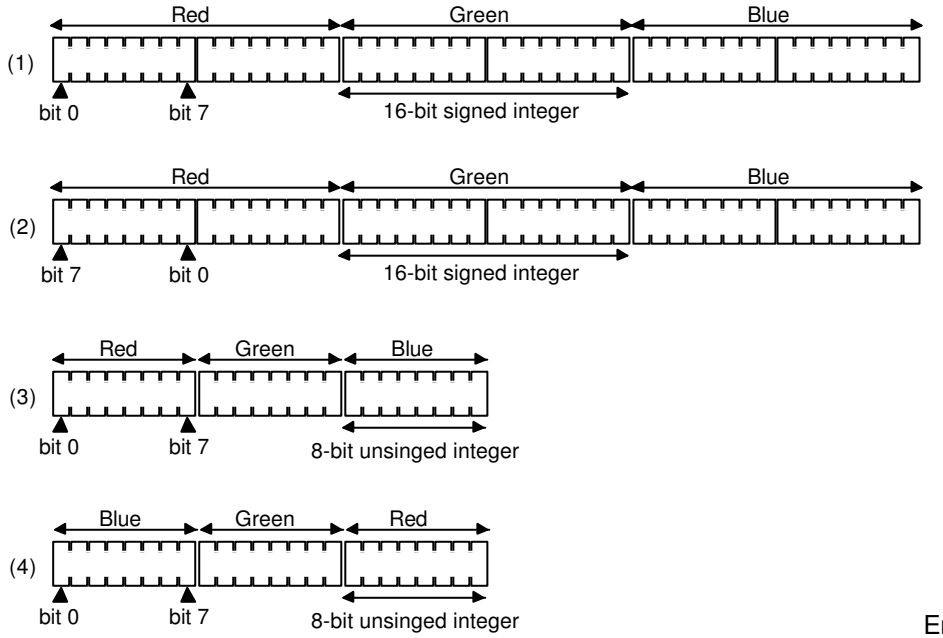
mapping between the abstract syntax of some data and its concrete syntaxes, that is, the same data can be represented in many different formats.

For example, consider the following statement:

> *An RGB color is defined as an object of three components (red, green, and blue), each of which is an integer quantity.*

This is an example of an abstract syntax specification. It specifies the essential characteristics of an RGB color without saying anything about its representation in a computer. Some of the ways in which this data can be represented at a bit-level are shown in Figure 7.70. Format 1 uses a 16-bit signed integer quantity for representing each primary color, in a machine where bits are ordered left to right. Format 2 is identical to format 1, except that bits are ordered right to left. Formats 3 and 4 both use 8-bit unsigned integers with bits ordered left to right. However, in format 4, the primary colors appear in reverse order.

Figure 7.70 Four alternate representation formats for an RGB color.



Error! Objects cannot be created from editing field codes.

Concrete syntax is essential wherever data is to be digitally stored or communicated. In general, each system has its own concrete syntax which may be different to the concrete syntax of other systems. Two communicating applications running on two such systems would have to convert their data into a common concrete syntax to facilitate transmission. We will use the terms **application concrete syntax** and **transfer concrete syntax** to distinguish between these two

syntaxes. To preserve the characteristics of the data which is subject to transformation, the two applications should standardize on the same abstract syntax notation. Figure 7.71 illustrates the role of various syntaxes.

Figure 7.71 The role of various syntaxes.

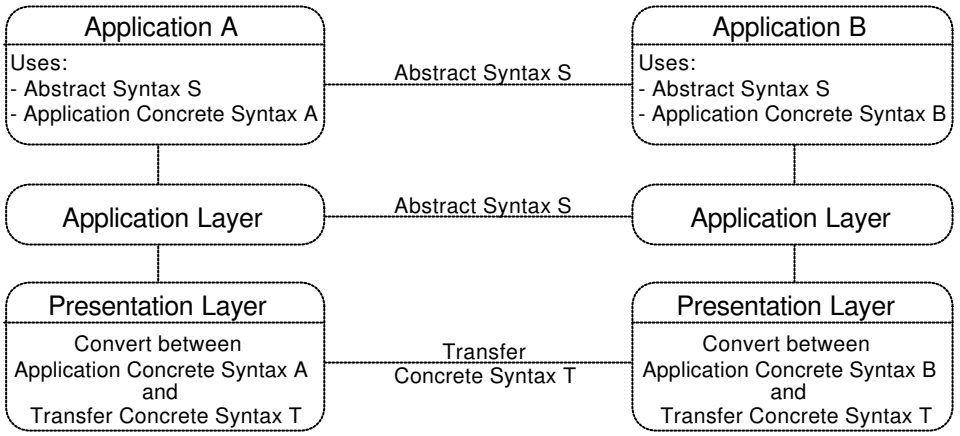


Figure 7.72 Presentation service primitives.

Presentation Primitive	**Types**	**Related Session Primitive**	**Purpose**
P-CONNECT	request indicate response confirm	S-CONNECT	Same as S-CONNECT except that it also negotiates the presentation functional units, a defined context set, and a default context.
P-RELEASE	request indicate response confirm	S-RELEASE	Same as S-RELEASE, but also deletes the defined context set.
P-U-ABORT	request indicate	S-U-ABORT	Same as S-U-ABORT, but also deletes the defined context set.
P-P-ABORT	indicate	S-P-ABORT	Same as S-P-ABORT, but also deletes the defined context set.
P-*other*	...	S-*other*	*Identical to its session counterpart.*
P-ALTER-CONTEXT	request indicate response confirm	none	Used for making changes to the defined context set after a connection has been established.

7.1.2. Service Primitives

As mentioned in the previous chapter, the presentation layer is transparent to the exchanges taking place between the application and session layers for structuring application communication. Consequently, each session layer service primitive is represented by a corresponding presentation layer primitive. The presentation layer adds functionality to four of the session service primitives and provides a new primitive (see Figure 7.72). The remaining service primitives are identical in functionality to their session layer counterparts.

A choice of syntaxes is made through negotiation between peer applications at connection time. Applications negotiate and agree on the abstract syntaxes for all data which is subject to transfer. For each abstract syntax *S*, they should also agree on a corresponding transfer syntax *T*. The combination of *S* and *T* is called a **presentation context**. In most cases, more than one presentation context is used for the same presentation connection. The set of all such contexts is called the **defined context set**. Applications also negotiate a **default context**, which is used when the defined context set is empty.

A presentation connection is directly mapped onto a session connection. Most of the presentation service primitives are mapped directly and unchanged onto corresponding session primitives. Figure 7.36 illustrates the use of the presentation services in a sample scenario which maps directly to Figure 6.2 at the session layer.

Conversion between application concrete syntaxes and the transfer concrete syntax is managed by the presentation layer (see Figure 7.71). All transferred data is *tagged* by the transmitting end to enable the receiving end to determine its presentation context. This enables the latter to correctly map the bit strings into meaningful data for use by the receiving application.

Figure 7.73 Sample scenario of presentation services.

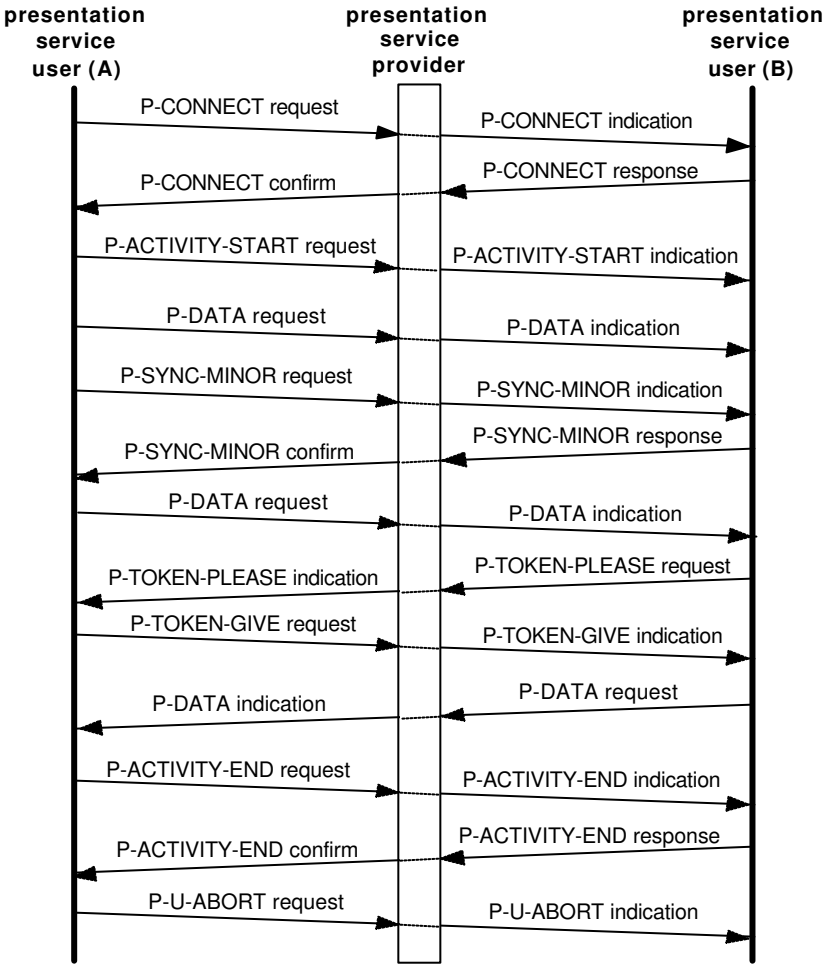


Figure 7.74 Presentation service functional units.

Presentation Functional Unit	Related Session Functional Unit	Purpose
Kernel	Kernel	Same as the session kernel functional unit with the addition of defined context set negotiation during presentation connection.
Context Management	none	Provides the ability to negotiate context changes using P-ALTER-CONTEXT after successful presentation connection.
Context Restoration	none	Provides the ability to restore context changes by remembering the defined context set for each synchronization point.

Other	*Other*	All remaining session functional units are also available as presentation functional units.

7.1.3. Functional Units

Figure 7.66 summarizes the available presentation functional units, which includes all of the session functional units. *Kernel* represents a mandatory subset of services which all presentation layer implementations must provide.

7.2. Abstract Syntax Notation One

Abstract Syntax Notation One (ASN.1) is a formal language which has been designed to enable applications share a common data definition. It supports the programming language-independent and machine-independent definition of data. ASN.1 is used by the application layer for defining application data, and by the presentation layer for defining the protocol data units used for exchanging the data. A separate notation, BER, is used for specifying concrete syntaxes. These two are discussed in the next two sections.

7.2.1. Definitions in ASN.1

A set of ASN.1 data type definitions is embodied by a **module**. The general (but simplified) structure of a module definition is as follows:

Module-name DEFINITIONS ::= BEGIN
    $Type\text{-}assignment_1$
    ...
    $Type\text{-}assignment_n$
END

where items appearing in *italics* are place holders, and items appearing in upper case throughout are literals. This definition simply states that a module definition consists of one or more type assignments. Each type assignment is of the form:

Type-name ::= *Type-definition*

A type definition may be just a reference to a simple type or consist of a structured type. An example of a **simple type** would be:

PacketLifetime ::= INTEGER

Figure 7.75 summarizes the built-in simple types of ASN.1.

Figure 7.75 Built-in simple types of ASN.1.

Type	Description
BOOLEAN	Represents logical data (taking true or false values).
INTEGER	Represents all whole numbers.
REAL	Represents scientific notation real numbers.
BIT STRING	Represents arbitrary sequences of bits.
OCTET STRING	Represents arbitrary sequences of octets.
ENUMERATED	Represents a set of uniquely-named values.
NULL	Represents the empty set (no information).

A **structured type** is always defined in terms of other (predefined or user-defined) types. An example of a structured type would be:

```
Message ::= SEQUENCE {
        protocol        INTEGER,
        ack-needed      BOOLEAN,
        source          Address,
        destination     Address,
        data            BIT STRING
}
```

which defines a *Message* as an ordered sequence of five other components. *Address* denotes a user-defined type which we assume has already been defined. Figure 7.76 summarizes the structured types of ASN.1.

Figure 7.76 Built-in structured types of ASN.1.

Type	Description
SET	A collection of items of possibly different types and in no particular order.
SET OF	Same as SET except that all the items are of the same type.
SEQUENCE	A collection of items of possibly different types but in a specific order.
SEQUENCE OF	Same as SEQUENCE except that all the items are of the same type.
CHOICE	A data type which may assume one of several types.
SELECTION	An item whose type is previously defined as a CHOICE type.
ANY	Represents any valid ANS.1 type.
TAGGED	Represents a way of identifying multiple occurrences of the same type.

Tagged types are used to resolve potential ambiguity in type definitions. For example, had *Message* been defined as a SET instead of a SEQUENCE, an ambiguity would arise between *source* and *destination* components, because both these are of the same type. This presents no problems at the abstract syntax level,

but as far as the transfer syntax is concerned, *source* and *destination* can occur in any order and are therefore indistinguishable. The ambiguity is resolved by simply tagging the clashing components:

```
Message ::= SET {
        protocol        INTEGER,
        ack-needed      BOOLEAN,
        source          [0] Address,
        destination     [1] Address,
        data            BIT STRING
}
```

Tags represent the general mechanism for encoding type information for use in the transfer syntax. Overall, there are four classes of tags. These are summarized in Figure 7.77. Explicit use of universal tags is redundant because each of these has a corresponding built-in type. So, for example, instead of [UNIVERSAL 1] which represents the built-in boolean type, the keyword BOOLEAN is used.

Figure 7.77 The four tag classes of ASN.1.

Tag Class	Usage	Description
Universal	*Type-name* ::= [UNIVERSAL *n*] *Type-name*	Application-independent types of general use. It represents the simple and structured built-in types described earlier.
Application-wide	*Type-name* ::= [APPLICATION *n*] *Type-name*	Application-specific types which have been devised to serve the needs of application layer standards.
Context-specific	[*k*] *Type-name* ... [*n*] *Type-name*	Used for distinguishing between the clashing types of the components of a group.
Private-use	*Type-name* ::= [PRIVATE *n*] *Type-name*	Reserved for the private use of applications, outside the scope of the standards.

7.2.2. Basic Encoding Rules

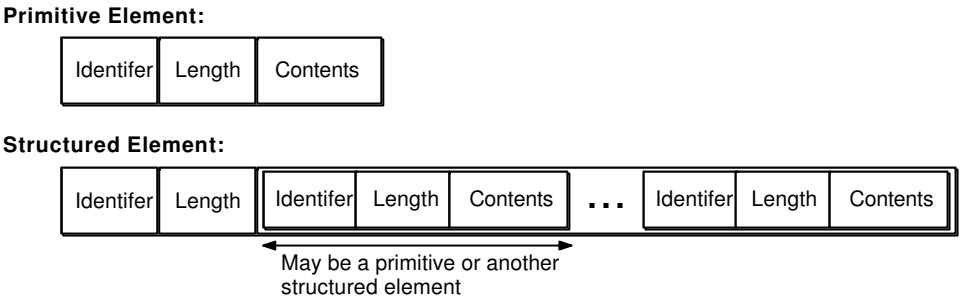
The Basic Encoding Rules (BER) of ASN.1 provide a mechanism for mapping an abstract syntax definition of data in ASN.1 to a representative transfer syntax.

The transfer syntax is defined as a set of transfer types, which are specified as data elements. A **data element** consists of three fields:

- **Identifier**. This uniquely identifies the data type. It is an encoding of a tag class, a tag number, and a flag which indicates the form of the element as being primitive or structured (see Figure 7.78).

- **Length**. This is an encoding of the length of the contents field in octets.
- **Contents**. This is an encoding of the actual value of the data based on the type information provided by the identifier field.

Figure 7.78 Data element forms.



The data element always consists of an integral number of octets, where the bits in each octet are ordered left to right (i.e., the most significant bit appearing on the left). The *identifier* field of a data element is encoded in one or more octets as follows. Bits 7 and 8 encode the tag class. Bit 6 encodes the data element form. The remaining five bits encode the different types within the same tag class; additional octets are used if the five bits are insufficient (see Figure 7.79).

Figure 7.79 Encoding of the identifier field of a data element.

Bit	Identifier Encoding		
8	*Tag Class*:	00 = Universal	01 = Application-wide
7		10 = Context-specific	11 = Private-use
6	*Form*:	0 = Primitive	1 = Structured
5	*Tag Type*:		
4		• If no more than 32 types, use 00000-11110	
3		• If more than 32 types, use 11111, followed by an arbitrary number of	
2		intermediate octets of the form:	1bbbbbbb
1		and terminated by an octet of the form:	0bbbbbbb

The *length* field of a data element is encoded as one or more octets in one of three forms:

- **Short Form**. This is one octet long, with bit 8 of the octet permanently set to 0 (i.e., 0bbbbbbb) and therefore can represent content fields of up to 127 octets long.
- **Long Form**. This can be up to 128 octets long, where the first octet is of the form 1bbbbbbb, whose least significant 7 bits denote the number of octets to follow.

- **Indefinite Form**. This is one octet long and the octet has the fixed value 10000000. End of the contents field (which may have an arbitrary length) is denoted by two consecutive zero octets (i.e., 00000000 00000000). This length form is only available to structured elements.

The *contents* field of a data element is encoded according to the data type. Figure 7.80 summarizes the encoding used for the built-in simple types. Contents encoding for values of structured types uses structured elements. For a sequence, for example, each sequence element is encoded as a data element within a structured element.

Figure 7.80 Encoding of the contents field of a data element.

Built-in Type	Contents Encoding
BOOLEAN	FALSE = all bit zero, TRUE = at least one bit non-zero.
INTEGER	2's complement binary number.
REAL	Encoded as a sign bit and three separate integer quantities: base, mantissa, and exponent (i.e., scientific notation).
BIT STRING	Zero or more bits. The left-most octet denotes the number of unused bits in the last octet.
OCTET STRING	zero or more octets.
ENUMERATED	Individual items encoded as sequenced INTEGER values.
NULL	Has no contents.

7.3. Presentation Protocol

The presentation protocol is very similar to the session protocol and the **Presentation Protocol Data Units** (PPDUs) differ from their corresponding SPDUs only in a few minor details. The only real major protocol addition at the presentation layer is the ability to negotiate the defined context set, both at connection time and during data transfer.

Defined context set negotiation involves a presentation service user proposing a set of abstract syntaxes, for each of which one or more transfer syntaxes is also proposed. Each 'abstract syntax + transfer syntaxes' pair is identified by a **context identifier**. The peer service user individually accepts or rejects each of the pairs. For each accepted pair, it also selects one of the proposed transfer syntaxes. This identifies a set of presentation contexts and establishes the defined context set.

Because there may be more than one presentation context in the defined context set, each data transfer is tagged with the context identifier of the presentation context to which it conforms. The receiving end uses this tag to determine how to decode the data.

7.4. Presentation Standards

Presentation services (discussed in Section 7.1) are defined by the ISO 8822 and CCITT X.216 standards. ISO 8823 and CCITT X.226 define the presentation protocol, and were the basis of the discussions in Section 7.3. ASN.1 is defined by the ISO 8824 and CCITT X.208 standards. ISO 8825 and CCITT X.209 describe the BER component of ASN.1.

7.5. Further Reading

General descriptions of the presentation layer and related protocols can be found in Tanenbaum (1989), Martin (1990), Stamper (1991), and Stallings (1994). Black (1989) and Dickson and LLoyd (1992) contain useful introductions to ASN.1.

8. The Application Layer

The application layer is comprised of a variety of standards, each of which provides a set of useful services for the benefit of end-user applications. Only those services which can be provided in a system-independent fashion are subject to standardization. These include: virtual terminal handling, message handling, file transfer, job transfer, and many others.

Describing all existing application layer standards is beyond the scope of this book, as they are numerous and detailed. This chapter will look at a few of such standards. The aim is to illustrate the flavor of the services provided rather than provide an exhaustive coverage of the standards.

We will first discuss application services, and explain some of the relevant terminology. Then we will describe two groups of application service elements. The first group is commonly used by other application service elements. The second group addresses very specific requirements. Finally, we will list a number of application layer standards.

8.1. Application Services

Depending on their scope of use, application services may be divided into two broad categories:

- General, low-level services which are used by most applications. This group includes three sets of services: association control, reliable transfer, and remote operations. They are further described in section 8.2.
- Specific, high-level services which are designed to support task-oriented application requirements. Examples include: virtual terminal handling, message handling, and file transfer. These directly utilize the general application services, and are further described in Sections 8.3-5.

User applications should make as much use of the above services as possible so that they can benefit from the standardizations. A user application can be imagined as consisting of two parts: (i) a standardized part, called **Application Entity** (AE), which is immersed in the application layer and is implemented in terms of the application layer services, and (ii) a non-standard part, called **Application Process** (AP), which remains outside the scope of the application layer. Typically, the application entity deals with the inter-application communication, and the application

process provides the user front-end and interfaces to the local environment (see Figure 8.85).

Figure 8.81 User applications in relation to the application layer.

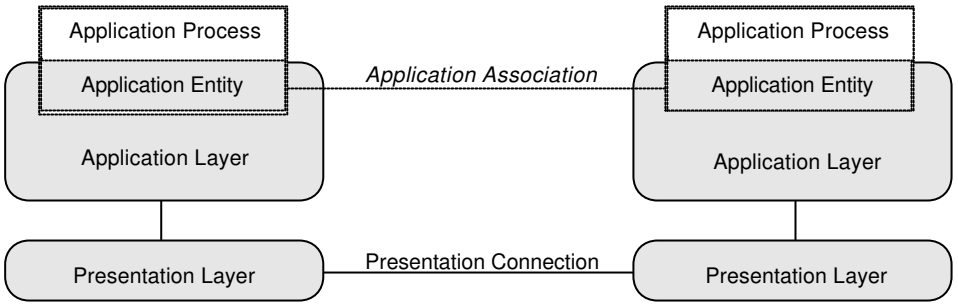


Figure 8.85 illustrates a further point. Unlike earlier layers, the application layer does not provide connections. This should not come as a surprise, because there are no further layers above the application layer that would otherwise use such a connection. Instead, two applications use an **association** between their application entities for exchange of information over the presentation connection. An association serves as an agreement so that the two parties are aware of each other's expectations.

Application processes will be no further discussed, as they are outside the scope of the OSI model and standards. Application entities are the focus of what the application layer standards are about, and are discussed further below.

8.1.1. Application Entity

As mentioned earlier, an application entity accounts for that part of a user application which falls within the scope of the application layer. An application entity consists of one or more application service elements and a control function (see Figure 8.82). An **Application Service Element** (ASE) represents an application service and its associated standard protocol. A **Control Function** (CF) regulates the activities of an application entity. This includes managing the ASEs and the association with the peer application entity.

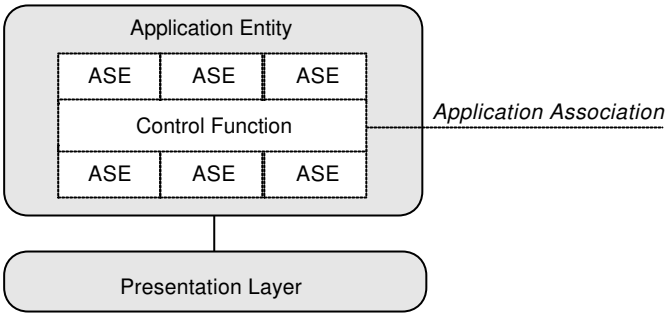
Peer application entities must share a common **application context** by ensuring that either entity uses exactly the same set of ASEs and uses them in the same fashion.

There are two classes of ASEs:

- A Common Application Service Element (CASE) denotes a general application service.
- A Specific Application Service Element (SASE) denotes a specific application service.

We will look at these in turn.

Figure 8.82 Makeup of an application entity.



8.2. Common Application Service Elements

This section describes three CASEs used by many other ASEs.

8.2.1. Association Control

The Association Control Service Element (ACSE) is used by every application entity. It provides a set of functions for managing the association between peer application entities. Since managing an association is independent of the activities the association is used for, ACSE serves as an appropriate and highly useful standard.

An association is established using the A-ASSOCIATE service primitive. This maps to the P-CONNECT primitive and hence also establishes a corresponding presentation connection. The application context and the abstract syntaxes are also conveyed during this process. The A-RELEASE primitive maps to P-RELEASE and handles the orderly release of an association as well as the release of the corresponding presentation connection after ensuring that all data in transit has been delivered. Error-initiated releases are managed by A-ABORT and A-P-ABORT which, respectively, map to and are similar to P-U-ABORT and P-P-ABORT .

The ACSE service primitives are implemented as **Application Protocol Data Units** (APDUs). Each of these represents a unique application-wide type and is tagged accordingly (see Figure 7.8).

ISO 8649 and CCITT X.217 standards describe the ACSE service. The ACSE protocol is described in ISO 8650 and CCITT X.227 standards.

8.2.2. Reliable Transfer

The Reliable Transfer Service Element (RTSE) hides much of the underlying complexity of the session dialogue management services by providing high-level error handling capabilities to other ASEs for data transfer. RTSE segments a data transfer

APDU into smaller PDUs, each of which is marked by a confirmed minor synchronization point. The transfer of an APDU is managed as one session activity.

RTSE uses the ACSE to establish an association: its RT-OPEN service primitive maps to the A-ASSOCIATE primitive. RT-TRANSFER is a confirmed primitive and manages the transfer of an APDU as segments. It maps to a sequence of P-ACTIVITY, P-DATA and P-SYNC-MINOR primitives. RT-CLOSE maps to A-RELEASE and gracefully terminates the transfer. In case of errors, RTSE provides appropriate recovery procedures. When possible, a transfer activity which has been disrupted by an error is restarted; otherwise, it is entirely repeated.

ISO 9066.1 and CCITT X.218 standards describe the RTSE service. The RTSE protocol is described in ISO 9066.2 and CCITT X.228 standards.

8.2.3. Remote Operations

The Remote Operations Service Element (ROSE) serves the needs of distributed applications in invoking remote operations. An example is an application which requires access to a remote database. ROSE enables a requester AE to submit an operation to a replier AE, then waits for a response, and finally delivers the response to the application. The response may indicate the result of successful completion of the operation, an error condition, or complete rejection of the operation.

ROSE supports two modes of operation: (i) synchronous, whereby the requester AE waits for a result before submitting the next operation, and (ii) asynchronous, whereby the requester may submit further operations before waiting for earlier ones to be completed.

ROSE provides a remote operation notation which other ASEs may use to define their operation interface. The ROSE protocol specifies how its four APDUs are transferred using the presentation service (which maps them to P-DATA primitives) or the RTSE service (which maps them to RT-TRANSFER primitives).

ISO 9072.1 and CCITT X.219 standards describe the ROSE service. The ROSE protocol is described in ISO 9072.2 and CCITT X.229 standards.

8.3. Specific Application Service Elements

This sections describes three widely-used SASEs.

8.3.1. Virtual Terminal

There are many different makes and types of character-based terminals in use throughout the world. Very few of them use the same set of commands to control the display or obtain input from keyboards or other input devices. Because of these incompatibilities, terminal dependency has been a common problem in applications.

The aim of the Virtual Terminal (VT) standards is to facilitate terminal independency by providing a model for connecting applications and terminals which hides the device specific information from applications.

VT consist of two standards: ISO 9040 describes the VT services, and ISO 9041 describes the VT protocol. VT employs a model in which terminal access is provided through a Conceptual Communication Area (CCA). CCA provides data abstractions for the terminal screen, keyboard, etc., in form of objects, of which there are three types:

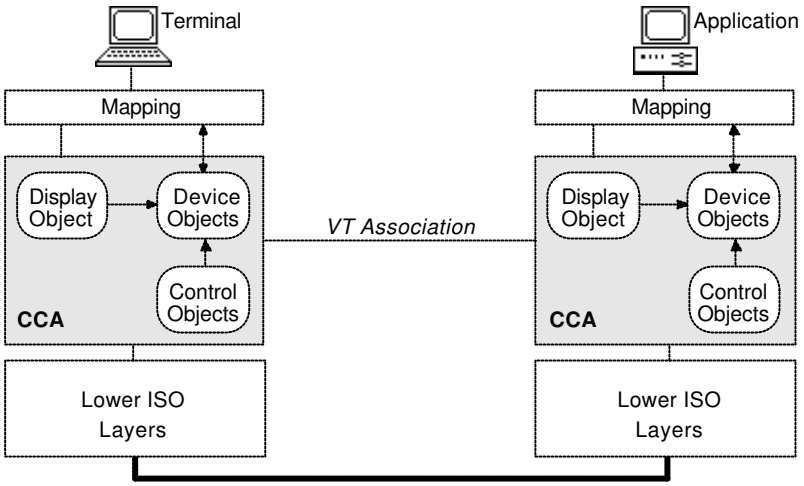
- **Display Object**. All terminal data is routed through a display object. The display object reflects the state of the terminal display and/or its related input devices.
- **Device Object**. The device object specifies the physical characteristics of a device. Naturally, the information provided by a device object is device dependent and outside the scope of the standard.
- **Control Object**. A control object manages a specific VT function. There are generally many control objects responsible for functions such as interrupts, character echoing, and field definition.

Figure 8.85 illustrates the role of the CCA and its objects. Using the device objects, the display and control objects are mapped to the actual terminal device. VT maintains a copy of CCA at both the terminal and the application end, and ensures that these two copies reflect the same picture by exchanging updates between them as they take place.

VT supports synchronous (called S-mode) and asynchronous (called A-mode) communication between terminals and applications. In the S-mode the same display object is used for input and output paths. In the A-mode two display objects are employed, one for the input device and one for the output device.

The VT service contains a set of facilities for managing the communication process. These are used to establish and terminate VT associations, negotiate VT functional units, transfer data, exchange synchronization and acknowledgment information, and manage access rights.

Figure 8.83 CCA and its objects.



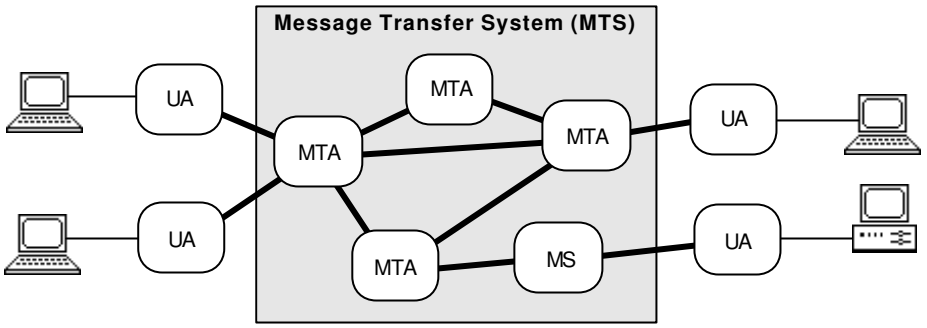
8.3.2. Message Handling Systems

Electronic mail (e-mail) represents one of the most successful classes of network applications currently enjoyed by many users. Early e-mail systems were network dependent, and their use was limited to the private networks of individual organizations. The CCITT X.400 and the ISO 10021 series of standards for Message Handling Systems (MHS) have paved the way for standardized and widely-available e-mail services.

Figure 8.85 illustrates the X.400 view of MHS architecture. At the center of MHS is a **Message Transfer System** (MTS) which handles the delivery of messages. The system consists of the following components:

- A **Message Transfer Agent** (MTA) is responsible for the routing of complete e-mail messages (called **envelopes**) through the MTS. MTAs handle envelopes in a store-and-forward fashion.
- A **User Agent** (UA) manages a user's mailbox. It enables the user to create, submit, and receive messages. The UA may serve an application or provide a user interface for direct interaction. UAs typically run on multi-user systems (e.g., mainframes).
- A **Message Store** (MS) acts on behalf of a UA running on a system which may not be available on a continuous basis (e.g., personal computers). MSs are typically used within a LAN environment, serving a collection of personal computers.

Figure 8.84 MHS architecture.



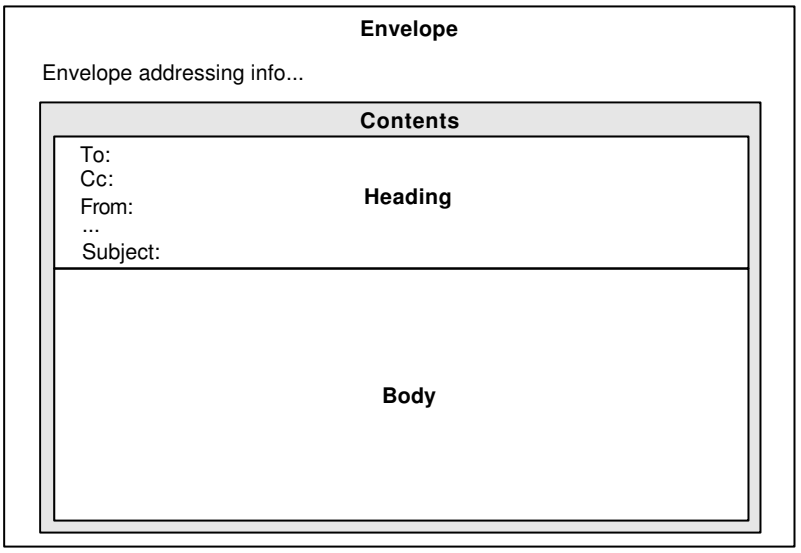
UAs and MSs make it possible for users to receive messages when they are not personally present, and while even their terminal or personal computer is not switched on. They simply store the messages and notify the user at the earliest opportunity.

Each user has its own UA. Furthermore, each user is identified by a unique address which is structured in a hierarchical fashion (similar to a postal address). The address structure reflects the division of MTAs into **domains**. Domains exist at various level of abstraction: country, organization, unit, etc.

Figure 8.85 illustrates the general structure of an envelope. It consists of contents and addressing information. The **contents** consists of two parts: heading and body. A **heading** is comprised of fields such as recipients' addresses, addresses to which the message should be copied, originator's address, subject, etc. Some of the heading information is provided by the user (e.g., recipients' addresses and subject), others are automatically inserted by the UA (e.g., date and originator's address). The **body** contains the message information itself. It may consist of more than one part, each of which may be of a different type (e.g., text, digitized voice, digitized image).

An envelope is constructed by a UA by deriving envelope **addressing** information from the heading and adding it to the contents. MTAs only deal with envelopes and their addressing. They have no interest in the contents of an envelope. Each receiving MTA looks at the addressing of an envelope, records on it the last leg of the route so far, time stamps it, and hands it over to the next MTA. The envelope therefore bears a complete trace of its route through the network of MTAs.

Figure 8.85 Envelope structure.



As would be expected, the MHS service is defined by a set of service primitives. Figure 8.35 summarizes the primitives and their purpose.

Figure 8.86 MHS service primitives.

Primitive	Issued By	Types	Purpose
LOGON	UA	request response	To initiate a session. If successful, the UA is informed of any messages waiting.
	MTA	indicate confirm	
LOGOFF	UA	request	To terminate a session.
	MTA	confirm	
CHANGE PASSWORD	UA	request	To change the UA's logon password.
	MTA	indicate confirm	
REGISTER	UA	request	To change the UA's profile maintained by the MTA.
	MTA	confirm	
SUBMIT	UA	request	To submit a message (an envelope) to the MTA.
	MTA	confirm	
DELIVER	MTA	indicate	To deliver a submitted message to the UA.
NOTIFY	MTA	indicate	To notify the UA of message delivery.
CANCEL	UA	request	To cancel a message delivery to a UA.
	MTA	confirm	
PROBE	UA	request	To check if a SUBMIT would be successful.
	MTA	confirm	
CONTROL	UA	request response	To alter the ways in which message storage and delivery is controlled by the MTA.
	MTA	indicate confirm	

MHS uses four protocols (P1, P2, P3, and P7) to provides two types of service:

- The **Message Transfer** (MT) service supports the handling of envelopes. This service operates between MTAs (P1 protocol), between UAs and MTA/MSs (P3 protocol), and between UAs and MSs (P7 protocol).
- The **Inter-Personal Messaging** (IPM) service supports the handling of the contents of envelopes. This service operates between UAs (P2 protocol). IPM depends on the MT service for its operation.

These services are organized as service groups, each of which contains related **service elements**. The service elements are supported by the service primitives depicted in Figure 8.35. Figure 8.87 summarizes sample MT and IPM service elements.

MHS protocols use the three common service elements described in Section 8.2 (i.e., ACSE, RTSE, and ROSE). All envelopes are transferred as APDUs using the RTSE. All MHS protocols and messages are defined in ASN.1 and have BER codings. This includes a set of application-wide tags defined for identifying the various data types defined for MHS.

Figure 8.87 Service groups and sample service elements.

	Group	Service Element	Purpose
MT	Basic	Content Type Indication	Allows the originating UA to indicate the message content types (e.g. text, binary) to the recipient.
		Message Identification	Used by the MTS to uniquely identify each message passing through it.
	Submission & Delivery	Delivery Notification	Allows the originating UA to request notification of receipt from the receiving UA.
		Multidestination Delivery	Allows the originating UA to specify multiple destination addresses.
	Conversion	Prohibit Conversion	Allows the originating UA to ask the MTS not to perform any conversion on the message.
		Explicit Conversion	Used by the UA to ask the MTS to perform code conversion on the message.
IPM	Basic	Same as MT	Same as MT
	Action	Receipt Notification	Used by the originating UA to ask the receiving UA to notify the originator of the receipt.
		Auto-forward Indication	Used by the receiving UA to determine if the message has been auto-forwarded.
	Information Conveying	Subject Indication	Used by the originating UA to convey the message subject to the receiving UA.
		Expiry Indication	Used by the originating UA to specify when the message will become invalid.

The X.400 series of standards consist of a number of recommendations, of which the following were touched upon in this section:

CCITT X.400 / ISO 10021.1	MHS Service Overview
CCITT X.402 / ISO 10021.2	MHS Architecture
CCITT X.411 / ISO 10021.4	MHS Message Transfer Service Definition
CCITT X.419 / ISO 10021.6	MHS Protocol Specification

8.3.3. File Transfer, Access, and Management

Working with files in a distributed system (consisting of a number of individual systems connected together by a network) presents a challenge. Each system has its own filestore which may be different from the filestores of other participating systems (i.e., different filestore structure and different file formats). The ISO 8571 series of standards for File Transfer, Access, and Management (FTAM) provide useful services for addressing this problem.

FTAM provides file service uniformity by having each **real filestore** represented by a **virtual filestore**. The latter serves as a generic model for the former. Although the real file stores are system dependent and potentially different, the virtual file stores all adhere to the same format and protocol. FTAM furnishes the specification for virtual filestores. The mapping between a virtual filestore and its corresponding real filestore is beyond the scope of FTAM; it is the responsibility of the system which own the real filestore.

FTAM provides the necessary services for applications to establish associations with remote file system to: transfer files, access and manipulate file contents (read and write records), manage files (create, delete, copy, move, etc.), and manipulate file attributes (name, access permissions, dates, etc.).

Figure 8.89 illustrates FTAM's role in a typical configuration, where a local application accesses the filestore of a remote system. FTAM provides an association between an **initiator** and a **responder** which make the FTAM services accessible to the local and remote systems.

For its virtual filestores, FTAM uses a model which resembles the UNIX file system model. It views the whole file system as a tree-like hierarchical structure, where the tree nodes represent directories and the leaves represent actual files or references (similar to symbolic links in UNIX) to other nodes. Each of these is represented by an FTAM **object** (i.e., file object, directory object, and reference object).

A file itself consists of **attributes** and **contents**. The file attributes denote its various properties, such as name, creation date, and size. FTAM classifies attributes into four groups; these are summarized in Figure 8.89. The file contents is comprised of a set of named **Data Units** (DUs). DUs are combined in a hierarchical manner to build larger structures called **File Access Data Units** (FADUs). Figure 8.**Error!**

Bookmark not defined. illustrates this for a simple file structure consisting of four DUs organized as five FADUs.

Figure 8.88 FTAM structure.

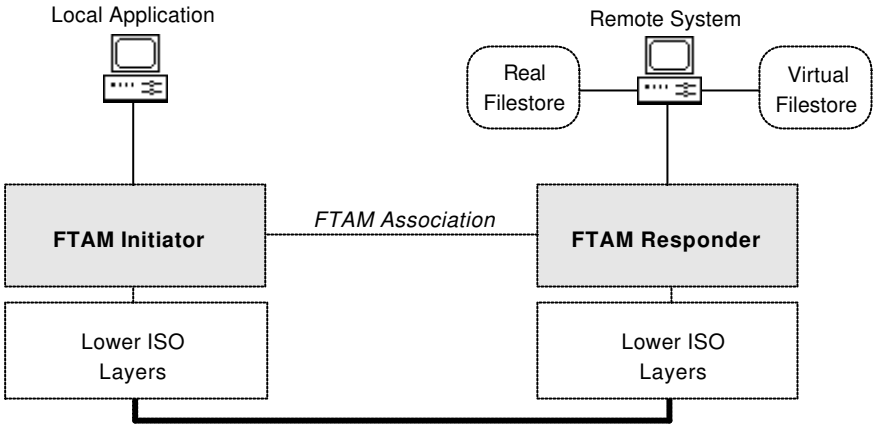
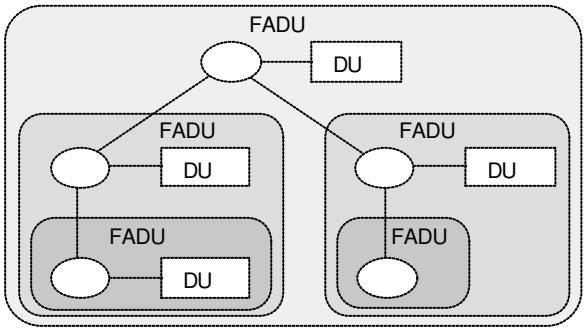


Figure 8.89 File attribute groups.

Group	Attributes	Description
Kernel	File Name File Structure File Mode	Basic properties required by all files.
Storage	File Size Creator ID User ID Read Date Write Date Attribute Change Date File Locks	Properties pertaining to the physical file storage.
Security	Access permission Encryption	Properties which regulate access to the file.
Private	*user-defined*	Properties beyond the scope of FTAM.

Figure 8.90 Sample FTAM file structure.



FTAM services are defined using service primitives, which operate on FADUs, DUs, and file attributes, as well as managing the initiation and termination of FTAM associations. FTAM defines many primitives, a sample of which is provided in Figure 8.11. FTAM uses ACSE for its operation.

Figure 8.11 Sample FTAM service primitives.

Primitive	Types	Purpose
F-INITIALIZE	request, indicate response, confirm	Used by an initiator-responder pair to establish an association.
F-TERMINATE	request, indicate response, confirm	Used by the initiator-responder pair to terminate their association.
F-U-ABORT	request, indicate	Used by an initiator-responder pair to report errors.
F-P-ABORT	indicate	Used by the service provider to report errors.
F-CREATE	request, indicate response, confirm	Used for creating a remote file.
F-DELETE	request, indicate response, confirm	Used for deleting a remote file.
F-SELECT	request, indicate response, confirm	Used for choosing a remote file to work with.
F-DESELECT	request, indicate response, confirm	Used for releasing control over a selected remote file.
F-COPY	request, indicate response, confirm	Used for making a copy of a remote file. The copy will remain on the remote system.
F-MOVE	request, indicate response, confirm	Used for moving a remote file within its filestore.
F-OPEN	request, indicate response, confirm	Used for opening a remote file so that operations can be applied to its contents.
F-CLOSE	request, indicate response, confirm	Used for closing an open remote file.
F-LOCATE	request, indicate response, confirm	Used for locating an FADU of an open remote file.
F-ERASE	request, indicate response, confirm	Used for erasing an FADU of an open remote file.
F-READ	request, indicate	Commences the reading of an open remote file or part thereof.
F-WRITE	request, indicate	Commences the writing of an open remote file or part thereof.
F-DATA	request, indicate	Used for transferring an open remote file or part thereof.
F-DATA-END	request, indicate	Finalizes one or more data transfer with F-DATA.
F-TRANSFER-END	request, indicate response, confirm	Finalizes an F-READ or F-WRITE operation in progress.
F-GROUP-*primitive*	request, indicate response, confirm	A set of primitives used for collectively working on groups of files (e.g., F-GROUP-DELETE).

FTAM service primitives are organized into four **regimes**. The regimes denote successive stages in an FTAM association which progressively narrow down the scope of the primitives. Figure 8.12. outlines the FTAM regimes and how they are

related to FTAM service primitives. Each regime comes into effect from the previous regime by certain primitives (denoted by the *Begin* column), and returns to the previous regime by another set of primitives (denoted by the *End* column). The *Do* column denotes the primitives that can occur while a regime is in effect.

Figure 8.12 FTAM regimes.

Regime	Begin	Do	End	Purpose
Application Association	F-INITIALIZE	F-GROUP-*primitive*	F-TERMINATE F-U-ABORT F-P-ABORT	This is the top-level regime and exists for the duration of FTAM association.
Object Selection	F-CREATE F-SELECT	F-COPY F-MOVE	F-DELETE F-DESELECT	Effective while the scope of association is a single file.
File Open	F-OPEN	F-LOCATE F-ERASE	F-CLOSE	Effective while a file is open for manipulation.
Data Transfer	F-READ F-WRITE	F-DATA F-DATA-END	F-CANCEL F-TRANSFER-END	Effective while a read or write operation is in progress.

FTAM services are also organized into related groups so that their selection can be negotiated when an association is being established. This grouping is done at two levels. The first level organizes the services into **functional units** (similar to functional units for earlier OSI layers). The second level organizes the functional units into **service classes**. FTAM defines five service classes:

- The **file transfer class** supports association management and the transfer of files between an application and a remote file system.
- The **file management class** supports association management and the remote management of files (e.g., creation, deletion, attribute manipulation) from an application.
- The **file access class** supports association management and the remote manipulation of file contents from an application.
- The **file transfer and management class** is the combination of services from the first two classes.
- The **unrestricted class** support user-selected functional units in an arbitrary fashion.

The ISO 8571 series of standards which describe all aspects of FTAM consist of a number of parts, four of which were touched upon in this section:

ISO 8571 .1 FTAM General Introduction
ISO 8571 .2 FTAM Virtual Filestore
ISO 8571 .3 FTAM File Service Definition

8.4. Other Standards

A number of application layer standards have been discussed so far in this chapter. Many more exist, including the following:

- The CCITT X.700 series of recommendations (and their corresponding ISO series) for System Management describe standards for coordinating resource utilization in networked information systems.
- The CCITT X.500 series of recommendations (and their corresponding ISO series) for Electronic Directory define standards for providing a global directory service with entries for network users as well as network components.
- The ISO 10026 standard for Distributed Transaction Processing is aimed at providing standard capabilities for distributed applications to simultaneously perform transactions on multiple systems.
- The ISO 8831 and ISO 8832 standards for Job Transfer Manipulation address remote job submission in distributed systems.

8.5. Further Reading

General descriptions of the application layer and related protocols can be found in Tanenbaum (1989), Black (1989), Stamper (1991), and Stallings (1994). Chilton (1989), Plattner *et al* (1991), and Radicati (1992) are two highly readable introductions to MHS and X.400.

9. Local Area Networks

Local Area Networks (LANs) have become an important part of most computer installations. Personal computers have been the main driving force behind the LAN proliferation. As personal computers became more widely used in office environments, so it became desirable to interconnect them to achieve two aims: to enable them to exchange information (e.g., e-mail), and to enable them to share scarce and expensive resources (e.g., printers). LANs have been so successful in realizing these aims that their cost is well justified even when there are only a handful of participating computers.

Figure 9.93 illustrates the main characteristics of LANs.

Current LANs are used for interconnecting almost any type of computing devices imaginable, including mainframes, workstations, personal computers, file servers, and numerous types of peripheral devices. Many LANs are further connected to other LANs or WANs via bridges and gateways, hence increasing the reach of their users.

In this chapter we will first look at some basic LAN concepts, and then discuss a number of widely-adopted LAN standards. As before, our aim will be to concentrate on general principles and protocols of importance rather than to get involved in the details of vendor-specific products.

Figure 9.93 LAN characteristics.

	Low End	Typically	High End
Number of Users	10s of users	20-100 users	1000s of users
Geographic Coverage	100s of meters	100-1000 meters	10s of km's
Data Rates	10s of kbps	1-10 mbps	100s of mbps
Raw Error Rates	1 bit in 100 million	1 bit in 1-10 billion	1 bit in 100 billion

9.1. Basic Concepts

A LAN consists of four general types of components:

- **User station**. This provides the user with access to the LAN. The most common example is a personal computer. The user station runs special network software (usually in form of a driver) for accessing the LAN.

- **LAN protocol stack**. This implements the LAN protocol layers. This usually takes the form of a hardware card inside the user station, containing a microprocessor and firmware which implements the non-physical protocols.
- **Physical Interface Unit**. This directly interfaces the user station-based LAN hardware to the LAN physical medium. The exact form of the PIU is highly dependent on the LAN physical medium. Coaxial cable connectors and cable TV taps are common examples.
- **Physical Medium**. This provides a physical path for signals to travel between stations. Coaxial cable, optical fiber, and infra red light are examples.

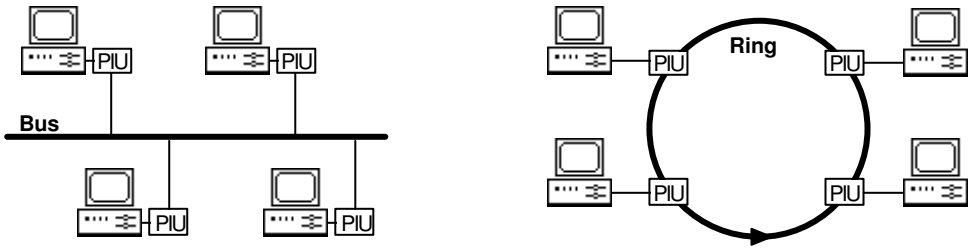
  The LAN protocol stack will be the main focus of our attention in this chapter.

9.1.1. Topologies and Access Protocols

There are two general categories of LAN topologies: bus and ring (see Figure 9.94). The **bus** topology uses a broadcast technique, hence only one station at a time can send messages and all other station listen to the message. A listening station examines the recipient address of the message and if it matches its own address, copies the message; otherwise, it ignores the message.

The **ring** topology uses a closed, point-to-point-connected loop of stations. Data flows in one direction only, from one station to the next. As with the bus topology, transmission is restricted to one user at a time. When a station gains control and sends a message, the message is sent to the next station in the ring. Each receiving station in the ring examines the recipient address of the message and if it matches its own address, copies the message. The message is passed around the ring until it reaches the originator which removes the message by not sending it to the next station.

Figure 9.94 LAN topologies.



Given that access to the bus or ring is restricted to one station at a time, some form of arbitration is needed to ensure equitable access by all stations. Arbitration is imposed by **access protocols**. A number of such protocols have been devised:

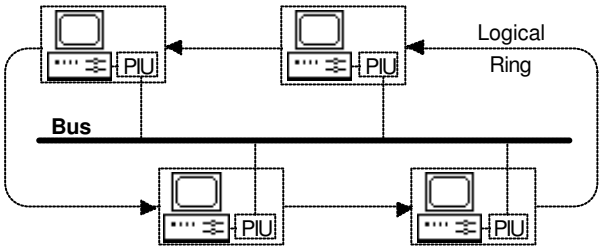
- **Carrier Sense**. This protocol is applicable to a bus topology. Before a station can transmit, it listens to the channel to see if any other station is already

transmitting. If the station finds the channel idle, it attempt to transmit; otherwise, it waits for the channel to become idle. Because of an unavoidable delay in a station's transmission to reach other stations, it is possible that two or more stations find the channel idle and simultaneously attempt to transmit. This is called a **collision**. Two schemes exist for handling collisions:

- **Collision Detection.** In this scheme a transmitting station is required to also listen to the channel, so that it can detect a collision by observing discrepancies in the transmission voltage levels. Upon detecting a collision, it suspends transmission and re-attempts after a random period of time. Use of a random wait period reduces the chance of the collision recurring.
- **Collision Free**. This scheme avoids collisions occurring in the first place. Each station has a predetermined time slot assigned to it which indicates when it can transmit without a collision occurring. The distribution of time slots between stations also makes it possible to assign priorities.

- **Token Ring**. This protocol is applicable to a ring topology. Channel access is regulated by a special message, called a **token**, which is passed around the ring from one station to the next. The state of the ring is encoded in the token (i.e., idle or busy). Each station wishing to transmit needs to get hold of the idle token first. When a station gets hold of the idle token, it marks it as busy, appends to it the message it wishes to transmit, and sends the whole thing to the next station. The message goes round the ring until it reaches the intended recipient which copies the message and passes it on. When the message returns to the originator, it detaches the message, marks the token as idle and passes it on. To ensure fair access, the token should go round the ring, unused, at least once before it can be used by the same station again.
- **Token Bus**. This protocol is applicable to a bus topology but makes it behave as a ring. Each station on the bus has two other stations designated as its logical predecessor and its logical successor, in a way that results in a logical ring arrangement (see Figure 9.95). A special message is provided which plays the role of a token. Each station receives the token from its predecessor, re-addresses it to its successor, and retransmits it on the bus. The rest of the protocol is as in a token ring.

Figure 9.95 Token bus arrangement.



9.1.2. Architecture

Figure 9.4 depicts the LAN protocol layers in relation to the OSI model. The role of the physical layer is the same as in the OSI model. It includes the connectors used for connecting the PIU to the LAN and the signaling circuitry provided by the PIU. (The next section describes the transmission methods employed by this layer.)

The OSI data link layer is broken into two sublayers. The **Media Access Control** (MAC) layer is responsible for implementing a specific LAN access protocol, like the ones described earlier. This layer is therefore highly dependent on the type of the LAN. Its aim is to hide hardware and access protocol dependencies from the next layer. As we will see shortly, a number of MAC standards have been devised, one for each popular type of access protocol.

The **Logical Link Control** (LLC) layer provides data link services independent of the specific MAC protocol involved. LLC is a subset of HDLC and is largely compatible with the data link layer of OSI-compatible WANs. LLC is only concerned with providing Link Service Access Points (LSAPs). All other normal data link functions (i.e., link management, frame management, and error handling) are handled by the MAC layer.

Figure 9.96 LAN protocol architecture.

OSI Layer	LAN Layer	Purpose
higher layers	*undefined*	Application dependent.
Data Link	Logical Link Control	Provides generic data link services to higher layers.
	Media Access Control	Implements the protocol for accessing the LAN.
Physical	Physical	Transmission of data bits over the channel.

LANs are not provided with a network layer (or any other higher layer) because such a layer would be largely redundant. Because the stations are directly connected,

there is no need for switching or routing. In effect, the service provided by the LLC is equivalent to the OSI network layer service.

9.1.3. Transmission

LAN transmission techniques are divided into two categories: baseband and broadband. In the **baseband** technique, the digital signal from a transmitting device is directly introduced into the transmission medium (possibly after some conditioning).

In the **broadband** technique, a modem is used to transform the digital signal from a transmitting device into a high frequency analog signal. This signal is typically frequency multiplexed to provide multiple FDM channels over the same transmission medium.

Baseband is a simple and inexpensive digital technique. By comparison, broadband has additional costs: each device requires its own modem; also, because transmission is possible in one direction only, two channels typically need to be provided, one for either direction. Broadband, however, has the advantages of offering a higher channel capacity which can be used for multiplexing data from a variety of sources (e.g., video, voice, fax), not just digital data. It is also capable of covering longer distances, typically tens of kilometers compared to up to a kilometer for baseband.

9.2. IEEE 802 Standards

The IEEE 802 series of recommendations provide a widely-accepted set of LAN standards. These recommendations are formulated by nine subcommittees (see Figure 9.97).

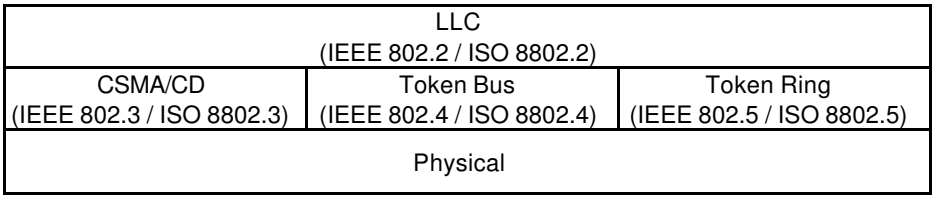
Figure 9.97 IEEE 802 Subcommittees.

Subcommittee	Title	Purpose
802.1	High-level Interface	Specification of standards for LAN architecture, interconnection, management
802.2	Logical Link Control	Specification of standards for the LLC layer
802.3	CSMA/CD	Specification of standards for CSMA/CD architectures
802.4	Token Bus	Specification of standards for token bus architectures
802.5	Token Ring	Specification of standards for token ring architectures
802.6	Metropolitan Area Networks	Specification of standards for MANs
802.7	Broadband Technical Advisory Group	Provision of guidance to other groups working on broadband LANs
802.8	Fiber Optic Technical Advisory Group	Provision of guidance to other groups working on fiber optic-based LANs

802.9	Integrated Data and Voice Networks	Specification of standards for interfaces to ISDN

Figure 9.98 illustrates the relationships between the main widely-used IEEE LAN recommendations. These are separately discussed below.

Figure 9.98 Main IEEE LAN Recommendations.



9.2.1. Logical Link Control

LLC is specified by the IEEE 802.2 and ISO 8802.2 standards. It provides link services to LAN users, independent of the MAC protocol involved. LLC offers three types of service:

- **Unacknowledged connectionless service**. This service must be provided by all 802.2 implementations. It is based on data being transferred in independent data units, the delivery of which is neither guaranteed, nor acknowledged. Furthermore, there are no provisions for ordered delivery of data units or for flow control. Obviously, a higher-level protocol is needed to make this service reliable.
- **Connection-oriented service**. This service is based on the use of logical connections. Data is transferred using ordered, acknowledged, and flow controlled data units. Transmission errors are detected and reported.
- **Acknowledged connectionless service**. Same as the unacknowledged connectionless service, except that the delivery of each data unit is acknowledged before the next data unit is sent.

The LLC service is provided through a set of service primitives. Figure 9.99 summarizes these primitives.

Figure 9.99 LLC service primitives.

Primitive	Types	Parameters	Purpose
DL-CONNECT	request indicate response confirm	(addresses, priority)	Used for the connection-oriented service. Establishes a connection between two LSAPs.

DL-DISCONNECT	request indicate	(addresses)	Disconnects a connection between two LSAPs.
DL-DATA	request indicate	(addresses, data)	Used for the connection-oriented transfer of data units.
DL-CONNECTION-FLOW-CONTROL	request indicate	(addresses, data-quantity)	Used to flow control data transfers by specifying how much data can be accepted.
DL-RESET	request indicate response confirm	(addresses, reason)	Used to reset a (possibly in error) connection. All data units in transit will be lost.
DL-UNIT-DATA	request indicate	(addresses, data, priority)	Used for the connectionless unacknowledged transfer of data units.
DL-DATA-ACK	request indicate response confirm	(addresses, data, priority, status)	Used for the connectionless acknowledged transfer of data units. Status returns the delivery status of the data unit.

LLC also uses two primitives to communicate with the MAC layer. MA-UNIT-DATA is used to transfer data units between LLC and MAC. MA-DATA-UNIT-STATUS is used for acknowledgments.

The general structure of an LLC protocol data unit is shown in Figure 9.100. The header (*Source* and *Destination addresses* and *Control* field) are appended to user data by LLC before the PDU is passed onto the MAC layer. The addresses identify peer LLC users. The *Control* field is very similar in format and content to the control field of HDLC frames discussed in Chapter 3.

Figure 9.100 LLC PDU structure.

Field	Description
DSAP Address	Destination Service Access Point Address.
SSAP Address	Source Service Access Point Address.
Control	Control field (similar to HDLC control field).
Data	Actual user data.

9.2.2. CSMA/CD

The Carrier Sense Multiple Access with Collision Detection (CSMA/CD) protocol is specified by the IEEE 802.3 and ISO 8802.3 standards. CSMA/CD is based on the widely-publicized and well-adopted Ethernet specification, and offers data rates in order of 10 mbps using the baseband or the broadband technique. The overall behavior of the protocol is as described in Section 9.1.1.

The general structure of a CSMA/CD MAC frame is shown in Figure 9.101. It consists of a *Data* field (which is an LLC PDU) with a header and a trailer added at either end. The header provides synchronization, addressing, and length information.

The trailer provides a CRC-style *Frame Check Sequence* (*FCS*). CSMA/CD imposes a minimum frame size which in turn translates to a minimum data field size. Should the data field be shorter than required, it is padded with enough octets to achieve the required minimum.

Figure 9.101 CSMA/CD MAC frame structure.

Field	Description
Preamble	Special bit pattern for synchronization purposes.
Start Delimiter	Marks the beginning of frame.
Addresses	Source and destination addresses.
Length	Denotes the length of the LLC data unit in octets.
Data	Actual user data (i.e., LLC PDU).
Padding	Appended to the LLC data unit to ensure minimum length.
FCS	Frame Check Sequence.

9.2.3. Token Bus

The token bus protocol is specified by the IEEE 802.4 and ISO 8802.4 standards. The logical ring is determined by the descending numeric order of the station addresses. When a station takes possession of the token, it is given exclusive network access for a limited period of time. The station either transmits during this time window or hands over the token to its successor. The token holder can also poll other stations in order to learn about their status.

The protocol provides for token-related fault handling. When a station passes the token to its successor, it listens for transmissions on the bus for a period of time. A transmission would be a sign that the successor has successfully received the token and is either passing it on or is transmitting data. If the token sender establishes that the token has not been received after two attempts, it will attempt to bypass the presumably faulty station. To do this, it polls other stations to find out who is the next logical successor and will attempt to pass the token to that station.

The general structure of a token bus frame is shown in Figure 9.102. The beginning and end of the frame are marked by two delimiting octets. The *Frame Control* field is used for building different types of frames (e.g., token frame, polling frame). For some frames (e.g., the token frame) the *Data* field may be completely empty. The *Preamble*, *Addresses*, and *FCS* fields are as in CSMA/CD.

Figure 9.102 Token bus MAC frame structure.

Field	Description
Preamble	Special bit pattern for synchronization purposes.
Start Delimiter	Marks the beginning of frame.

Frame Control	field.
Addresses	Source and destination addresses.
Data	Actual user data (e.g., LLC PDU).
FCS	Frame Check Sequence.
End Delimiter	Marks the end of frame.

9.2.4. Token Ring

The token ring protocol is specified by the IEEE 802.5 and ISO 8802.5 standards. The most well-known realization of this protocol is the IBM token ring product.

The token ring protocol operates largely as described in section 9.1.1. A time limit is imposed on each station for holding the token and transmitting on the ring. The protocol includes a scheme for handling priority traffic, and for a station to assume a monitoring role to keep an eye on the status of the network. It is also possible to bypass an inactive or faulty station without disrupting ring traffic.

The general structure of a token ring frame is shown in Figure 9.103. The shaded fields appear in data frames but not in a token frame. As before, the start and end of the frame is delimited by two special octets. *Addresses*, *Data*, and *FCS* fields are as before. The *Access Control* field is an octet divided into four components:

- A **token** bit flag to indicate if this is a token frame.
- A **monitor** bit flag set by the monitor station during recovery.
- A three-bit **priority** field which can indicate up to eight token priority levels. Only a station which has a frame of equal or higher priority can gain control of the ring.
- A three-bit **reservation** field is used for implementing a reservation scheme for handling priority traffic. When a frame (i.e., busy token) is passing through a station which has a frame waiting to be transmitted, it can raise the reservation field value to the value of the priority of the frame. When the station finally gets hold of the token and transmits its frame, it restores the previous reservation value. This scheme ensures that higher priority frames have a better chance of being transmitted first, without totally denying lower-priority frames an opportunity to be also transmitted.

Figure 9.103 Token ring MAC frame structure.

Field	Description
Start Delimiter	Marks the beginning of frame.
Access Control	Various access-related flags and values.
Frame Control	Denotes the frame type.

Addresses	Source and destination addresses.
Data	Actual user data (e.g., LLC PDU).
FCS	Frame Check Sequence.
End Delimiter	Marks the end of frame.
Frame Status	Used by receiving station to record receipt status.

The *Frame Control* field specifies the frame type (i.e., whether it is an LLC PDU or some kind of MAC control frame). The *Frame Status* field is used by the receiving station to indicate that it has successfully received the frame and extracted its data, so that when the frame reaches the originator, it can be disposed of.

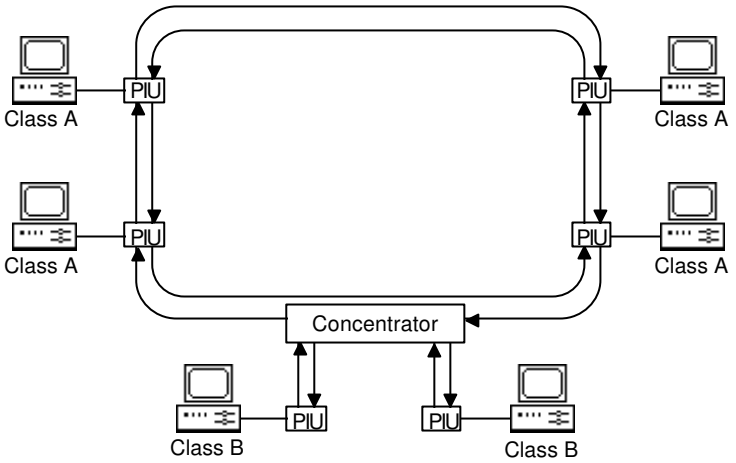
9.3. ANSI FDDI Standard

Fiber Distributed Data Interface (FDDI) is a high-speed LAN protocol designed by ANSI for use with optical fiber transmission media. It is capable of achieving data rates in order of 100 mbps, and a network size in order of 1000 stations. Furthermore, with the high reliability of optical fiber, station distances in order of kilometers and a geographic spread of the LAN in order of hundreds of kilometers are feasible.

9.3.1. Topology

FDDI utilizes a ring topology that involves two counter-rotating rings and two classes of stations (see Figure 9.104). **Class A** stations are connected to both rings. Each class A station also has a bypass switch (i.e., short circuit) which when activated causes the station to be excluded from the ring without affecting the ring continuity. **Class B** stations are only connected to one of the rings via a concentrator. The concentrator also provides a bypass switch. Furthermore, because each of the B stations is independently connected to the concentrator, it can be switched off without affecting the other stations. Typically class A represents the more important stations (e.g., servers) and class B represents the less significant stations (e.g., infrequently-used PCs and terminals).

Figure 9.104 FDDI ring topology.



9.3.2. Token Ring Protocol

The FDDI protocol is specified by the ANSI X3T9 standard. It is a token ring protocol similar to IEEE 802.5 but with an important difference: stations can transmit even if the token they receive is busy. When a station receives a frame, it examines the address of the frame to see if it matches its own address, in which case it copies the data. In either case, if the station has nothing to transmit, it passes the frame to the next station. If it does have a frame to transmit, it absorbs the token, appends its frame(s) to any existing frames and then appends a new token to the result. The whole thing is then sent to the next station in the ring. As with earlier token ring protocols, only the originating station is responsible for removing a frame.

Figure 9.105 illustrates the FDDI frame structure, which is almost identical to the IEEE 802.5 frame structure, except for the different physical size of the fields, and that it includes a *Preamble* and excludes the *Access Control* field.

Figure 9.105 FDDI frame structure.

Field	Description
Preamble	Special bit pattern for synchronization purposes.
Start Delimiter	Marks the beginning of frame.
Frame Control	Denotes the frame type.
Addresses	Source and destination addresses.
Data	Actual user data (MAC or LLC).
FCS	Frame Check Sequence.
End Delimiter	Marks the end of frame.
Frame Status	Used by receiving station to record receipt status.

The FDDI protocol requires that each station maintains three timers for regulating the operation of the ring:

- The **token holding timer** determines how long a transmitting station can keep the token. When this timer expires, the station must cease transmission and release the token.
- The **token rotation timer** facilitates the normal scheduling of token rotation between stations.
- The **valid transmission timer** facilitates recovery from transmission errors.

9.4. Further Reading

Introductory texts on LANs and MANs include: Kessler and Train (1992), Stallings (1993a), Stamper (1993), and Martin *et al* (1994). Other useful texts include: Chorafas (1989), Fortier (1992), and Hunter (1993). Zitsen (1990) and Layland (1994) discuss the internetworking aspects of LANs. Hegering and Lapple (1993) describe the Ethernet in detail. Gohring and Kauffels (1992) discuss token rings. Jain (1993) and Nemzow (1994) provide detailed descriptions of the FDDI technology.

10. Telephone Networks

By far the largest communication network on this planet is the global Public Switched Telephone Network (PSTN). This network brings together about a billion telephone sets in a highly connected fashion which allows almost every set to be reached by every other set. The global telephone network is the center-piece of the field of telecommunication.

Although the telephone system was originally designed to carry analog voice signals, the advent of computers and data communication has lead to it being increasingly used to also carry computer data traffic. At the same time, telephone networks have evolved to take advantage of the digital technology made available by computers. Modern telecommunication networks are designed to carry information in a variety of forms (e.g., voice, data, video, fax). Telecommunication and data communication are fast converging.

This chapter serves as an introduction to telecommunication. It looks at the various components of telephone networks and the signaling technology which helps operate them. Unlike data networks, telephone networks have been traditionally circuit-switched and largely remain so. More recently, the technology for packet switching has made it possible to also carry real-time conversations in packets. Our discussions in this chapter assume circuit switching.

We will first present some fundamental telecommunication concepts, and then discuss signaling. Signaling remains the most complex aspect of telecommunication networks. We will look at common channel signaling and its major standards: SS7. Finally, we will examine private telephone networks and their relevance to public networks.

10.1. Basic Concepts

The most familiar component of a telephone network is the telephone set provided for each user. It is a relatively simple device which can exchange control signals with the network to help establish and release calls, and to send and receive a user's speech signal.

Users are referred to as **subscribers**, because they subscribe to a service provided by the telephone company. Typically, many subscribers are attached to the same network, and each subscriber can contact every other subscriber. The system employs **switches** to facilitate connections between subscribers. The arrangement of

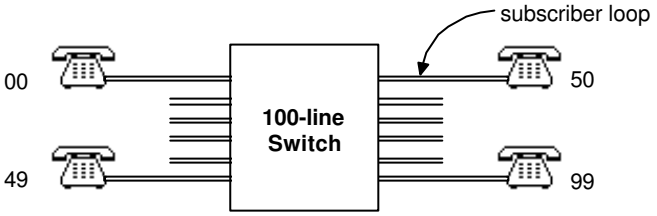
switches and subscribers and the interconnections between them determines the structure of the network.

To better understand how the system works, we will begin by looking at a very simple network.

10.1.1. A Simple Network

Consider a small community of 100 families. The communication needs of this community can be served by one switch, to which every family (or subscriber) is directly connected (see Figure 10.1). The circuit which connects each subscriber (via a pair of wires) to the switch is called a **subscriber loop**. The switch itself is called a **local exchange** (local because the connection is direct).

Figure 10.106 Single switch network.



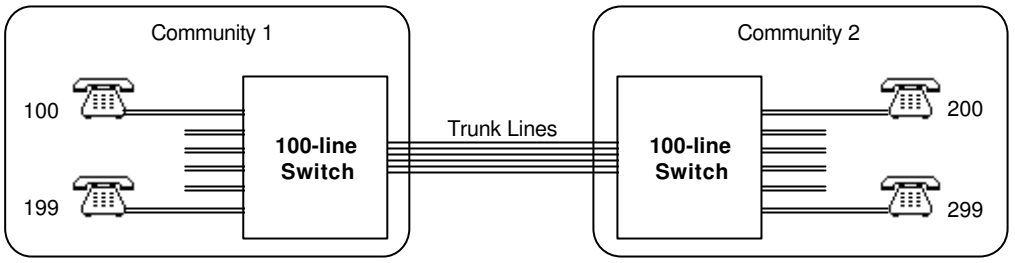
A numbering scheme is used to uniquely identify each subscriber and to gather call-related information for billing purposes. In this case, a two-digit number would suffice (i.e., 00-99). When a subscriber dials the number of another subscriber, the switch follows a sequence of steps for establishing a communication path between the two subscribers. Provided the destination number is not engaged, a connection is guaranteed.

Now suppose that there is a similar community with its own local exchange, and that we wish to provide phone access between the two communities. To do this, the two exchanges are connected using a set of interexchange lines called **trunk lines** (see Figure 10.2). In practice, it is unlikely that all of the subscribers in one community would want to simultaneously contact subscribers in the other community. If we establish that at most say six subscribers are likely to call the other community, then six trunk lines will be sufficient.

The numbering scheme is expanded to take into account the above changes. Each subscriber is now allocated a three digit number. The first digit identifies the exchange (say 1 for the first community and 2 for the second community) and the next two digits identify a subscriber in that community as before. When a subscriber dials a number, the local exchange looks at the first digit of the number. If it matches its own number then it treats the remaining two digits as for a subscriber on the same exchange, and attempts to establish a direct connection between the two as before. If the first digit identifies the other exchange, then it uses one of the trunk lines to

inform the other switch of the call and passes to it the two remaining digits. The two switches then cooperate to establish a call between the two subscribers.

Figure 10.107 A network with two switches.



If more than six subscribers attempt to simultaneously call subscribers on the other exchange, the seventh call and beyond will be blocked due to insufficient trunk lines.

10.1.2. Networks Topologies

The simple network described in the previous section can be further expanded by adding exchanges. This may lead to two different types of topologies: mesh and star (see Figure 10.**Error! Bookmark not defined.**). A **mesh** topology is one in which each exchange is directly connected to every other exchange. In the **star** topology, each exchange is directly connected to a tandem exchange. A **tandem** exchange does not directly serve subscribers, but provides for interconnection between other switches. All of its lines, therefore, are trunk lines.

The mesh configuration is suited to situations where there is heavy traffic between a set of local exchanges (e.g., in city centers with high business concentration). The star configuration is better suited to areas where there is not much traffic between the local exchanges (e.g., in suburbs). Most situations, however, are best served by a hybrid configuration which is primarily star-like but has additional trunk lines between exchanges where there is heavy traffic.

The star topology can be taken a step further by interconnecting tandem exchanges from a number of star configurations to form a higher order star. Repeating this process results in a **hierarchical** network which consists of a number of levels.

Figure 10.3 Basic network topologies.

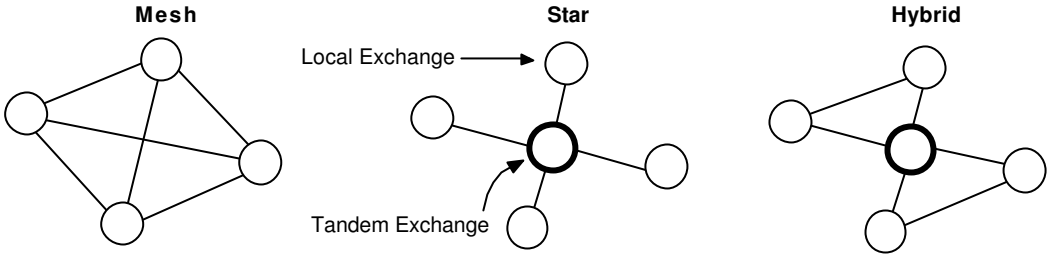
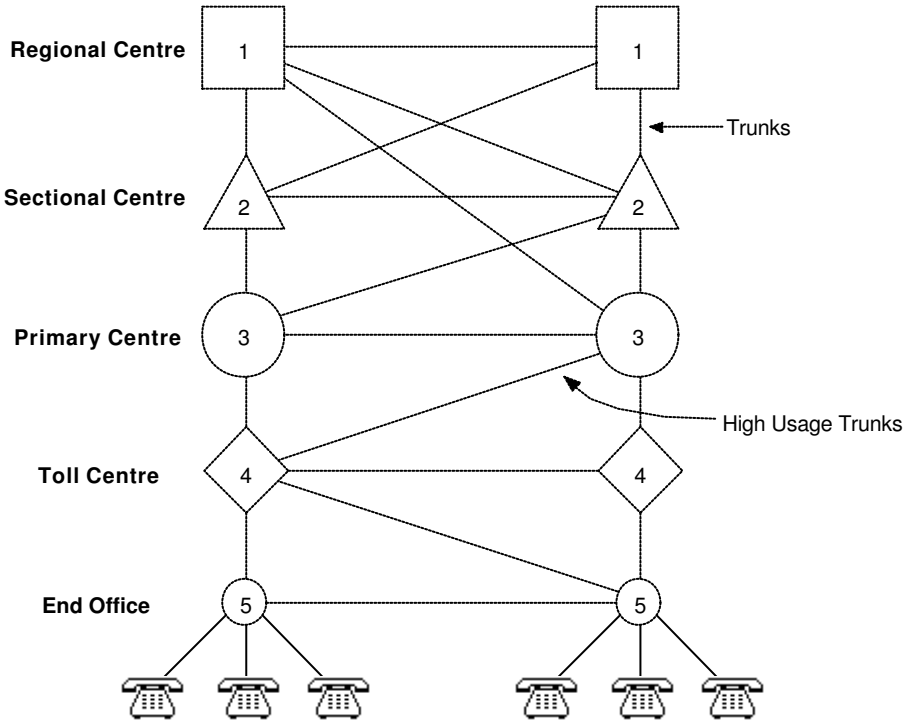


Figure 10.**Error! Bookmark not defined.** illustrates the North American hierarchical network configuration. It is made up of five classes of exchanges. A class 1 exchange represents a regional center and has the highest order in the hierarchy. A class 5 exchange represents an end office (local exchange) and has the lowest order in the hierarchy. Solid lines represent main trunk lines between exchanges. Dashed lines represent **high usage trunk** lines and can be established between any two exchanges, regardless of their levels in the hierarchy. Should the high usage trunks reach their maximum capacity, additional traffic is overflowed to the main trunks, from which further overflow is permitted. Because of this, the latter is also referred to as the **final route**.

Figure 10.4 Hierarchical network.



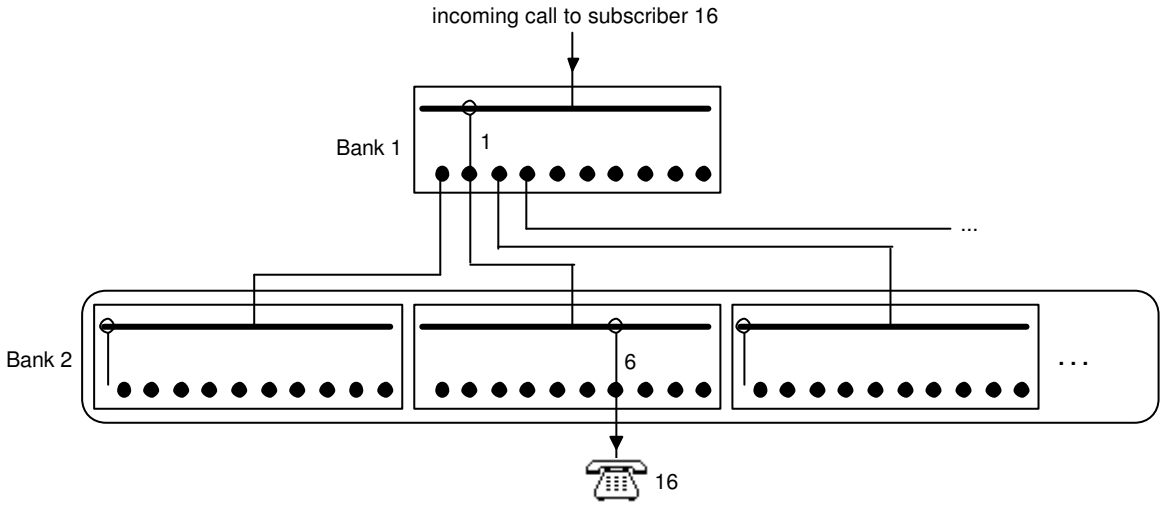
Hierarchical networks have the advantage of requiring relatively simple switch designs. Their main disadvantage is that loss of connection at higher order trunks may severely disrupt the traffic. To reduce this risk, the highest order exchanges are usually interconnected in a mesh-like fashion.

10.1.3. Switching Systems

Early telephone switches were **electromechanical**, many of which are still in use. They are so named because they use electromechanical relays to perform switching functions. The relays respond to the dial pulses from subscriber phones, and hence activate connections.

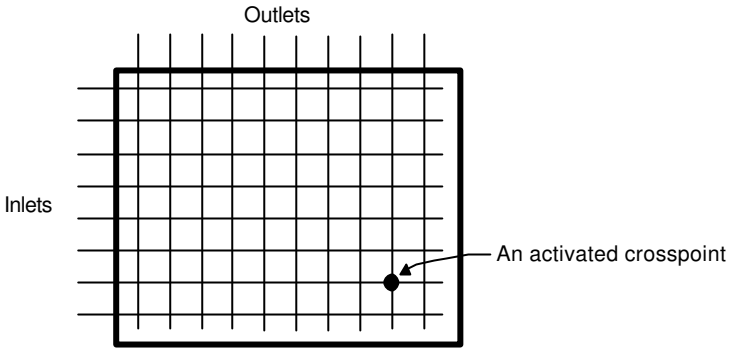
Existing electromechanical switches are of two types: step-by-step and crossbar. A **step-by-step switch** (also called Strowger after its inventor) uses step relays capable of assuming ten separate levels, to represent a decimal digit. For example, each of the two switches in Figure 10.**Error! Bookmark not defined.** would consist of two banks of relays, arranged in a tree-like fashion. Figure 10.**Error! Bookmark not defined.** illustrates how an incoming call to a subscriber whose number is 16 is connected.

Figure 10.5 A step-by-step switch.



A **crossbar switch** consist of a matrix of crosspoints between rows of inlets and columns of outlets. Activating the relay of an inlet and the relay of an outlet causes the crosspoint at which they overlap to be activated and therefore result in a physical connection (see Figure 10.**Error! Bookmark not defined.**).

Figure 10.6 A crossbar switch.



Modern switches are microprocessor controlled and use digital switching technology. They are generally referred to as **Stored Program Control** (SPC) switches. Unlike electromechanical switches which are inherently hardwired and therefore inflexible, SPC switches achieve significant flexibility by using programmable logic. Because there are no moving parts involved, connections can be established orders of magnitude faster. Furthermore, the functions of the switch can be reconfigured through software (even remotely). Software control has facilitated the introduction of many new service features which were previously beyond the scope of electromechanical switches.

10.2. Signaling

Signaling refers to the exchange of control information between the components of a network (telephones, switches, etc.) in order to establish, manage, and disconnect calls. Four different types of signaling are used by telephone networks:

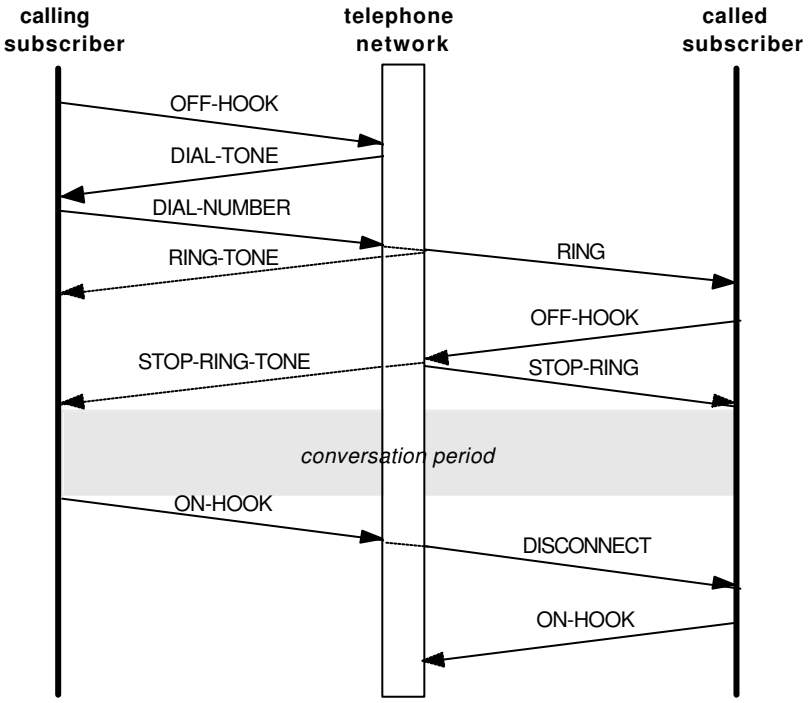
- **Supervisory**. This type of signaling provides the necessary control and status signals to establish calls, release calls, and make other service features possible. It includes the informing of exchanges about subscriber loop on-hook/off-hook conditions, and providing information about the status of calls.
- **Address**. This type of signaling conveys addressing information (subscriber number, area code, access code) between network components.
- **Network Management**. This type of signaling supports the management of network resources. It includes the handling of congestion and component failure situations, and the gathering and reporting of useful status information such as traffic conditions and operating anomalies.
- **Audio-Visual**. This type of signaling informs the calling subscriber about the status of a call, and alerts the called subscriber about a waiting call.

Two categories of signaling are discussed below: subscriber signaling and interexchange signaling.

10.2.1. Subscriber Signaling

Subscriber signaling refers to the signals exchanged between a subscriber and a local exchange via the subscriber's loop. The sequence diagram in Figure 10.**Error! Bookmark not defined.** illustrates (a simplified version of) the signals exchanged for successfully establishing a call between two subscribers.

Figure 10.7 Sample subscriber signaling scenario.



This process works as follows. We assume that initially both subscribers have their phones on-hook. The calling subscriber sends an off-hook signal to the local exchange by lifting the receiver. The switch responds by activating an audible dial tone over the subscriber loop, informing the subscriber that a number may be dialed now. The subscriber dials the other party's number, and each dialed digit is signaled to the local exchange. This number is used as an address to route the call through the network and finally arrives at the local exchange of the called party. This local exchange applies a ringing signal to the called subscriber's loop. At the same time, a confirmation is sent back to the calling subscriber's local exchange, which in turn applies a ring tone to the calling subscriber's local loop. This serves as an audible feedback that the number is dialed. When the called subscriber lifts the receiver, an off-hook signal is sent back to its local exchange. This causes the local exchange to

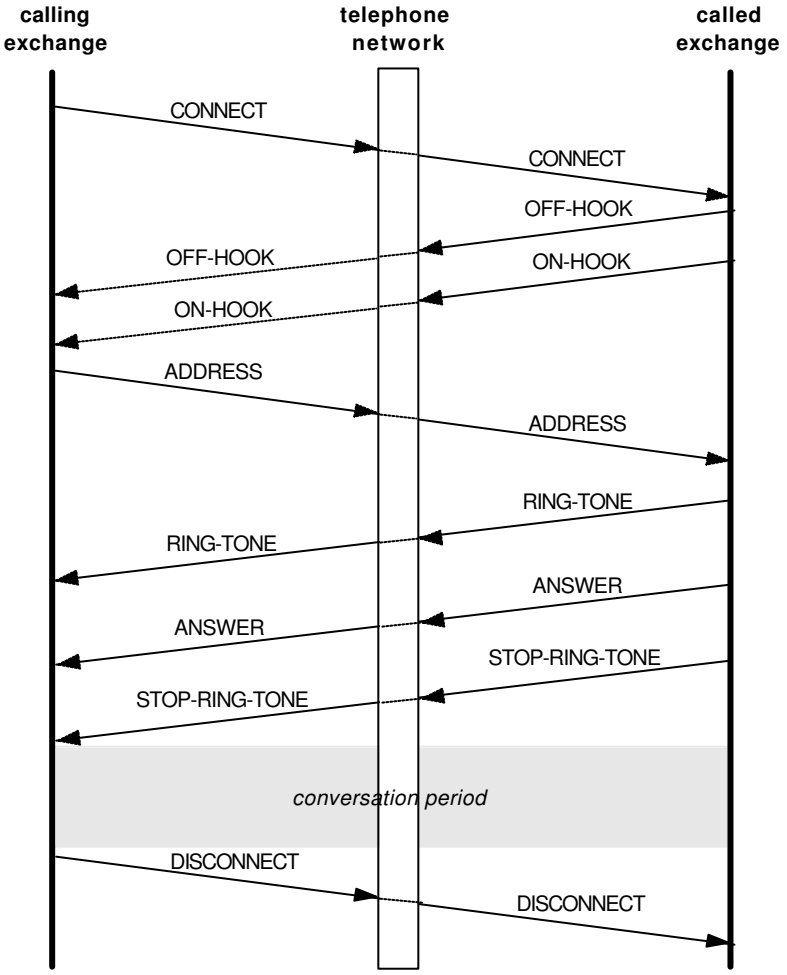
stop the ringing, and is propagated back to the calling subscriber's exchange which in turn stops the ring tone. The network connects the two parties and they may engage in a conversation. Either subscriber can terminate the call by pressing the hook switch. This sends an on-hook signal to the local exchange, which in turn terminates the call and informs the other subscriber accordingly.

10.2.2. Interexchange Signaling

Interexchange signaling refers to the signals exchanged between two or more network exchanges in order to handle calls. The sequence diagram in Figure 10.**Error! Bookmark not defined.** illustrates (a simplified version of) the signals exchanged between two local exchanges via the trunk lines which may involve zero or more tandem exchanges. The scenario is the same as that of Figure 10.**Error! Bookmark not defined.**, except that here we are looking inside the network.

This process works as follows. After the calling subscriber has dialed the number of the called subscriber and this address has been provided to the calling exchange, the latter conveys a connect signal to the local exchange of the called subscriber via the network. The called exchange *winks* in response to this by sending an off-hook followed by an on-hook signal, confirming that it is ready to receive the address. Once the address is communicated to the called exchange, it starts ringing the called subscriber and at the same time sends a ring tone to the calling subscriber's phone. This feedback confirms to the calling subscriber that the called subscriber's phone is ringing. When the called subscriber answers, this is signaled back to the calling exchange and the ring tone is terminated to confirm that the call has been answered. The two parties can now engage in a conversation. As soon as either subscriber hangs up the phone, a disconnect signal is communicated by its local exchange to the local exchange of the other party and the call is terminated.

Figure 10.8 Sample interexchange signaling scenario.



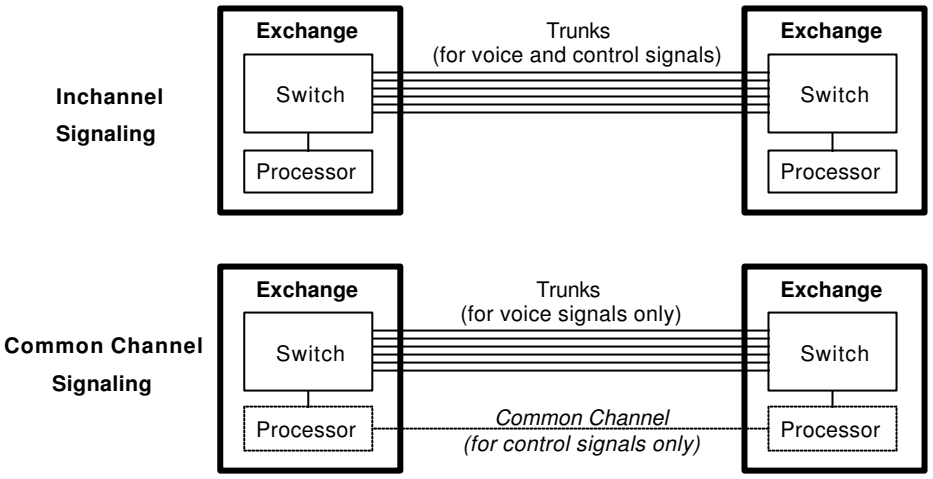
10.2.3. Common Channel Signaling

There are two general methods for conveying control signals between the components of a network: inchannel signaling and common channel signaling. With **inchannel signaling** Inchannel the control signals occupy the same channel as the voice signal. It comes in two forms: **in-band signaling** uses audio tones for conveying its signals, **out-of-band signaling** reserves a narrow band within the voice band for conveying control signals. Inchannel signaling has the advantage of using the same trunk lines and equipment for carrying control signals as for voice signals. It has the disadvantage of being very limited both in speed and richness of signals that can be conveyed.

Common Channel Signaling (CCS) overcomes these limitations by using a channel separate from the voice channel for carrying the control signals. Because control signals have a lower bandwidth requirement than voice signals, the same

control channel can be used for carrying the control signals of multiple voice channels, hence the term *common channel*. Figure 10.**Error! Bookmark not defined.** compares inchannel signaling to CCS.

Figure 10.9 Inchannel versus common channel signaling.



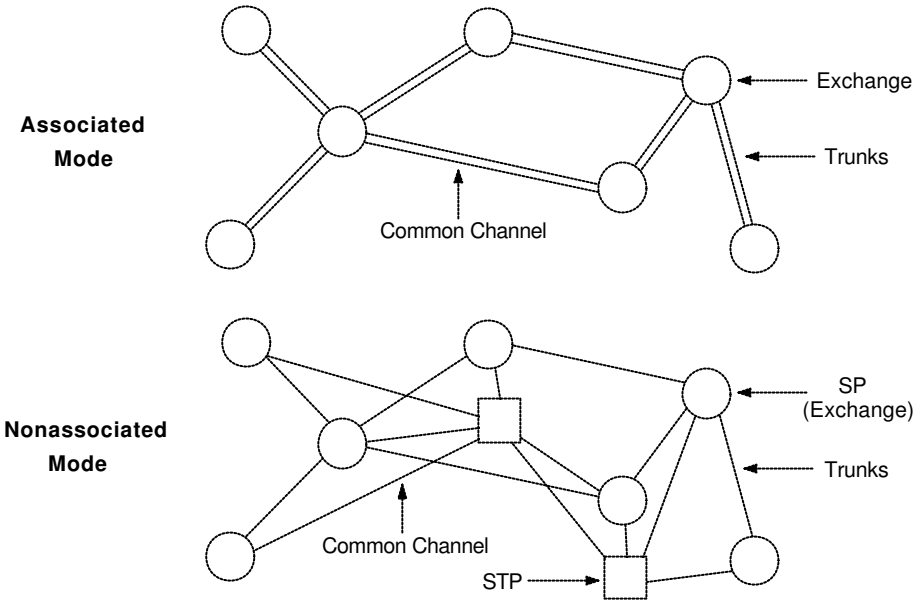
Because the two signal types are physically separated and independently handled, the overall network now consists of two networks: a voice network for carrying speech, and a distributed data network for carrying control signals. The relationship of these two networks results in two modes of operation: associated and nonassociated.

In the **associated mode** the common channel is closely associated with the trunks so that the control signals for a call follow the same route as the voice signal. In the **nonassociated mode** the exchanges act as the Signaling Points (SP) as before, but additional **Signal Transfer Points** (STPs) are introduced to implement a signaling network of different topology to the voice network. As a result, the control signals no longer necessarily follow the same route as the voice signals. This produces two independent networks with links at the switching points. Figure 10.**Error! Bookmark not defined.** compares the two modes.

The nonassociated mode, while considerably more complex, offers all the flexibility of data networks for routing the control signals. For example, it is now possible to route call signaling information to a central point where information can be looked up in a database to obtain a service profile for handling the call.

The Common Channel Interoffice Signaling (CCIS) system in North America which is based on the CCITT Signaling System Number 6 (SS6) is a living example of common channel signaling. CCIS supports both the associated and the nonassociated mode of operation. It uses packet switching for its control signaling. Each packet consists of a set of Signal Units (SUs) for conveying signaling information.

Figure 10.10 CCS modes of operation.



The limited line speed of CCIS and SS6 (4.8 kbps and 2.4 kbps, respectively) and the limited size of the SUs has lead to the development of a replacement for SS6. This is discussed in the next section.

10.3. Signaling System Number 7

CCITT's Signaling System Number 7 (SS7) was designed with the requirements of digital telephone networks in mind. As a common channel signaling standard, it is suitable for use with a wide range of circuit-switched digital networks. Like its predecessor, SS6, it uses packet switching, but is primarily designed to work with 64 kbps digital channels of modern exchanges. However, it can also operate over analog channels of lower speeds, as well as digital channels of higher speeds.

SS7 is a very large and complex system. It is the signaling system of choice for ISDN (discussed in the next chapter). The CCITT Q.700 series of recommendations which is comprised of numerous parts is entirely devoted to the definition of SS7.

Since SS7 is a data network (similar to X.25, but designed for the specific application of signaling), it is useful to compare its protocol architecture to the OSI reference model. This is illustrated in Figure 10.**Error! Bookmark not defined.**. Each level is separately discussed below.

Figure 10.11 SS7 protocol architecture.

OSI Layer	SS7 Level		Purpose
higher layers	4	Operations and Maintenance Applications Part	Provides network management functions for the operation, administration, and maintenance of the network.
Network		User Parts	Deals with signaling data contents.
		SCCP	Provide additional network services.
	3	Signaling Network Functions	Routing of signaling data through a set of signal transfer points.
Data Link	2	Signaling Link Control	Reliable, ordered delivery of signal units over a signaling data link.
Physical	1	Signaling Data Link	Transmission of signaling data bits over the physical channel.

10.3.1. Signaling Data Link

This layer maps to the OSI physical layer and is responsible for the transmission of signaling data bits over a full-duplex physical channel. Although SS7 has been designed to work with 64 kbps digital channels, it can also operate with higher and lower speed channels as well as analog channels (via modems).

10.3.2. Signaling Link Control

This layer maps to the OSI data link layer and is responsible for providing a reliable data link for exchanging signal units. It ensures that transmitted signal units are delivered in order, without loss or duplication. It also provides flow control capabilities. In its operation, this layer is very similar to HDLC (see Chapter 3). The main difference is in message formats.

Figure 10.**Error! Bookmark not defined.** illustrates the general structure of a signal unit (equivalent to a frame in HDLC). The signal unit is delimited in exactly the same fashion as a HDLC frame. The *BSN* and *FSN* correspond, respectively, to HDLC receive and send sequence numbers. These are used to implement an error control (Go-Back-N) mechanism for dealing with transmission errors, and a flow control (sliding window) mechanism for dealing with congestion situations. The *Data Length* field specifies the length of the following *Data* field (which contains information used by levels 3 and 4) in octets. The *Checksum* field is a 16-bit CRC over the whole unit except for the flags and the CRC field itself.

Figure 10.12 Signal unit structure.

Field	Description
01111110	Start flag: marks the beginning of the signal unit.
BSN & Flag	Backward Sequence Number and associated flag.
FSN & Flag	Forward Sequence Number and associated flag.
Data Length	Length of the following data field in octets.
Data	Contains the data for levels 3 and 4.
Checksum	An error checksum field for error detection.
01111110	End flag: marks the end of the signal unit.

There are three possible structures for the *Data* field, leading to three different signal unit types:

- **Message Signal Unit** (MSU). This is used for carrying signaling information from higher levels. The data field consists of a Service Information Octet (SIO) and a Signaling Information Field (SIF). The SIO denotes the role of the MSU (i.e., what type of message it is and what type of network it relates to). The SIF consists of source and destination message addresses, a Signaling Link Selection (SLS) field, and user data from a higher level entity.
- **Link Status Signal Unit** (LSSU). This is used for carrying signaling link control information. The data field consists of a Status Field (SF) which is used to communicate the link status between signaling points, and may be used by network management entities. One major use of LSSUs is for flow control.
- **Fill-in Signal Unit** (FISU). This is used for continued transmission in absence of other signals. The data field is empty.

10.3.3. Signaling Network Functions

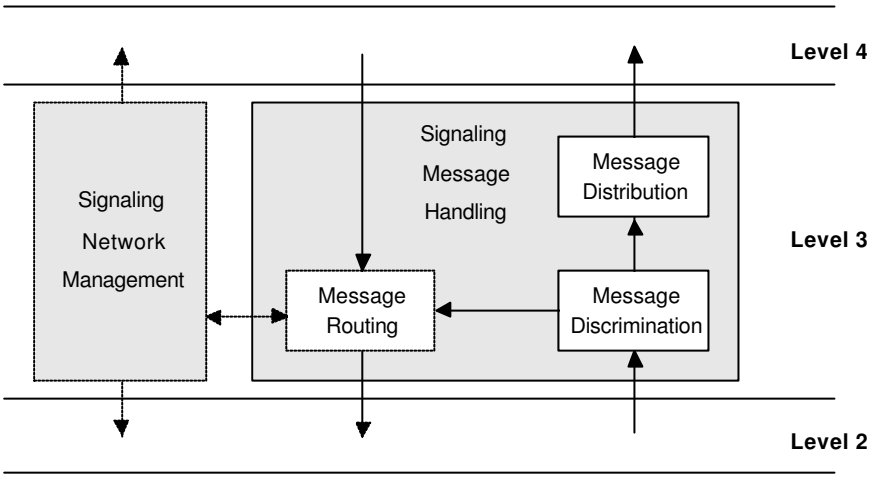
Figure 10.**Error! Bookmark not defined.** illustrates the SS7 signaling network functions, which consist of two groups: signaling message handling functions and signaling network management function.

The **signaling message handling** functions are responsible for communicating signaling messages between two signaling points. These consist of three main functions:

- The **message discrimination** function looks at the SIF field of the MSU to decide if the message has reached its final destination (i.e., its destination address matches this signaling point) or it should be routed further. If the latter is the case, the MSU is handed over to the routing functions. Otherwise, it is handed over to the distribution function for local consumption.

- The **message routing** function handles the further routing of an MSU by choosing a signaling link to forward it on. Routing is facilitated by a **routing label** which consists of the source and destination addresses and the SLS field. The SLS guides link selection so that the load is shared between alternate links.
- The **message distribution** function accepts an MSU destined for this signaling point and uses the SIO field to determine the user part the message should be forwarded to.

Figure 10.13 Signaling network functions.



The **signaling network management** function is responsible for monitoring and maintaining the signaling service by overcoming signaling failures and by congestion control. It provides re-routing information to signaling points and controls the activation and deactivation of links in response to changed link status.

SS7 levels 1-3 (i.e., signaling data link, signaling link control, and signaling network functions) are collectively referred to as the **Message Transfer Part** (MTP). The MTP provides a highly reliable connectionless (datagram) service. It is described by CCITT recommendations Q.701-Q.710. Methods for monitoring and measuring MTP performance are described by CCITT recommendation Q.791.

10.3.4. Signaling Connection Control Part

The network level functions provided by the MTP are relatively minimal. The **Signaling Connection Control Part** (SCCP) augments the services of MTP to provide user parts with enriched connectionless or connection-oriented services. It is specified by CCITT recommendations Q.711-Q.714. The combination of MTP and SCCP is called the **network service part**.

SCCP provides services (similar to those of the OSI network layer) to the SS7 user parts. It serves as a means of using higher layer OSI protocols to communicate with SS7. Five classes of SCCP service are identified:

0. Basic connectionless class.
1. Sequenced connectionless class.
2. Basic connection-oriented class.
3. Flow control connection-oriented class.
4. Error recovery and flow control connection-oriented class.

Class 1 denotes the MTP service described earlier. The connection-oriented service is defined in terms of the same set of service primitives as presented for the OSI network layer (see Chapter 4) and will not be further discussed here. SCCP messages appear as the SIF field in the signal unit. Figure 10.**Error! Bookmark not defined.** illustrates the general structure of an SCCP message.

Figure 10.14 SCCP message structure.

Field	Description
Routing Label	The routing label in SIF as explained earlier.
Message Type	Denotes the type of SCCP message.
Mandatory Fixed Part	Denotes the mandatory message parameters.
Mandatory Variable Part	Denotes additional parameters for variable messages.
Optional Part	Denotes optional parameters.

10.3.5. User Parts

User part messages appear as the SIF field in signal units. Also important, is the SIO field which, as mentioned earlier, guides message distribution. A subfield in SIO determines the message category as one of:

- SCCP message
- Telephone user part message
- ISDN user part message
- Data user part message
- Signaling network management message
- Signaling network testing and maintenance message

The **Telephone User Part** (TUP) defines the SS7 telephone signaling messages and functions for national and international calls. It directly uses the services of the

MTP. The structure of a TUP message is dependent on the message type and, because of its complexity, is beyond the scope of our discussion. The TUP is defines by CCITT recommendations Q.721-Q.725.

The **ISDN User Part** (ISUP) defines the SS7 signaling functions and messages for supporting ISDN applications. ISUP provides a variety of services: control of circuit-switched connections between subscribers, calling line identification, malicious call identification, call waiting, call redirection, as well as others. ISUP either uses the SCCP for end-to-end signaling transport between local exchanges or uses its own internal service for this purpose. The structure of ISUP messages is very similar to SCCP messages. The ISUP is defined by CCITT recommendations Q.761-Q.766.

10.3.6. Operations and Maintenance Applications Part

This part provides network management functions and messages for the Operation, Administration, and Maintenance (OA&M) of the network. It is specified by CCITT recommendation Q.795.

10.4. Private Telephone Networks

Just as WANs and LANs address different, but often complementary, data communication requirements, private telephone networks coexist with the public telephone network to address the specific telecommunication needs of independent organizations.

Private telephone networks come in different configurations and sizes, with varying degrees of connectivity or integration with public telephone networks and data networks. Most private networks revolve around a private exchange, which we will look at first.

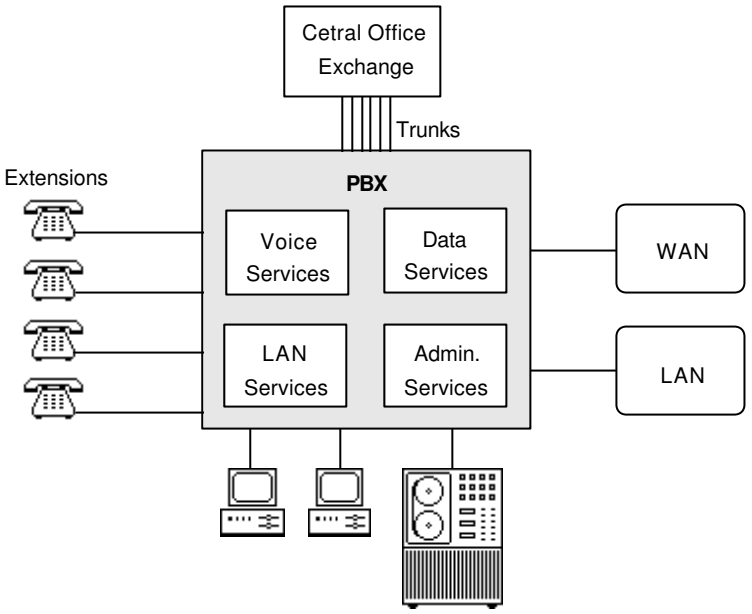
10.4.1. PBX Networks

A Private Branch Exchange (PBX; also called a Private Automatic Branch Exchange, or PABX) is a small to medium scale telephone switch typically used in office environments. One can think of a PBX as a local exchange, with the difference that it is owned and operated by a private organization other than a phone company. It is connected to a central office exchange via trunk lines and provides a number of local lines or **extensions** (ranging from tens to thousands) for connecting office telephones. All calls between the extensions are locally handled by the PBX. Outside calls are routed by the PBX as normal public network calls via the trunk lines.

Modern PBX systems are digital and can be configured to provide integrated voice and data services. Consequently, the PBX can be used in a LAN capacity to interconnect personal computers, mainframes, and other computing devices (though typical PBX data rates are much lower than LAN data rates). Furthermore, it can be

interconnected to other public and private networks, such as a public X.25 network or a corporate LAN. Figure 10.**Error! Bookmark not defined.** illustrates a sample PBX system connected to a variety of other systems.

Figure 10.15 Sample PBX configuration.

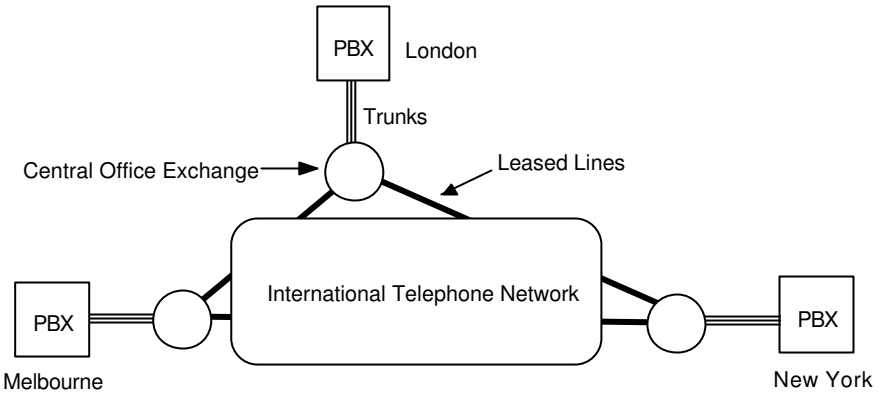


Telephone companies have also attempted to gain a share of the PBX market by offering PBX-equivalent services on their public networks. The Centrex service in North America is an example. It provides the same features as a typical PBX system but uses the public network for handling calls. Given that most PBX installations represent significant investments with continued upgrading and reconfiguration costs, the Centrex approach has obvious economic attractions.

10.4.2. Corporate Networks

Many large corporations operate their own distributed private telephone network, which consists of a number of PBX systems at different sites, interconnected through leased trunk lines. Each user is provided with a network-wide number which can be used to contact them, independent of their geographic location. The network also provides an interface to the public network for contacting people outside the corporation. By concentrating traffic on leased trunk lines, significant reductions in running costs can be achieved. Figure 10.**Error! Bookmark not defined.** illustrates the structure of a simple corporate network.

Figure 10.16 A corporate network.

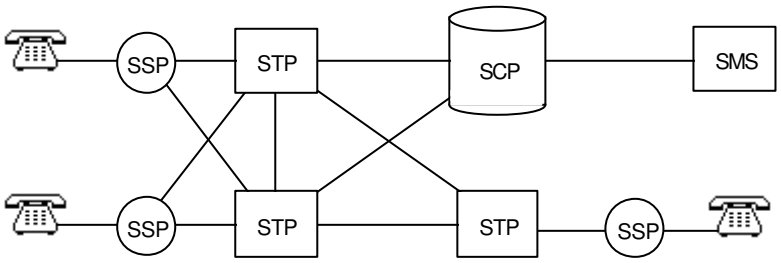


10.4.3. Intelligent Networks

SS7 opens the scope for user-defined procedures for the way calls are handled. This leads to the concept of Intelligent Networks (IN). A conceptual view of an intelligent network is shown in Figure 10.**Error! Bookmark not defined.**. The various components in the diagram are explained below..

Network access is facilitated through a set of **Service Switching Points** (SSPs). The signaling link between the switches is controlled through a set of **Signaling Transfer Points** (STPs). A **Service Control Point** (SCP) is in charge of dictating how calls should be handled and routed. For its operation, the SCP uses a service database which contains service profile definitions provided by the customer. Service profiles use a number of network-provided parameters (e.g., time and date of the call, origin of the call, network-related conditions) to determine how to handle calls. An **Support Management System** (SMS) provides network management and customer control capabilities.

Figure 10.17 An intelligent network.



IN applications include: Call Management Services (CMS), Customer Local Area Signaling Service (CLASS), Personal Communications Service (PCS), Centrex, Automatic Calling Card Service (ACCS), and basic and enhanced 800 service. The 800 (or 008) service is available in many countries. A typical 800

customer is an organization with numerous outlets throughout the country (e.g., a fastfood chain). The service is characterized by a single number with the 800 prefix which, when dialed from anywhere in the country, is redirected to the SCP for handling. A typical service profile would use the origin of the call to direct it to the nearest outlet.

10.5. Further Reading

Martin (1988), Tanenbaum (1989), Black (1989), and Stallings (1991) provide highly readable introductory descriptions of telephone networks. Detailed descriptions of telecommunication concepts protocols can be found in Schwartz (1987), Freeman (1989), Martin (1990), Bellamy (1991), Spragins *et al* (1991), Van Duuren *et al* (1992), Heldman (1993), and Heldman (1994). Modarressi and Skoog (1990) and Jabbari (1991) describes SS7 and its use for ISDN and intelligent networks. Bush and Parsosns (1992) provide a readable account of PBX systems.

11. Integrated Services Digital Network

Current trends in telecommunication are toward integration of voice and data services. So far these services have been available separately, requiring separate subscription, communication links, and equipment. It has long been acknowledged that the integration of these services will result in significant flexibility and cost benefits to both service users and service providers. The Integrated Service Digital Network (ISDN) is a major attempt to realize these objectives.

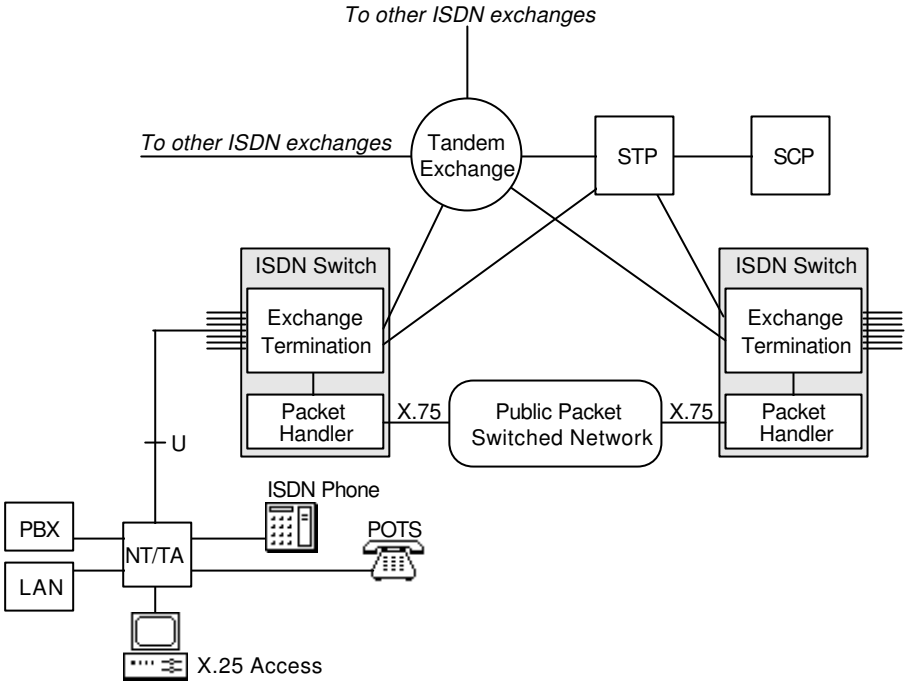
The three most important ingredients of ISDN have already been discussed in earlier chapters: circuit switching, packet switching, and common channel signaling (SS7). This chapter looks at the rest of the ISDN technology. We will start with some basic ISDN concepts, including its channels, reference points, functional groupings, and services. We will then describe the ISDN protocol architecture in relation to the OSI model, and discuss various ISDN standards. Finally, we will examine the potential future of ISDN within the context of global communication networks.

11.1. Basic Concepts

ISDN provides a fully integrated digital network for voice and data communication. It supports both circuit and packet switching. Figure 11.123 illustrates the overall arrangement of a typical ISDN network, which brings together a mix of old and new technologies and devices.

Each ISDN switch consists of an **exchange termination** part, which performs the necessary circuit switching functions, and a **packet handler**, which performs the necessary packet switching functions. The packet handlers implement X.25 and are connected to a public packet switched network via X.75. The exchange terminations are interconnected via tandem exchanges. STPs and SCPs provide network intelligence, and were described in the previous chapter. Subscriber access is provided via a network termination and/or terminal adapter (NT/TA). This provides the connectivity for a variety of user devices, including ISDN phones, Plain Old Telephone Sets (POTS), LANs, PBXs, and X.25 terminals.

Figure 11.123 ISDN network overview.



11.1.1. ISDN Channels

Subscriber access to ISDN is via digital channels, of which there are three types:

- **B channels** are used for carrying user data (digitized voice or computer-generated data) at 64 kbps. This data rate is more than necessary in many situations (e.g., compressed digitized voice can be transmitted using less bandwidth). Consequently, a B channel is sometimes subdivided into smaller subchannel. Whether there is a subdivision or not, the network treats the whole thing as one channel. All subchannels therefore are between the same two end-points and follow the same route.
- **D channels** are primarily used for common channel signaling purposes. They are typically associated with B channels and carry the control signals for B channel calls. D channels are also used for packet-switched data communication. A D channel may operate at 16 or 64 kbps.
- **H channels** are used in a high-speed trunk capacity. They are suitable for applications that require higher than 64 kbps data rates. Multi-media applications (e.g., audio, video, and graphics multiplexed over the same channel) are examples. H channels are divided into three categories depending on their speed:
    - H0 operates at 384 kbps (= 6 B channels)

    - H11 operates at 1536 kbps (= 23 B channels)
    - H12 operates at 1920 kbps (= 30 B channels)

Only D channels can be used for carrying signaling information. B and H channels can only be used for carrying user data.

In practice, channels are offered to users in a packaged form. Two such packages have been defined: basic access and primary access. The **Basic Rate Access** (BRA) package (also called 2B+D) is primarily intended for residential subscribers and consists of the following:

- Two B channels
- One 16 kbps D channel
- Overhead of 48 kbps for framing, synchronization, etc.

This produces a total bit rate of 192 kbps. The channels may be used for a variety of purposes. For example, the two B channels can be used for two independent voice services, or one of them can be used for voice and the other for a data service such as fax, teletex, or remote LAN access. Modest data communication requirements (e.g., remote banking transactions) may be met by the D channel alone. Other permitted combinations for basic access are: B+D or just D.

The **Primary Rate Access** (PRA) package is aimed at business users with greater bandwidth requirements. Primary access comes in two configurations:

- At a bit rate of 1.544 mbps (North America and Japan) and consisting of:
    - 23 B channels
    - One 64 kbps D channel
    - Overhead of 8 kbps
- At a bit rate of 2.048 mbps (Europe) and consisting of:
    - 30 B channels
    - One 64 kbps D channel
    - Overhead of 64 kbps

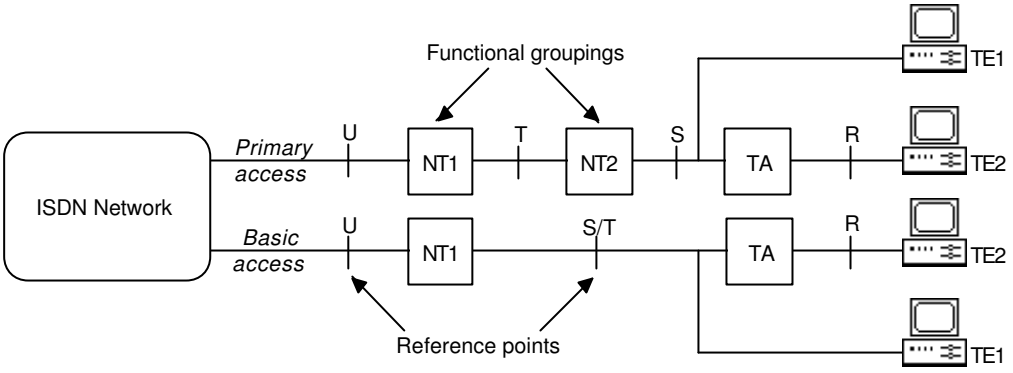
As with the basic access, lower configurations are also possible, depending on requirements. Primary access can also support H channels.

11.1.2. Functional Groupings and Reference Points

User access to ISDN is provided at a number of different levels of abstraction. These levels are defined by **functional groupings**, which encompass functions

equivalent to those denoted by one or more OSI layers. The interfaces between the functional groupings are called **reference points** (see Figure 11.132).

Figure 11.124 ISDN functional groupings and reference points.



The U (User) interface is a 2-wire physical interface to the network. The **Network Termination 1** (NT1) functional grouping provides OSI layer 1 capabilities and deals with signal transmission and physical connectors for interfacing Customer Premises Equipment (CPE) to ISDN. The NT1 transforms the U interface into a 4-wire subscriber S/T interface which supports 2B+D channels (in case of basic access) or T interface which supports 23B+D or 30B+D (in case of primary access). NT1 multiplexes these channels using TDM into a continuous bit stream for transmission over the U interface. NT1 also supports up to eight CPEs connected in a multidrop line arrangement to basic access. The NT1 device may be owned and operated by the service provider, baring the customer from direct access to the U interface, or it may be a CPE.

The **Network Termination 2** (NT2) functional grouping provides additional OSI layer 2 and 3 capabilities on top of NT1. NT2 is a CPE which transforms the T (Terminal) interface into an S (System) interface. The S interface supports 2B+D channels. NT2 may perform switching and concentration functions. A typical NT2 device would be a digital PBX, serving a set of digital phones, or a LAN, serving a set of personal computers.

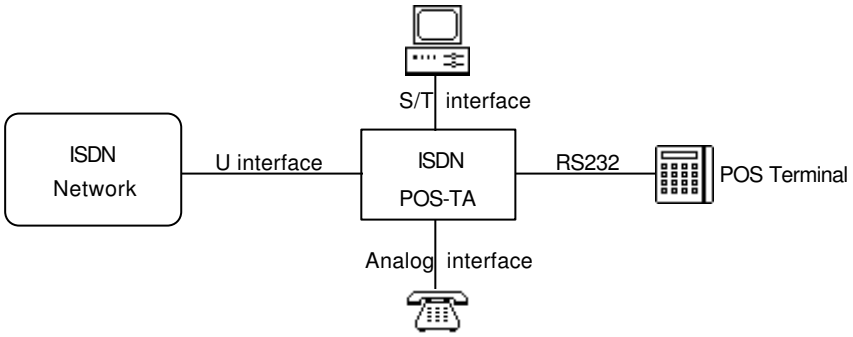
Two types of terminal equipment may be used for ISDN access. **Terminal Equipment 1** (TE1) denotes ISDN terminals which use a 4-wire physical link to the S or S/T interface. TE1 devices conform to ISDN standards and protocols and are especially designed for use with ISDN. A digital ISDN telephone and a PC with an ISDN card are examples.

Terminal Equipment 2 (TE2) denotes non-ISDN terminal equipment. Ordinary terminals and personal computers are examples. These devices can be connected to ISDN at the R (Rate) reference point. RS-232 and V.21 are examples of the type of standards that may be employed for the R reference point. The mapping between the

R interface and the S or S/T interface is performed by a **Terminal Adapter** (TA), which performs the necessary protocol conversions and data rate adaptations between the two interfaces.

It is worth pointing out that although NT1, NT2, and TAs may be offered as separate devices, in practice this is not always the case. For example, some CPE manufacturers produce TAs that have NT1 and NT2 capabilities, as well as additional interfaces for other devices (e.g., analog telephones). Figure 11.125 illustrates an example of such a device: an ISDN point-of-sale terminal adapter designed as a retail environment CPE which provides a variety of interfaces.

Figure 11.125 ISDN point-of-sale terminal adapter.



11.1.3. ISDN Services

ISDN provides three types of services:

- Bearer services
- Teleservices
- Supplementary services

Tele and supplementary services represent the type of features and functions which are visible to end-users, while bearer services represent the parts of the network which remain hidden from end-users.

Bearer services facilitate the real-time communication of digital information between end-users. These services mainly relate to network functions and account for OSI layers 1-3. An example of a bearer services is the *64 kbps, 8 kHz structured, speech* service. This service uses a data rate of 64 kbps together with 8 kHz timing information (which structures the data into octet intervals) for transmitting a Pulse Code Modulated (PCM) speech signal. The fact that the signal represents speech is known to the network, allowing it to employ transformations which may not preserve bit integrity but will result in good quality audio reproduction.

By contrast, the *64 kbps, 8 kHz Structured, Unrestricted* service makes no assumptions about the signal's data content and is a general purpose service. There are also a number of other unrestricted services, offering successively higher bandwidths (e.g., *384, 1536, and 1920 kbps, 8 kHz Structured, Unrestricted* services).

In addition to the above circuit-switched bearer services, there are a number of packet-switched bearer services. For example, the *User Signaling* service is a packet-switched bearer service suitable for exchanging signaling information between end-users and is defined by recommendation I.451, which we will examine later in this chapter. Another example is the *Connectionless on a D Channel* service which provides a datagram service on a D channel and is suitable for control-oriented or transaction-oriented applications.

Teleservices provide a set of higher-level functions on top of bearer services. These services account for OSI layers 4-7. Examples of teleservices are:

- *Telephony* services which provide speech communication over a B channel with control signaling over the D channel.
- *Facsimile* services which facilitate the communication of bitmap images over a B channel with control signaling over the D channel.
- *Teletex* services which facilitate the interchange and communication of textual as well as formatted documents over a B channel with control signaling over the D channel.

Supplementary services enhance bearer and teleservices in an independent fashion. Examples of supplementary services are:

- The *Centrex* service emulates a private network and provides specialized features to a set of subscribers.
- The *Call Transfer* service allows a user to transfer an active call to a third-party.
- The *Call Waiting* service allows a user already engaged in a call to be informed of another incoming call.
- The *Calling Line ID* service provides the calling party's address information to the called party.

Although these services all appear geared toward circuit-switched telephone calls, they are equally applicable to packet-switched data calls.

11.2. Protocol Architecture

Figure 11.126 illustrates the ISDN protocol architecture in relation to the OSI reference model. Access to the ISDN is controlled by layers 1-3. The physical layer

defines the physical U interface for basic and primary rate access. The data link layer provides two HDLC-type protocols, one for use with B channels (LAP-B) and one for use with D channels (LAP-D). The network layer provides the X.25 packet level protocol for packet switching (on either the B channel or the D channel) and a call control protocol for D channel signaling. Layers 4-7 are concerned with the end-to-end exchange of user data.

Figure 11.126 ISDN protocol architecture.

<table>
<tr><th>OSI Layer</th><th colspan="5">ISDN Layer</th><th>Purpose</th></tr>
<tr><td>higher layers</td><td>4+</td><td rowspan="3">B Channel

Circuit-

Switched</td><td colspan="3">End-to-End
User Signaling</td><td>Higher-level functions related to end-to-end signaling.</td></tr>
<tr><td>Network</td><td>3</td><td>X.25</td><td>Call
Control</td><td>X.25</td><td>Circuit-switching, packet switching, and call control signaling.</td></tr>
<tr><td>Data Link</td><td>2</td><td>LAP-B</td><td colspan="2">LAP-D</td><td>Data link control for voice/data channels and signaling channels (B and D).</td></tr>
<tr><td>Physical</td><td>1</td><td colspan="4">Physical</td><td>Physical interface for basic access and primary access (T and S interfaces).</td></tr>
</table>

Below we will look at each of layers 1-3 separately.

11.2.1. The Physical Layer

The physical user interfaces for basic and primary rate access are, respectively, defined by CCITT recommendations I.430 and I.431. The S and T interfaces both use four wires to provide full-duplex connections as two separate physical connections, one in either direction. Because of the very short distances involved between the U interface and the S and T interfaces, this is the most effective approach to full-duplex connection.

The basic access interface at the S or T reference point consists of 2B+D channels which are multiplexed onto a 192 kbps channel using TDM. As mentioned earlier, this carries an overhead of 48 kbps which accounts for framing and synchronization. Each frame is 48 bits long (see Figure 11.127) and consists of: 16 bits from either B channel (marked as B1 and B2), 4 bits from the D channel (marked as D), and 12 framing and synchronization bits (shown in *italics*). As shown in Figure 11.127, the structure of frames sent from an NT to a TE is somewhat different from the structure of frames sent from a TE to an NT. The *Framing Bit* marks the beginning of a frame. The *Balancing Bit* is a negative pulse and is intended to balance the DC voltage. These two bits are used for frame synchronization. An NT echoes the most-recently received D channel bit from a TE using the *Echo Bit* (reasons explained below). The *Activation Bit* is used by an NT to activate or deactivate a TE. The *Auxiliary Bit* and the *Auxiliary Complement Bit* (which is simply the logical negation of the *Auxiliary Bit*) are used for frame alignment. The *Multiframe Bit* is used for multiframing.

Figure 11.127 Basic rate frame structures.

NT to TE Frame	
Bit	**Description**
1	*Framing Bit*
2	*Balancing Bit*
3-10	B1
11	*Echo Bit*
12	D
13	*Activation Bit*
14	*Auxiliary Bit*
15	*Auxiliary Complement Bit*
16-23	B2
24	*Echo Bit*
25	D
26	*Multiframing bit*
27-34	B1
35	*Echo Bit*
36	D
37	*Reserved for future use*
38-45	B2
46	*Echo Bit*
47	D
48	*Balancing Bit*

TE to NT Frame	
Bit	**Description**
1	*Framing Bit*
2	*Balancing Bit*
3-10	B1
11	*Balancing Bit*
12	D
13	*Balancing Bit*
14	*Auxiliary Bit*
15	*Balancing Bit*
16-23	B2
24	*Balancing Bit*
25	D
26	*Balancing Bit*
27-34	B1
35	*Balancing Bit*
36	D
37	*Balancing Bit*
38-45	B2
46	*Balancing Bit*
47	D
48	*Balancing Bit*

The primary access interface at the T reference point consists of 23B+D or 30B+D channels which are multiplexed onto a 1.544 or 2.048 mbps using TDM. The 1.544 mbps interface is used in North America and Japan, and employs a 193-bit frame structure, while the 2.048 interface is used mainly in Europe and employs a 256-bit frame structure (see Figure 11.128). The 23B+D frames are organized as multiframes of 24 frames, with the *Framing Bit* of every fourth frame set to 0 or 1 to generate the bit pattern 001001 for the multiframe. This is used for frame alignment and synchronization. The 30+D frames, on the other hand, use the first 8 bits as a *Framing Channel*. Of these, the first bit is not used and bits 2-8 are set to 0011011 for frame alignment and synchronization.

One notable difference between the basic access and primary access interfaces is that the latter can only be used for point-to-point connections, while the former can also be used in a multidrop arrangement, where an NT serves a number of TEs via a simple bus. Such an arrangement would allow the TEs to individually use the B channels and share the D channel. Since the B channels cannot be shared, at most two TEs at any one time can have access to them. D channel frames destined for the TEs are addressed using LAP-D (explained in the next section). To transmit a frame, a TE listens to the echo of a series of 1 bits that it transmits on the D channel in absence of a signal. These bits can be echoed back by the NT using the *Echo Bit*. When the TE receives a certain number of successive echoes of its 1 bits (the exact number is dependent on the TE's predetermined priority level), it concludes that the

line is idle and it starts transmitting its frame. To avoid collisions, it continues monitoring the echo bits and compares them against the transmitted bits. It suspends transmission as soon as it detects a discrepancy between the two.

11.2.2. The Data Link Layer

Two data link layer protocols are provided. LAP-B (see Chapter 3) is used for X.25 connections over B channels. LAP-D is similar to LAP-B and is used for D channel connections. It is capable of supporting multiple network level connections. LAP-D is defined by CCITT recommendation I.441.

The LAP-D frame structure is very similar to the HDLC frame structure (see Chapter 3) and similarly supports three types of information, supervisory, and unnumbered frames via its control field. The only field where there are notable differences is the address field which consists of the following components:

- A Terminal Endpoint Identifier (TEI) which uniquely identifies a user device.
- A Service Access Point Identifier (SAPI) which uniquely identifies a network level LAP-D user entity. Four such entities have been identified: call control, packet-mode call control, X.25 packet level, and management.
- A Command/Response (C/R) bit which indicates whether the frame is a command or a response.

The LAP-D commands and responses are summarized in Figure 11.129. LAP-D provides two types of service: acknowledged and unacknowledged.

Figure 11.128 Primary rate frame structures.

23B+D Frame	
Bit	**Description**
1	*Framing Bit*
2-9	B1
10-17	B2
18-25	B3
26-33	B4
34-41	B5
42-49	B6
50-57	B7
58-65	B8
66-73	B9
74-81	B10
82-89	B11
90-97	B12
98-105	B13
106-113	B14

30B+D Frame	
Bit	**Description**
1-8	*Framing channel*
9-16	B1
17-24	B2
25-32	B3
33-40	B4
41-48	B5
49-56	B6
57-64	B7
65-72	B8
73-80	B9
81-88	B10
89-96	B11
97-104	B12
105-112	B13
113-120	B14

114-121	B15
122-129	B16
130-137	B17
138-145	B18
146-153	B19
154-161	B20
162-169	B21
170-177	B22
178-185	B23
186-193	D

121-128	B15
129-136	D
137-144	B16
145-152	B17
153-160	B18
161-168	B19
169-176	B20
177-184	B21
185-192	B22
193-200	B23
201-208	B24
209-216	B25
217-224	B26
225-232	B27
233-240	B28
241-248	B29
249-256	B30

Figure 11.129 LAP-D command and response frames.

Frame Type	Command	Response	Description
Information	I	I	Used for exchange of user data.
Supervisory	RR	RR	Receive Ready. Station is ready to receive frames and acknowledged receipt of earlier frames.
	RNR	RNR	Receive Not Ready. Station is unable to receive frames, but acknowledged receipt of earlier frames.
	REJ	REJ	Reject. Rejects a frame according to the Go-Back-N scheme.
Unnumbered	SABM		Set Asynchronous Balanced Mode.
	SABME		Set Asynchronous Balanced Mode Extended.
	UI		Unnumbered Information. Used for unacknowledged mode of information transfer.
		UA	Unnumbered Acknowledgment. Unsequenced acknowledgment frame.
	DISC		Disconnect. Forces a slave station into disconnect mode.
		DM	Disconnect Mode. Used by a slave to indicate it is in disconnect mode.
	XID	XID	Exchange identification information.
		FRMR	Frame Reject of a protocol-violating frame.

The **acknowledged service** involves the exchanging of all three types of frames between the network and a TE. First, a link is requested using the SABM or SABME command, which may be accepted by a UA response or rejected by a DM response. Then, the two ends may exchange information frames using the I command. The supervisory frames are used during this phase for error control (using the Go-Back-N scheme) and flow control (using the sliding window protocol). Finally, the connection may be terminated by a DISC command. FRMR is used to reject a frame which violates the protocol (e.g., incorrect frame length or invalid sequence numbers). This results in the connection being aborted. The XID command is used for exchanging identification information (e.g., negotiating a new set of values for LAP-D parameters).

The **unacknowledged service** provides basic exchange of user information using the UI command, which are received unacknowledged. No error control or flow control is provided, although CRC-based error detection is still performed and incorrect frames are simply discarded.

Though LAP-D is based on LAP-B, some important differences exist. Firstly, unlike LAP-B which is designed for point-to-point connection, LAP-D can support multiple DEs connected via a multidrop line. Secondly, LAP-D supports UI frames, while LAP-B only supports sequenced information frames. Lastly, LAP-D uses a different addressing scheme.

11.2.3. The Network Layer

The ISDN network layer is defined by CCITT recommendations I.450 and I.451. It supports circuit and well as packet switched connections between a TE and the network, and end-to-end.

Figure 11.130 summarizes the ISDN network layer messages exchanged between TEs and the network over the D channel. Figure 11.131 illustrates the use of these messages in a typical scenario which involves various D channel signals for handling a B channel circuit-switched call between two subscribers with ISDN telephones.

The scenario proceeds as follows. The calling subscriber picks up the receiver and dials the called subscriber's number. The calling subscriber's telephone accumulates the digits (address) and uses them as a parameter in a *Setup* message which it sends to the network. This message is conveyed by the network to the called subscriber's telephone using SS7 signaling. At the same time, the network sends a *Setup Acknowledge* message to the calling telephone and asks for further information, which the latter provides using an *Information* message, and the network confirms, using a *Call Proceeding* message, that the call is being established.

Figure 11.130 ISDN network layer messages.

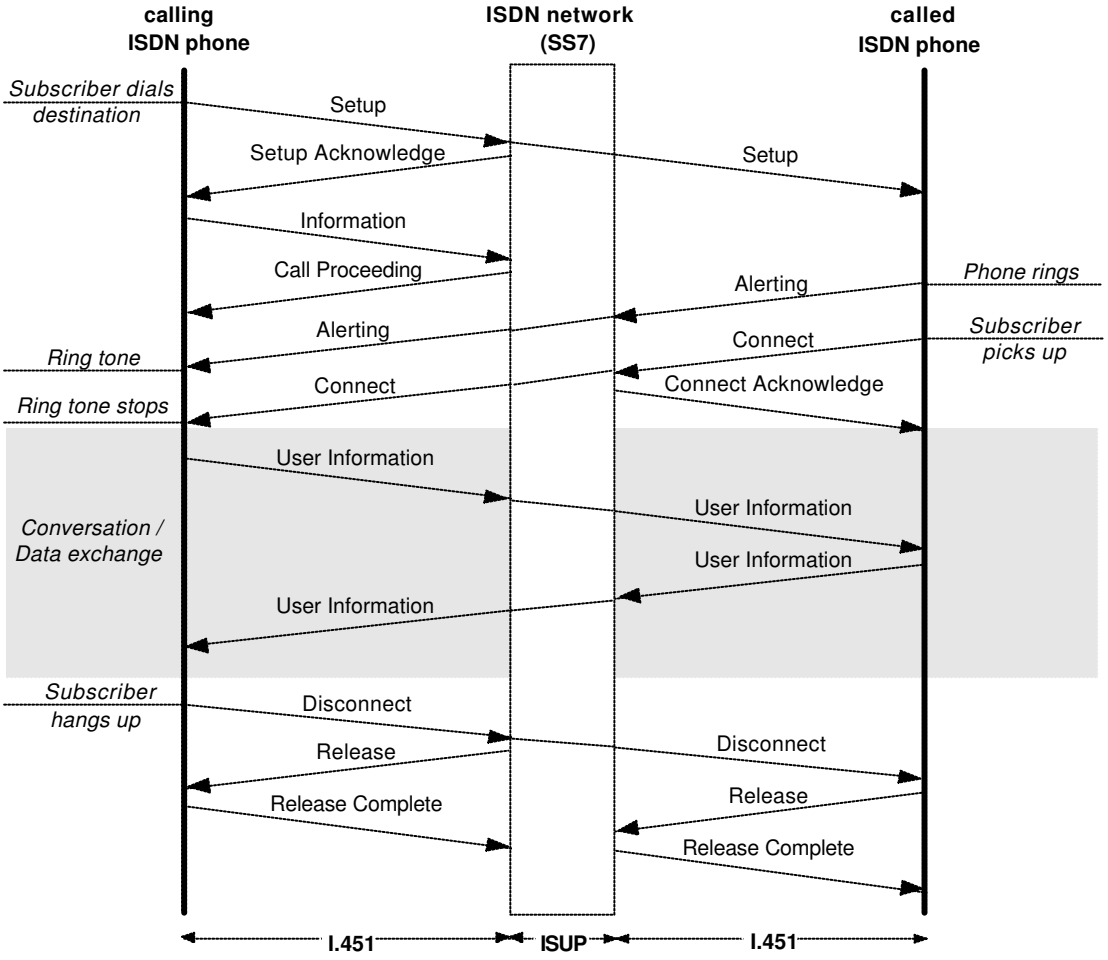
Message	Purpose
Setup	Used by calling TE to convey to the called TE a request for call establishment.
Setup Acknowledge	Used by the network to acknowledge the connect request from the calling TE.
Alerting	Used by called TE to convey to calling TE that it has started alerting the user.
Call Proceeding	Used by the network to inform the calling TE that the call is being established.
Connect	Used by called TE to convey to calling TE that it has accepted the call.
Connect Acknowledge	Used by network to acknowledge the receiving of the connect message from the called TE.
Detach	Used for releasing the B channel without disposing of the call reference information.
Detach Acknowledge	Acknowledges the releasing of the B channel in response to a Detach message.
Disconnect	Used to indicate the completion of the call so that the channel and call reference may be released.
Release	Used to indicate that the channel and call reference have been released.
Release Complete	Used to indicate the acceptance of a Release request and therefore the releasing of the channel and call reference.
Suspend	Used by a TE to request from the network the suspension of an existing call.

Suspend Acknowledge	Used by the network to acknowledge that the call has been suspended.
Suspend Reject	Used by the network to indicate that the call suspend request has been rejected.
Resume	Used by a TE to request the resuming of a previously suspended call.
Resume Acknowledge	Used by the network to acknowledge that the suspended call has been resumed.
Resume Reject	Used by the network to indicate that the call resume request has been rejected.
User Information	Used by a TE to exchange user information with another TE.
Register	Used by a TE to register itself for the use of a long-term network facility.
Register Acknowledge	Used by the network to acknowledge that the TE has been registered with the requested long-term facility.
Register Reject	Used by the network to reject a long-term facility registration requested by a TE.
Facility	Used by a TE to request the activation of a network facility.
Facility Acknowledge	Used by the network to acknowledge that the requested facility has been activated.
Facility Reject	Used by the network to reject a facility activation requested by a TE.
Cancel	Used by a TE to request the cancellation of a network facility.
Cancel Acknowledge	Used by the network to acknowledge that the requested facility has been canceled.
Cancel Reject	Used by the network to reject a facility cancellation requested by a TE.
Information	Used for providing additional information.
Status	Used during a call to report call-related conditions, including unexpected messages.
Congestion Control	Used for the flow control of messages exchanged between TEs.

When the called phone starts ringing, it conveys an *Alerting* message via the network to the calling phone, which in response produces a ring tone to indicate that the destination phone is ringing. When the called subscriber picks up the receiver, its phone conveys a *Connect* message via the network to the calling phone. As a result, the ring tone stops and a B channel circuit is established for exchanging digitized speech. During this time, other information may also be conveyed over the D channel (e.g., using *User Information* messages).

The call is terminated when one of the two subscribers hangs up its phone. This causes a *Disconnect* message being conveyed to the other phone via the network. The network responds with a *Release* message, which the phone confirms with a *Release Confirm*. On the other side, the phone responds with a *Release* message, which the network confirms with a *Release Confirm*. Hence the call is completed and the B and the D channel are released.

Figure 11.131 Sample circuit-switched call scenario.



There are other messages which are not covered by our scenario. Calls can be temporarily suspended and later resumed using the *Suspend* and *Resume* set of messages. The *Register*, *Facility*, and *Cancel* set of messages are used to manage facilities. ISDN supports a variety of facilities, such as: reverse charging, call completion after busy, provision of charging information, X.25 flow control negotiation, X.25 fast select.

As indicated in Figure 11.131, I.451 handles the signaling between TE and the local exchange, while the ISUP (ISDN User Part was explained in Chapter 10) handles the common channel signaling in between exchanges within the network. (For clarity, the ISUP signals are not shown here.) I.451 and ISUP interwork to provide end-to-end signaling to the users.

Figure 11.132 ISDN network layer message structure.

		Field	Description
		Protocol Discriminator	Denotes the message protocol.
		Call Reference Length	Length of the reference flag + value.
		Call Reference Flag	Denotes the initiating D channel end.
		Call Reference Value	Uniquely identifies the B channel call.
		Message Type	Denotes the D channel message type.
E	sample	1	Denotes a fixed-length element.
L	fixed-length	Element ID	Uniquely identifies the element.
E	element	Element Contents	Element information contents.
M		*zero or more elements*	*A total of zero or more elements are permitted.*
E	sample	0	Denotes a variable-length element.
N	variable-length	Element ID	Uniquely identifies the element.
T	element	Contents Length	Length of the element contents.
S		Element Contents	Element information contents.

Figure 11.132 shows the general structure of the I.451 network layer messages just described. Each message consists of four parts. The *Protocol Discriminator* identifies the network level protocol to which the message belongs. This will either denote I.451 or X.25. The *Call Reference* denotes the B channel call to which this message applies. It consists of a *Length* field, a *Flag*, and a *Value*. The *Flag* denotes the call-initiating end of the D channel. The *Message Type* denotes one of the messages listed in Figure 11.130. The *Elements* (of which there may be zero or more, depending on the *Message Type*) are used for representing the message parameters. Elements may be fixed or variable length. A fixed length element consists of a 1 bit, followed by an *Element ID*, and *Element Contents* (total of one octet). A variable length element consists of a 0 bit followed by an *Element ID*, a *Content Length*, and *Element Contents* (total of at least three octets).

11.3. Frame Relay

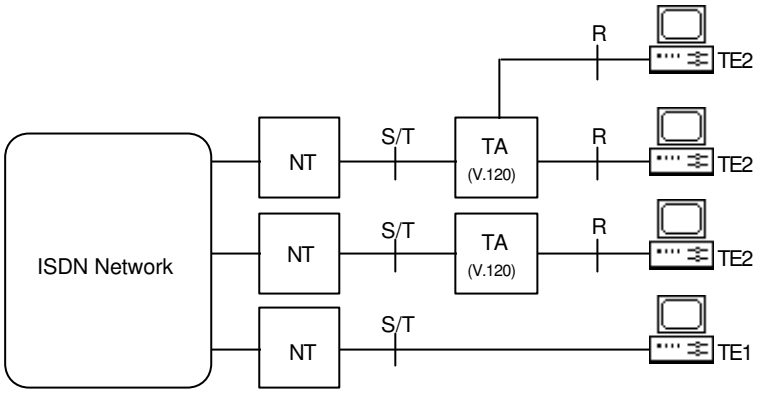
Frame relay is a major ISDN data link protocol for end-to-end connections over the B channel. Whereas LAP-B supports packet-switched connections, frame relay is designed to support circuit-switched connections. Frame relay is an evolution of an earlier protocol, V.120, which supports multiple logical connections over the same B channel cicuit. Frame relay provides similar capabilities but removes the restriction that all the multiplexed logical connections over a B channel should be between the same two end-users.

We will look at V.120 first and then discuss frame relay.

11.3.1. V.120

V.120 is a popular protocol implemented by ISDN terminal adapters. It enables a TE2 to communicate with a TE1 (or another TE2 connected to a TA) over a B channel circuit (see Figure 11.133).

Figure 11.133 V.120-based connections.



V.120 supports asynchronous as well as synchronous transmissions. Synchronous transmission may be either as raw data or HDLC-based. Multiple logical channels between two end-users can be multiplexed onto the same B channel. Rate adaptation is provided for devices operating at less than 64 kbps. A sliding window protocol is used for flow control.

Except for the address field and the data field, the V.120 frame structure is very similar to LAP-D and HDLC. The address field contains a Logical Link Identifier (LLI) for the identification of multiple logical connections over the same B channel circuit. Certain LLI values are assigned to specific purposes. For example, 0 denotes in-channel signaling and 256 means no multiplexing.

As well as carrying user data, the data field may conatin one or two header octets. The first octet is called the terminal adaptation header and consists of a set of bit flags for controling a number of adaptation functions, including: segmentation of HDLC frames, error control, HDLC idle condition, and extension of the header with additional control information. The second octet, the control state information octet, is optional and also consists of a set of bit flags.

The V.120 procedure involves the following steps. First, a B channel circuit is established using the Q.931 call control protocol over the D channel. The resulting circuit may be used in connectionless mode by using an LLI value of 256 for the frames. For a connection-oriented mode, however, additional steps are necessary to manage multiplexed logical links. Four messages are provided for this purpose:

- **Setup** may be issued by either side to request a logical link to be established.

- **Connect** may be issued in response to a setup message in order to accept the connection.
- **Release** may be issued by either side to release an existing connection.
- **Release Complete** may be issued in response to a setup message to deciline a connection, or in response to a release message to confirm the release.

These messages may be exchanged either over the D channel (using Q.931 messages) or over the B channel (using V.120 frames with LLI set to 0).

When the B channel circuit is no longer needed, it may be terminated by either side using the Q.931 call control protocol on the D channel.

11.3.2. Frame Relay

Frame relay is an evolution of V.120 and provides similar capabilities to X.25, but in a considerably streamlined form, thus facilitating higher data rates. Simplifications employed by frame relay include: use of two protocol layers (as opposed to three in X.25), end-to-end (as opposed to node-to-node) error control and flow control, and use of a separate logical connection for control signaling. The main argument in support of frame relay is the fact that ISDN utilizes highly reliable transmission media (e.g., optical fiber) which do not justify the high overheads of X.25. Like X.25 and V.120, frame relay supports the multiplexing of multiple logical connection over the same B or H channel. Unlike V.120, it does not requires these connections to be between the same two end-users.

As with ISDN, frame relay uses I.430 or I.431 for its physical layer. The data link layer is sublayered. The core functions of Q.922 (an enhanced version of LAP-D) are used for end-to-end link control. These are essentially the same as LAP-D, except that they also include congestion control functions. At the UNI, the whole of Q.922 (including error control and flow control functions) is used for the reliable transfer of Q.931 messages.

End-user access to the network is always via a **frame handler**. The frame handler may be an integrated component of the local exchange or a completely separate node in the network. A B or H channel end-user connection to a frame handler is established by exchanging Q.931 messages over the D channel. Multiple logical frame relay connections may then be established over this connection.

Frame relay uses a frame structure very similar to that of V.120, except that there is no control field. Consequently, there are no control frames, which means that functions such as inband signaling, flow control, and error control are not supported. Error handling is limited to checking the FCS of a frame and discarding it if it is incorrect.

Given that the role of frames in a frame relay network is similar to packets in a pcaket-switched network, some means of congestion control is needed to ensure

adequate performance. Because a frame relay network has no means of flow control, it cannot assume full responsibility for congestion control. Instead, this is jointly handled by the user and the network. Two forms of signaling are provided to facilitate this: explicit signaling and implicit signaling.

Explicit signaling is based on two bits in the frame address field. These bits may be set by the frame handlers in the network in order to communicate congestion information to the user. The Backward Explicit Congestion Notification (BECN) bit serves as a warning to the user that frames transmitted by the user in the opposite direction to this frame may experience congestion situations. The user can respond to this by simply reducing the frame transmission rate. The Forward Explicit Congestion Notification (FECN) bit warns the user of congestion in the same direction as the frame itself. To respond to this, the user can submit a request (via Q.922) to its peer user to reduce its frame transmission rate.

Implicit signaling involves the user detecting that the network has discarded a frame due to congestion. In response, the user can reduce its transmission rate. An additional bit in the address field of a frame, called Discard Eligibility (DE), can be set by the user to indicate to the network that when discarding frames due to congestion, preference should be given to this frame.

11.4. Internetworking

For economic and practical reasons, the introduction of ISDN involves a rather long transitory phase, during which time ISDN and earlier networks have to coexist. For this to work, some means of internetworking is needed. For examples, subscribers who are served by ISDN may need to contact subscribers who are served by PSTN. Such internetworking involves a number of considerations, including the following:

- Mapping between the numbering plans of the two networks.
- Mapping between the services offered by the two networks.
- Mapping between the control signals employed by the two network.
- The physical interfacing of the two networks.

An additional set of reference points have been defined by CCITT to address the above. These reference points define interfaces between ISDN and other networks:

- The K reference point is used for interfacing ISDN to an existing, non-ISDN voice or data network, where the internetworking mappings are performed by ISDN.

- The L reference point is used for interfacing ISDN to an existing, non-ISDN voice or data network, where the internetworking mappings are performed by the non-ISDN network.
- The M reference point is used for interfacing ISDN to a specialized network (e.g., MHS), where the internetworking mappings are performed by the latter.
- The N reference point is used for interfacing two ISDN networks, where both networks may need to do internetworking mappings to provide a common set of services.

11.5. ISDN Standards

Figure 11.134 provides a high-level view of CCITT ISDN recommendations. Each of the series consists of many parts. The I.100 series provides a general introduction to ISDN, its principles, objectives, and envisaged evolution path. The I.200 series defines the ISDN end-user services and a set of attributes for each service. The I.300 series defines the ISDN network and its functions and protocols in relation to ISDN services. The I.400 series defines the physical and logical interfaces between ISDN and its users. The I.500 series deals with the issues of internetworking between ISDN networks and between ISDN and non-ISDN networks (e.g., PSTN). The I.600 series deals with the issues of maintenance of subscriber access and installation.

Figure 11.134 High-level view of CCITT ISDN recommendations.

Part / Series	Description
I: I.100 Series	General description of ISDN concepts
II: I.200 Series	Service capabilities
III: I.300 Series	Overall network aspects and functions
IV: I.400 Series	User-network interfaces
V: I.500 Series	Internetwork interfaces
VI: I.600 Series	Maintenance principles

Figure 11.135 summarizes the major ISDN-related standards for four areas of basic rate, primary rate, SS7, and packet-switched network.

Figure 11.135 Major ISDN-related standards.

	Layer	Standard	Description
Basic Rate	1	I.430	2B+D channels (D=16 kbps)
	2	I.441	LAP-D (with multiple TEs)
	3	I.451	User-network interface
Primary Rate	1	I.431	23B+D or 30B+D channels (D=64 kbps)
	2	I.441	LAP-D (point-to-point)
	3	I.451	User-network interface

Signaling System 7	1-3	Q.701-Q.708	Message Transfer Part (MTP)
	4	Q.711-Q.714	Signaling Connection Control Part (SCCP)
	4+	Q.761-Q.766	ISDN User Part (IUP)
	4+	Q.791-Q.795	Operation, Admin and Maintenance (OA&M)
Packet Network	1	X.21	Physical interface
	2	LAP-B	Link access
	3	X.25, X.75	Packet interface

11.6. Further Reading

Black (1989) and Freeman (1989) both contain introductory chapters on ISDN. Griffiths (1990), Helgert (1991), Stallings (1992), and Kessler (1993) provid detailed discussions of ISDN architecture, services, and protocols. Dickson and LLoyd (1992) discuss many of ISDN protocols within the context of the OSI model. Deniz (1993) describes the use of ISDN for interconnecting LANs. Lai (1989) provides a useful introduction to frame relay. Deatiled descriptions of frame relay applications and protocols can be found in Chen and Rege (1989), Smith, P. (1993), and Black (1994).

12. Broadband ISDN and ATM

This chapter describes Broadband ISDN (B-ISDN) and the transfer mode designed for its implementation: Asynchronous Transfer Mode (ATM). ATM provides a means for fast switching and transmission at data rates required by B-ISDN services. It relies on the division of information into small fixed-size packets, called cells, and their *demand*-based transmission over optical fiber – hence the term *asynchronous*.

We will first discribe some of the basic concepts underlying ATM, including its protocol architecture, and then discuss each of its layers in turn. We will also discuss the organization of ATM networks and their potential applications.

12.1. Broadband ISDN

Increasing market demand for data rates substantially greater than those supported by ISDN has lead to the notion of **Broadband ISDN** (B-ISDN). By relying on optical fiber transmission systems, B-ISDN opens the door to a whole range of new applications (e.g., high quality digital audio, real-time video, pay TV, video phone, high speed LAN connection) with emphasis on interactivity and high speed transfer services.

B-ISDN is developed as an evolution of ISDN and hence follows the same principles. This section provides an overview of B-ISDN, its services, and its protocol architecture.

12.1.1. B-ISDN Services

B-ISDN services are clasified into **interactive** and **distribution** services. Interactive services involve the bidirectional flow of user information between two subscribers or between a subscriber and a service provider. Interactive services are divided into three subcategories:

- **Conversational** services involve real-time exchange of information such as sound, video, data, or entire documents. Typical examples include: video-telephony, video-conference, and high speed data transfer. Video-telephony is like the normal voice telephony service but also includes video capture, transmission, and display capabilities. Video-conference provides voice and video communication between two conference rooms or between a number of individuals.

- **Messaging** services involve non-real-time exchange of information between subscribers in a store-and-forward fashion. Examples include: video mail and document mail. Video mail supports the exchange and storage of moving video images and sound as messages. Document mail allows documents of different types and formats to be exhanged and stored.
- **Retrieval** services provide subscribers with retrieval access to centrally-stored public information. Examples include: broadband videotex (retrieval of video images/sequences with sound, text, and graphics), video retrieval (subscriber access to video libraries of movies), and retrieval of high-resolution images and documents from various archives and information centres.

Distribution services involve the unidirectional flow of user information from a service provider to a subcriber. Distribution services are divided into two subcategories:

- Distribution services **without user presentation control** involve the central broadcast of information to a large number of subscribers, where subscribers have no control over the presentation of information. Examples include: broadcast of TV programmes, electronic newspapers, and electronic publishing.
- Distribution services **with user presentation control** are the same as the previous category except that here the information is offered as cyclically-repeated frames, thereby enabling the subscriber to control the start and the order of the presentation of frames. Examples include: electronic newspaper and tele-advertising.

Figure 12.136 summarizes the B-ISDN services.

Figure 12.136 Summary of B-ISDN services.

Category	Subcategory	Type of Information	Example
Interactive	Conversational	Moving picture and sound	Broadband video-telephony
			Broadband video-conference
			Video-surveillance
		Sound	Multiple sound prog. signal
		Data	High speed data transfer
			High volume file transfer
			High speed teleaction
		Document	High speed telefax
			High resolution image comm.
			Document communication
	Messaging	Moving picture+sound	Video mail
		Document	Document mail
	Retrieval	Text, data, graphics, still images, moving pictures	Broadband videotex
			Video/Image retrieval
			Document/Data retrieval
Distribution	Without user	Data	Data distribution

	presentation control	Text, graphics, images	Electronic newspaper
		moving pictures+sound	Video signal distribution
		Video	TV programme distribution
	With user presentation control	Text, graphics, sound, images	Electronic newspaper Tele-advertising

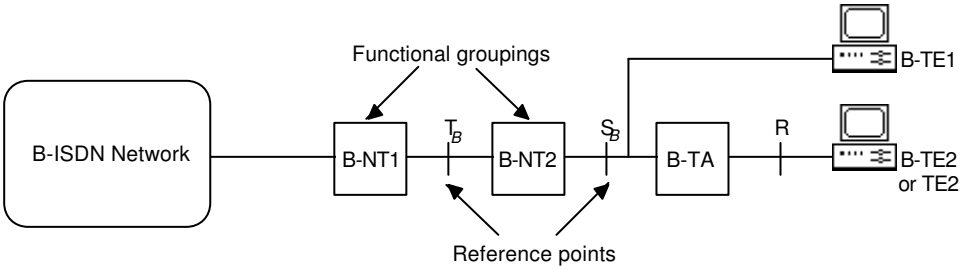
12.1.2. B-ISDN User-Network Interface

In addition to the narrowband channels defined for ISDN, B-ISDN supports the following three User-Network Interfaces (UNIs):

- **Symmetric full-duplex 155.52 Mbps**. This interface provides the basis for interactive services. This will therefore be the most common interface offered. It is commonly referred to as the 150 Mbps interface. The effective payload of this interface is 149.76 Mbps.
- **Symmetric full-duplex 622.08 Mbps**. This interface is suitable for situations which involve very high traffic volumes. It is commonly referred to as the 600 Mbps interface. The effective payload of this interface is 599.04 Mbps or 600.768 Mbps, depending on how it is structured.
- **Asymmetric full-duplex 622.08/155.52 Mbps**. This is a hybrid interface which combines the earlier two. It uses bit rates of 622.08 in the network-to-subscriber direction and 155.52 Mbps in the subscriber-to-network direction. It is suitable for situations where the traffic volume in the network-to-subscriber direction is much higher than in the opposite direction.

The functional groupings and reference points for B-ISDN UNI closely follow those of the narrowband, as illustrated by Figure 12.137. Their role remains identical to those of narrowband ISDN. To highlight the broadband nature of the functional groupings and reference points, the letter 'B' is added (e.g., B-TE1). The R reference point is an exception because it may or may not have broadband capabilities. Accordingly, it may support a B-TE2 or TE2.

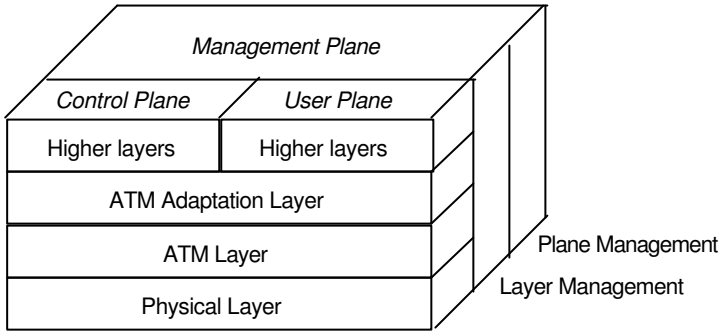
Figure 12.137 B-ISDN functional groupings and reference points.



12.1.3. B-ISDN Protocol Architecture

Figure 12.138 shows the reference model for the B-ISDN protocol architecture. The architecture is partitioned by three planes. The **User Plane** facilitates the transfer of user information and associated control signals. The **Control Plane** is responsible for connection control functions such as call setup and release. The **Management Plane** is responsible for management functions relating to the system as a whole and coordination between the planes (Plane Management), and for management functions related to resources and parameters residing in its protocol entities (Layer Management).

Figure 12.138 B-ISDN protocol architecture reference model.



The **Physical Layer** is responsible for the physical transmission of ATM cells. To achieve some degree of physical medium independency, this layer is divided into two sublayers. The **Physical Medium Sublayer** covers the medium-dependent aspects of the physical layer and provides bit transmission and bit timing functions. The **Transmission Convergence Sublayer** is independent of the physical medium and provides five functions:

- Generation and recovery of transmission frames (similar to frame structures described in Chapter 11 for narrowband ISDN).
- Transmission frame adaptation which involves the packing of cells into transmission frames and their extraction upon arrival.
- Cell delineation to enable the receiving-end identify the boundaries of cells whose information fields have been scrambled prior to transmission and are descrambled by the receiver.
- HEC sequence generation by the transmitter, and verification by the receiver.
- Cell rate decoupling through insertion of idle cells by the transmitter in order to adapt the cell rate to the playload capacity, and their removal by the receiver.

The **ATM Layer** is responsible for the transfer of cells across the network and is independent of the physical medium. It provides four functions:

- Cell multiplexing by the transmitter in order to support multiple logical channels, and their demultiplexing by the receiver.
- Virtual path and virtual channel identifier translation (explained in the next section).
- Cell header generation by the transmitter, and its extraction by the receiver.
- Generic flow control for the control of traffic in a customer network by placing flow control information in cell headers.

The **ATM Adaptation Layer** (AAL) packages information from higher layers into ATM cells to be transported by the ATM layer at the transmitter end, and extracts information from ATM cells for passing to higher layers at the receiver end. It consists of two sublayers. The **Segmentation and Reassembly Sublayer** handles the segmentation of information from higher layers into smaller units so that they can be transported by ATM cells, and their subsequent reassembly upon arrival. The **Convergence Sublayer** specifies the services that AAL provides to higher layers and is service-dependent.

The physical and the ATM layer provide the same functions for the user and the control plane. The ATM adaptation and higher layers, however, may provide different functions for these two planes.

12.2. Asynchronous Transfer Mode

The recommended switching technology for B-ISDN is the Asynchronous Transfer Mode (ATM). Given the high reliability of optical fiber for transmission of digital information, the significant error control overheads involved in earlier protocols (such as X.25) become very questionable. Like frame relay, ATM takes advantage of this increased reliability to improve network performance by providing a highly streamlined protocol stack.

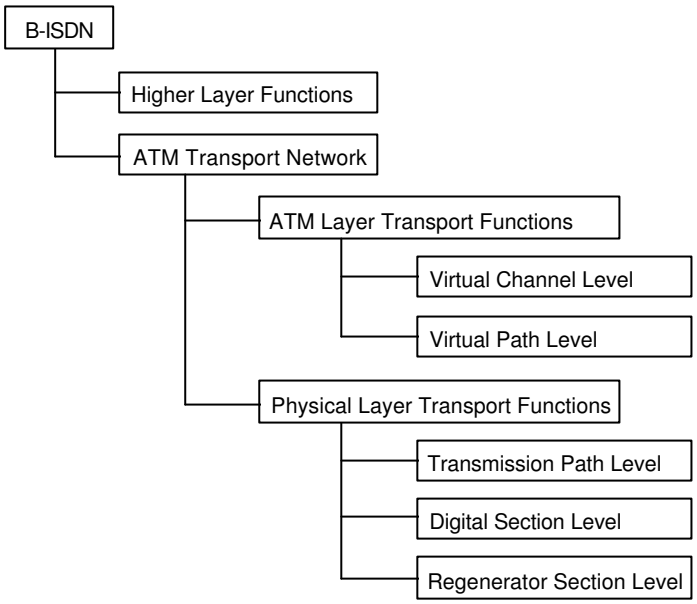
ATM uses two methods to achieve this. First, it transmit all its information in fixed-size small packets, called cells, hence simplifying the processes of packaging and unpackaging user information. Second, unlike X.25 which requires error control and flow control functions to be performed in a link-by-link fashion, it only requires end-to-end support of these functions.

12.2.1. Channels and Paths

Figure 12.139 illustrates the layered structure of B-ISDN. The **higher layer functions** are application-dependent and beyond the scope of this chapter. The

ATM transport network is divided into two layers, both of which are hierarchically organized.

Figure 12.139 Layered structure of B-ISDN.



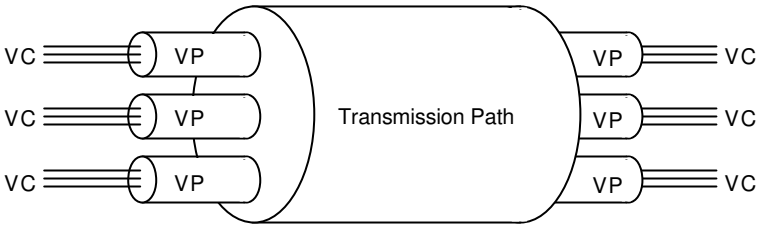
The **ATM layer transport functions** are divided into virtual channel level and virtual path level. A **Virtual Channel** (VC) denotes the transport of ATM cells which have the same unique identifier, called the Virtual Channel Identifier (VCI). This identifier is encoded in the cell header. A virtual channel represents the basic means of communication between two end-points, and is analogous to an X.25 virtual circuit.

A **Virtual Path** (VP) denotes the transport of ATM cells belonging to virtual channels which share a common identifier, called the Virtual Path Identifier (VPI), which is also encoded in the cell header. A virtual path, in other words, is a grouping of virtual channels which connect the same end-points. This two layer approach results in improved network performance. Once a virtual path is setup, the addition/removal of virtual channels is straightforward.

The **physical layer transport functions** are divided into three levels of functionality. The **transmission path** connects network elements that assemble and disassemble the transmission system payload. This payload may contain user or signalling information to which transmission overheads are added. The **digital section** connects network elements (such as switches) that assemble and disassemble continuous bit/byte streams. The **regenarator section** is simply a portion of a digital section which connects two adjacent repeaters along a transmission path which is otherwise too long to sustain the signal.

Figure 12.140 illustrates the relationship between virtual channel, virtual path, and transmission path.

Figure 12.140 Virtual channel, virtual path, and transmission path.



Two types of switching are possible: VP switching and VC switching. A VP switch connects a set of incoming VP terminations to a set of outgoing VP terminations. The mapped VPs will have different VPIs, but the VCI will remain unchanged. A VC switch, on the other hand, connects a set of incoming VC terminations to a set of outgoing VC termination. VCIs are not preserved. Because VCs are embedded in VPs, VC switching also involves VP switching.

12.2.2. ATM Cells

An ATM cell consists of 53 consecutive octets, of which 48 octents are used to represent user information and the remaining 5 octets form a header for the private use of the ATM protocols (see Figure 12.141).

Figure 12.141 ATM cell structure.

	Field	Description
Header (5 octets)	GFC	Generic Flow Control field for end-to-end flow control
	VPI	Virtual Path Identifier for routing
	VCI	Virtual Channel Identifier for routing
	PT	Payload Type for the information field
	CLP	Cell Loss Priority for discarding of cells in congestions
	HEC	Header Error Control code
Body (48 octets)	Information	User information or inband signaling information

The **GFC** field is intended for the user-network interface only. (It does not appear in the header format of cells used for the internal operation of the network. Its 4 bits are instead used by the VPI.) It can be used by applications to provide end-to-end flow control. The **VPI** and **VCI** fields collectively support the routing of a cell by the network. The **PT** field denotes the type of information stored in the information field.

The **CLP** field prioritizes the order in which cells may be discarded. The **HEC** field is an 8-bit CRC calculated over the remaining 32 bits in the header. It is used for error detection and single-bit error correction.

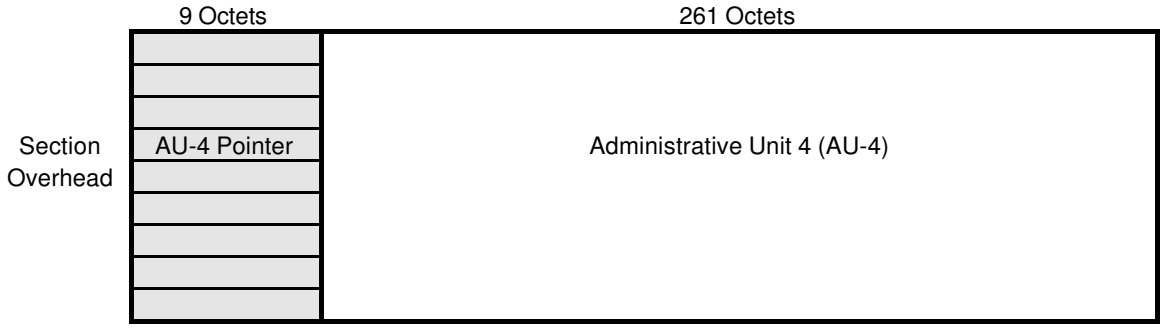
12.3. Physical Layer

This section describes the transmission convergence functions of the physical layer. The physical layer may be either cell-based or SDH-based (SDH stands for Synchronous Digital Hierarchy and will be explained in Section 12.7). The generation, recovery, and adaptation of transmission frames are done differently for these two interfaces, and are therefore described separately below. The remaining functions are identical in both cases.

12.3.1. SDH-Based Interface

Figure 12.142 shows the transmission frame format of the 155.52 Mbps SDH-based interface. It consists of a table of 9 rows by 270 columns of octets (total of 2430 octets). The frame is transmitted row-by-row at a frequency of 8 kHz, thus yielding a data rate of 2430 × 8kHz × 8 bits = 155.52 Mbps. The technical term for the frame format is Synchronous Transport Module 1 (STM-1).

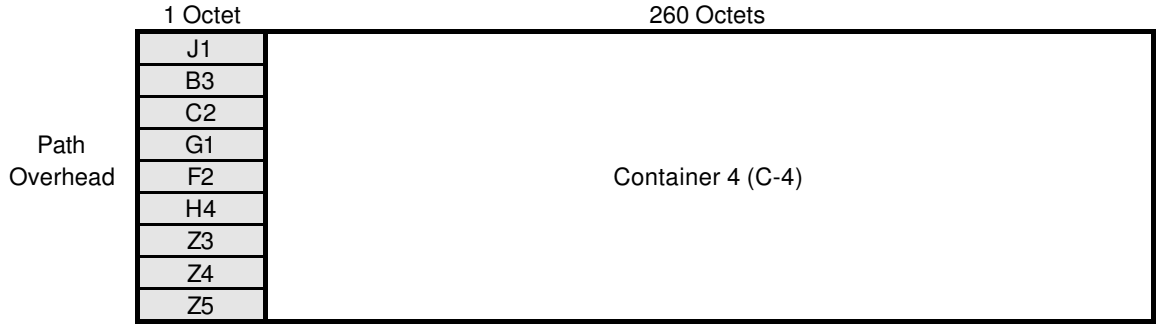
Figure 12.142 Frame format for 155.52 Mbps SDH-based interface (STM-1).



The first 9 columns of octets in the frame consist of transmission overheads called the Section Overhead (SOH). This includes framing bytes, parity bytes, additional narrowband channels for other uses, and the AU-4 pointer which points to a Virtual Container 4 (VC-4) block. The VC-4 block starts somewhere in the AU-4 which comprises the remaining 261 columns of the frame.

The format of a VC-4 block is shown in figure 12.143. It consists of 9 rows by 261 columns of octets. Of these, the first column is called the Path Overhead (POH) and the remaining 260 columns consist of ATM cells. Because the total size of the latter is not an integral multiple of 53 octets, ATM cells may cross VC-4 boundaries.

Figure 12.143 VC-4 block format for 155.52 Mbps SDH-based interface.



The POH octets are as follows. J1 provides a narrowband channel which repeatedly transmits a fixed-length, user-specified string so that path integrity can be verified by the receiver. B3 provides path-level parity. C2 specifies the path signal label. G1 contains status information. F2 provides a narrowband channel for the path user. H4 is the cell offset indicator It may assume a value in the range 0-52 which acts as an offset from the H4 octet to the first cell boundary in the same row. Z3, Z4, and Z5 are reserved for future use.

The effective payload capacity of STM-1 is 260 × 9 × 8kHz × 8 bits = 149.76 Mbps. Physical layer Operation, Administration, and Maintenance (OAM) functions are allocated to the frame overheads (SOH and POH).

The frame format for the 622.08 Mbps SDH-based interface is STM-4 and is four times the size of the STM-1. It consists of an overhead of 9 × 4 columns and a payload part of 261 × 4 columns. The payload contains only one POH of one column, thus providing a payload capacity of 600.768 Mbps.

12.3.2. Cell-Based Interface

The cell-based interface is considerably simpler than the SDH-based interface because there is no framing involved. Transmission consists of a continuous stream of ATM cells.

To cater for OAM functions, ATM cells with unique header patterns are used. This pattern consists of setting the first three octets in the cell header to 0 and setting the fourth octet to the values shown in Figure 12.144. The latter determines the cell type which in turn classifies the OAM functions it caters for. OAM cells are inserted in the transmission stream on a per demand basis.

Figure 12.144 Physical layer OAM cells.

OAM Cell Type	Fourth Octet	Applies to Level	OAM Functions
F1	0000 0011	Regenerator Section	Frame alignment Section error monitoring

F2	0000 0101	Digital Section	Frame alignment Section error monitoring/reporting
F3	0000 1001	Transmission Path	Network status monitoring Header error monitoring Path error monitoring/reporting Cell delineation/rate decoupling

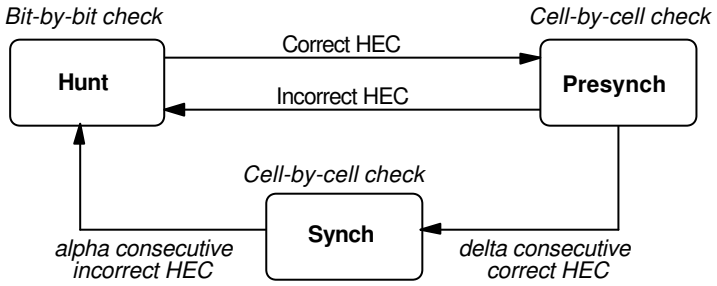
The main advantage of the cell-based interface is its inherent simplicity. The main advantage of the SDH-based interface is that it provides a means for combining multiple ATM channels to achieve a higher bit rate.

12.3.3. Cell Delineation

Because the cell-based interface involves no framing, some other means of synchronization is needed. Cell delineation provides this by identifying cell boundaries. The HEC field (which applies to the remaining fields in the header) is used for this purpose.

Cell boundary is identified by comparing potential header bits against their corresponding HEC bits using the state diagram shown in Figure 12.145.

Figure 12.145 Cell delineation state diagram.



A receiver is initially in the hunt state. Here a bit-by-bit checking of the cell stream is made with the objective of finding a header with matching HEC field. When a correct HEC is detected, the presynch state entered. In this state HEC fields are checked on a cell-by-cell basis. When *delta* (usually set to 6) consecutive correct HEC fields are detected the synch state is entered. While in presynch state, an incorrect HEC field results in a return to the hunt state. In the synch state HEC fields are still checked on a cell-by-cell basis. Detection of alpha (usually set to 7) consecutive incorrect HEC fields causes the receiver to revert back to the hunt state.

The information contents of cells is scrambled as a means of protection against imitated HEC field correlations which may have been generated accidently or maliciously. With even a relatively high error rate, the receiver will spend very long periods in the synch state and extremely short periods in the other two states.

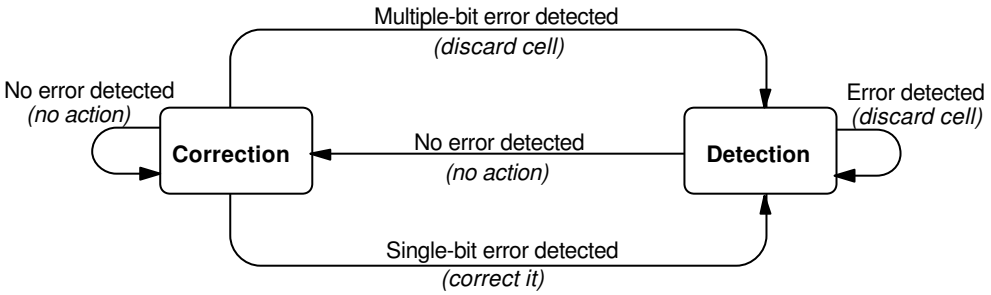
12.3.4. HEC Generation and Verification

The HEC field is generated by the transmitter using the first four octets of the cell header. The outcome is a single octet which becomes the final octet in the cell header. A HEC field is calculated as follows:

1. Multiply the polynomial denoted by the first 32 bits of the header (to which HEC applies) by x^8.
2. Divide the outcome by the generator polynomial $x^8 + x^2 + x + 1$ using modulo 2 division.
3. The resulting 8 bits are added to 01010101 using modulo 2 addition to produce the final HEC value.

Because the HEC field is fairly large compared to the data to which it applies, it also provides some scope for error correction. This includes all single-bit errors and some multiple-bit errors. Figure 12.146 illustrates the receiver behavior in response to HEC values.

Figure 12.146 HEC state diagram for a receiver.



The receiver is initially in the correction state. A single-bit error in this state is corrected and results in a transition to the detection state. A multi-bit error causes the cell being discarded and also results in a transition to the correction state. While in correction state, detection of any error causes the cell to be discarded. A correct cell, however, results in a return to the correction state. The receiver remains in the correction state as long as no errors are detected.

The above algorithm is designed to take into account the error behavior of optical fiber transmission media (i.e., a combination of single-bit errors and error bursts).

12.3.5. Cell Rate Decoupling

ATM provides an asynchronous means of cell transfer. However, actual physical transmission remain synchronous (fixed rate). Hence cells need to be transmitted even when the transmitter has no data to send. To adapt the cell rate to the

transmission rate, the transmitter inserts idle cells in the stream when it has no cells to send. The receiver identifies and discards idle cells upon their arrival.

Idle cells are similar to OAM cells in that they have their VPI set to 0 and their VCI set to a fixed value (1). The cell header conatins a valid HEC and the cells information octets are set to the bit pattern 01101010.

12.4. ATM Layer

The ATM layer uses a cell as its basic unit of communication. Unlike the physical layer, at the ATM layer only the logical structure of a cell is of interest. This was described in Section 12.2.2. Cells which are for the use of the physical layer only are distinguished by having the VPI and VCI set to 0, and the least significant bit of the fourth octet in the cell header set to 1.

Except for the HEC field which is not used by the ATM layer, the remaining ATM cell fields are separately discussed below.

12.4.1. Generic Flow Control

The GFC field is 4 bits wide and defaults to 0. The GFC field provides a means of exercising UNI-based control over the flow of traffic to counter overload situations. It only applies to traffic generated by the CPE as opposed to the network. The exact GFC details remain largely undefined.

The ATM network itself has no provision for flow control comparable to those provided by packet networks. This essentially refelects its streamlined nature.

12.4.2. Virtual Path Identifier

The VPI field is 8 bits wide for the UNI and 12 bits wide for the Network-Network Interface (NNI). It is used to distinguish between VP links multiplexed into the same physical connection. All cells belonging to the same VP are assigned the same VPI. As explained earlier, certain cells may be assigned predetermined VPIs. A VPI is mapped at a VP link termination, such as a switch.

A Virtual Path Connection (VPC) is a concatenation of VP links, along which the VCIs may be mapped link-by-link. A VPC preserves the cell sequence for each of the channels it contains.

12.4.3. Virtual Channel Identifier

The VCI field is 16 bits wide and is used to distinguish between VCs sharing the same VP connection. Combined with the VPI, it provides a complete routing identifier. All cells belonging to the same virtual channel are assigned the same VCI.

Predetrmined VCIs are used for cells serving special purposes. A VCI is mapped at a VC link termination, such as a switch.

A Virtual Channel Connection (VCC) is a concatenation of VC links. It preserves the cell sequence. At the UNI, a VCC may be established/released using different methods: through user-network signaling, through meta signaling, or as a result of subscription (without signaling).

12.4.4. Payload Type

This field is 2 bits wide and denotes the type of the information stored in the cell. A PT value of 00 is used for cells carrying user information. Other values can be used for representing control information to provide an inband signaling capability.

12.4.5. Cell Loss Priority

This is a single-bit field and hence may assume a value of 0 or 1. When set to 0, it indicates to the network that this cell should not be discarded as long as possible. In a congestion situation, therefore, the network will give preference to the discarding of cells with a CLP of 1. This field may be set by the user or the network.

12.5. ATM Adaptation Layer

The AAL supports the use of non-ATM protocols (e.g., LAP-D) for the transfer of information. It does so by mapping information from other protocols (e.g., LAP-D frames) into ATM cells before transmission and their reassembly upon arrival.

Four classes of AAL services have been identified on the basis of the timing relation between source and destination, the bit rate, and the connection mode. These are summarized by Figure 12.147. The AAL provides four protocol types (types 1-4) to, respectively, support these service classes.

Figure 12.147 AAL service classes.

Service Class	Timing Relation	Bit Rate	Connection Mode	Example
Class A	Required	Constant	Connection-oriented	Circuit emulation
Class B	Required	Variable	Connection-oriented	Video/Audio
Class C	Not required	Variable	Connection-oriented	Data transfer
Class D	Not required	Variable	Connectionless	Data transfer

As mentioned earlier, the AAL is divided into two sublayers: the Segmentation And Reassembly (SAR) sublayer and the Convergence Sublayer (CS). These are separately discussed below.

12.5.1. Segmentation and Reassembly Sublayer

SAR handles the segmentation of information from CS into ATM cells, and their subsequent reassembly upon arrival. Figure 12.148 illustrates the SAR PDU format for the four protocol types.

The Type 1 SAR PDU is intended for transmissions involving a constant bit rate source (see Class A in Figure 12.147). It consists of 8 bits of overhead and a payload of 47 octets. The overhead is comprised of two fields: a 4-bit Sequence Number (SN) and a 4-bit Sequence Number Protection (SNP). The SN is used for the detection of cell sequence violation (e.g., lost or misinserted cells). The SNP serves as a CRC for error detection and single-bit error correction related to the overhead.

The Type 2 SAR PDU (as well as Type 3 and 4) supports transmissions involving a variable bit rate source. It consists of a payload and four overhead fields. The SN is as before. The Information Type (IT) field is used to denote the relationship of the SAR PDU to the CS PDU (i.e., one of: beginning of message, continuation of message, end of message, or single-segment message). The Length Indicator (LI) denotes the actual number of CS PDU octets carried in this SAR PDU. The CRC field protects the SAR PDU against errors. Type 2 is suitable for the transmission of analog data, such as audio and video, using PCM.

Figure 12.148 SAR PDU formats.

Type	PDU Format					
1	SN (4 bits)	SNP (4 bits)	Payload (47 octets)			
2	SN (4 bits)	IT (4 bits)	Payload (45 octets)		LI (6 bits)	CRC (10 bits)
3	ST (2 bits)	SN (4 bits)	RES (10 bits)	Payload (44 octets)	LI (6 bits)	CRC (10 bits)
4	ST (2 bits)	SN (4 bits)	MID (10 bits)	Payload (44 octets)	LI (6 bits)	CRC (10 bits)

The Type 3 SAR PDU consists of 32 bits of overhead and a payload of 44 octets. The Segment Type (ST) field is similar to the IT field of Type 2. The SN, LI, and CRC fields are as before. The remaining 10 bits are reserved for future use. The Type 4 SAR PDU is very similar to Type 3, except that here a 10-bit Multiplexing Identifier (MID) is used. It supports the multiplexing of multiple CS PDUs over the same connection. Type 3 and 4 are suitable for data transfer applications.

Two modes of service are provided for Type 3 and 4: stream mode and messaging mode. The **stream mode service** involves the transfer of fixed-size

blocks, where each block is mapped to one cell. It is intended for continuous data at low rates. The **message mode service** is suitable for framed data (e.g., from LAP-D). Each frame is mapped to one or more cells.

12.5.2. Convergence Sublayer

The convergence sublayer is responsible for a variety of functions (depending on which of the AAL protocol types described above is used), including the following:

- Indication of the information contained by CS PDUs.
- Segmenation and reassembly of higher layer PDUs into CS PDUs.
- Handling of lost of misinserted cells (following their detection by the SAR sublayer).
- Detection of corrupted CS PDUs and their proper handling.
- Explicit time indication through time stamps (as required by some services).
- Clock recovery (as required by some services).

12.6. B-ISDN Standards

12.7. Further Reading

An introductory coverage of B-ISDN and ATM can be found in Minzer (1989), De Prycker (1991), Van Duuren *et al* (1992), Stallings (1992), and Onvural (1994). Kessler (1993) and Handel and Huber (1994) provide a wide coverage of ATM protocols and standards. Ballart and Ching (1989), Stallings (1992), Spohn (1993) describe SONET. De Prycker *et al* (1993) relate B-ISDN to the OSI reference model. Ahmadi and Denzel (1989) and Rooholamini *et al* (1994) describe a number of ATM switching techniques and products. Onvural (1994) presents an in-depth treatment of ATM network performance issues. Viniotis and Onvural (1993) is a useful collection of paper on ATM networks.

Bibliography

12, Ahmadi, H. and Denzel W. E. (1989) 'A Survey of Modern High-Performance Switching Techniques,' *IEEE Journal on Selected Areas in Communications*, Vo. 7(7), pp. 1091-1103.

12 Ballart, R. and Ching, Y. (1989) 'SONET: Now It's the Standard Optical Network,' *IEEE Communications Magazine*, March issue, pp.

10 Bellamy, J. (1991) *Digital Telephony*, Second Edition, Wiley, NY.

2 Bic, J. C., Duponteil, D., and Imbeaux, J. C. (1991) *Elements of Digital Communication*, Wiley, Chichester, UK.

2 Black, U. (1988), *Physical Level Interfaces and Protocols*, IEEE Computer socienty Press, Washington DC.

1, 3 Black, U. (1989), *Data Networks: Concepts, Theory, and Practice*, Prentice Hall, Englewood Cliffs, NJ.

11 Black, U. (1994) *Frame Relay Networks: Specification and Implementation*, McGraw-Hill, NY.

4 Black, U. (1992) *TCP/IP and Related Protocols*, McGraw-Hill, NY.

2

Blahut R. (1990) *Digital Transmission of Information*, Addison-Wesley, Reading, MA.

1 Brown, W., Simpson, C., and Black, U. (1993) *The OSI Dictionary of Acronyms and Related Abbreviations*, McGraw-Hill, NY.

10 Bush, S. and Parsosns, C. (1992) *Private Branch Exchange Systems and Applications*, McGraw-Hill, NY.

11 Chen, K. and Rege, K. (1989) 'A Comparative Performance Study of Various Congestion Controls for ISDN Frame Relay Networks,' *Proceedings IEEE INFOCOM '89*, April.

8 Chilton, P. A. (1989) *Introducing X.400*, NCC Publications, Manchester, UK.

9 Chorafas, D. (1989) *Local Area Network Reference*, McGraw-Hill, NY.

4 Comer, D. (1993), *Internetworking with TCP/IP, Volume I: Principles, Protocols, and Architectures*, Prentice Hall, Englewood Cliffs, NJ.

1 De Noia, L. (1987) *Data Communications: Fundamentals and Applications*, Merrill/Macmillan, NY.

12 De Prycker, M. (1991) *Asynchronous Transfer Mode Solutions for Broadband ISDN*, Eliis Horwood, New York, NY.

12 De Prycker, M., Peschi, R., and Van Landegem, T. (1993), 'B-ISDN and the OSI Protocol Reference Model,' *IEEE Network*, Vol 7(2), pp. 10-18.

11 Deniz, D. (1993) *ISDN and its Application to LAN Interconnection*, McGraw-Hill, NY.

4 Dhas, C. and Konangi, U. (1986) 'X.25: An Interface to Public Packet Networks,' *IEEE Communications Magazine*, September issue.

1,7 Dickson, G. and LLoyd, A. (1992), *Open Systems Interconnection: Computer Communications Standards and GOSIP Explained*, Prentice Hall Australia.

2

F. J. McClimans (1992) *Communication Wiring and Interconnection*, McGrawHill, NY.

4 Feit, S. (1992) *TCP/IP: Architecture, Protocols, and Implementation*, McGraw-Hill, NY.

9 Fortier, P. (1992) *Handbook of LAN Technology*, Second Edition, McGraw-Hill, NY.

4, 10 Freeman, R. (1989), *Telecommunication System Engineering*, Second Edition, Wiley, NY.

2, 3 Gitlin, R. D., Hayes, J. F., and Weinstein, S. B. (1992) *Data Communication Principles*, Plenum, New York, NY.

9 Gohring, H. and Kauffels, F. (1992) *Token Ring: Principles, Perspectives and Strategies*, Addison-Wesley, Reading, MA.

11 Griffiths, J. (1990) *ISDN Explained: Worldwide Network and Applications Technology*, Wiley, Chichester, UK.

1, 3 Halsall, F. (1992), *Data Communications, Computer Networks and Open Systems*, Third Edition, Addison Wesley, Wokingham, UK.

12 Handel, R. and Huber M. (1994) *ATM: An Introduction to Integrated Broadband Networks*, Second Edition, Addison-Wesley, Wokingham, UK.

9 Hegering, H. and Lapple A. (1993) *Ethernet: Building a Communications Infrastructure*, Addison-Wesley, Reading, MA.

10 Heldman, R. (1993) *Future Telecommunications: Information Applications, Services, and Infrastructure*, McGraw-Hill, NY.

10 Heldman, R. (1994) *Information Telecommunications: Networks, Products, and Services*, McGraw-Hill, NY.

11 Helgert, H. (1991) *Integrated Services Digital Networks*, Addison-Wesley, Reading, MA.

1 Hughes, L. (1992) *Data Communications*, McGraw-Hill, NY.

9 Hunter, P. (1993) *Local Area Networks: Making the Right Choice*, Addison-Wesley, Reading, MA.

10 Jabbari, B. (1991) 'Common Channel Signaling System Number 7 for ISDN and Intelligent Networks,' *Proceedings of the IEEE*, February issue.

9 Jain, R. (1993) *FDDI Handbook: High-Speed Networking Using Fiber and Other Media*, Addison-Wesley, Reading, MA.

1 Jain, B. and Agrawala, A. (1993) *Open Systems Interconnection: Its Architecture and Protocols*, McGraw-Hill, NY.

11, 12 Kessler, G. (1993) *ISDN: Concepts, facilities, and Services*, Second Edition, McGraw-Hill, NY.

9 Kessler, G. and Train, D. (1992) *Metropolitan Area Networks: Concepts, Standards, and Service*, McGraw-Hill, NY.

11 Lai, W. (1989) 'Frame Relaying service: An Overview,' *Proceedings IEEE INFOCOM '89*, April.

9 Layland, R. (1994) *LAN Internetworking: Building the Corporate Enterprise Network for the 90's*, Addison-Wesley, Reading, MA.

4 Lynch, D. and Rose, M. (1993) *The Internet System Handbook,* Addison-Wesley, Reading, MA.

1 Marshall, R. (1990), *The Open Book: A Practical Perspective on OSI*, Prentice Hall, Englewood Cliffs, NJ.

1, 3 Martin, J. and Leben, J. (1988), *Principles of Data Communication*, Prentice Hall, Englewood Cliffs, NJ.

10 Martin, J. (1990) *Telecommunications and the Computer*, Prentice Hall, Englewood Cliffs, NJ.

9 Martin, J., Chapman, K., and Leben, J. (1994) *Local area Networks: Architectures and Implementations*, Prentice Hall, Englewood Cliffs, NJ.

12 Minzer, S. (1989) 'Broadband ISDN and Asynchronous Transfer Mode (ATM),' *IEEE Communication Magazine*, September issue, pp. 17-24.

10 Modarressi, A. and Skoog, R. (1990) 'Signaling System Number 7: A Tutorial,' *IEEE Communications Magazine*, July issue.

9 Nemzow, M. (1994) *FDDI Networking: Planning, Installation and Management*, McGraw-Hill, NY.

12 Onvural, R. (1994) *Asynchronous Transfer Mode Networks: Performance Issues*, Artech House, Norwood, MA.

4 Perlman, R. (1992) *Interconnection: Bridges and Routers in OSI and TCP/IP*, Addison-Wesley, Reading, MA.

4 Piscitello, D. and Chapin, L. (1993) *Open Systems Networking TCP/IP and OSI*, Addison-Wesley, Reading, MA.

8 Plattner, B., Lanz, C., Lubich, H., Muller, M, and Walter, T. (1991) *X400 Message Handling: Standards, Interworking, Applications*, Addison-Wesley, Reading, MA.

8 Radicati, S. (1992) *Electronic Mail: An Introduction to the X.400 Message handling Standards*, McGraw-Hill, NY.

12 Rooholamini, R., Cherkassky, V., and Garver, M. (1994) 'Finding the Right ATM Switch for the Market,' *IEEE Computer*, pp. 16-28, April 94.

4 Schlar, S. (1990) *Inside X.25: A Manager's Guide*, McGraw-Hill, NY.

5, 10 Schwartz, M. (1987), *Telecommunications Networks: Protocols, Modelling and Analysis*, Addison Wesley, Reading, MA.

11 Smith, P. (1993) *Frame Relay*, Addison-Wesley, Reading, MA.

12 Spohn, D. (1993) *Data Network Design*, McGraw-Hill, NY.

10 Spragins, J., Hammond, J., and Pawlikowski (1991) *Telecommunication Networks: Protocols and Design*, Addison-Wesley, Reading, MA.

1, 4 Stallings, W. (1990), *Handbook of Computer Communications Standards*, *Volumes I and II*, Howard Sams and Company, Carmel.

12 Stallings, W. (1992), *ISDN and Broadband ISDN*, Second Edition, Macmillan, NY.

1, 3 Stallings, W. (1994), *Data and Computer Communications*, Fourth Edition, Macmillan, NY.

9 Stallings, W. (1993a), *Local and Metropolitan Area Networks*, Fourth Edition, Macmillan, NY.

4 Stallings, W. (1993b) *Networking Standards: A Guide to OSI, ISDN, LAN, and MAN Standards*, Addison-Wesley, Reading, MA.

9 Stamper, D. (1993) *Local Area Networks*, Addison-Wesley, Reading, MA.

1, 4 Stamper, D. (1991) *Business Data Communications*, Third Edition, Addison-Wesley, Reading, MA.

2 Stone, H. (1982), *Microcomputer Interfacing*, Addison-Wesley, Reading, MA.

1, 4 Tanenbaum, A. (1989), *Computer Networks*, Second Edition, Prentice Hall, Englewood Cliffs, NJ.

10, 12 Van Duuren, J., Schoute, F., and Kastelein, P. (1992) *Telecommunications Networks and Services*, Addison-Wesley, Reading, MA.

12 Viniotis Y. and Onvural R. (editors) (1993) *Asynchronous Transfer Mode Networks*, Plenum, New York, NY.

4 White, G. (1992) *Internetworking and Addressing*, McGraw-Hill, NY.

9 Zitsen, W. (1990) 'Metropolitan Area Networks: Taking LANs into the Public Network,' *Telecommunications*, pp. 53-60.