COMPUTER NETWORKS

(Common to CSE & IT)

B.Tech – V Semester (20IT502)

Prerequisites

- None

Course Objectives

- Students will be able to

COB 1: To learn the architectural principles of data communications and computer networking.

COB 2: To learn the network layer design, routing algorithms and congestion control algorithms.

COB 3: To know the quality of services, transport services and Elements of transport Protocols.

COB 4: To gain the knowledge on internet transport protocols (UDP, TCP) and DNS application.

Course Outcomes

- After the course the students are expected to be able to

CO 1: Understand the architectural principles of data communications and computer networking.

CO 2: Understand the network layer design, routing algorithms and congestion control algorithms.

CO 3: Understand the quality of services, transport services and elements of transport layer protocols.

CO 4: Understand the knowledge on internet transport protocols (UDP, TCP) and DNS application.

UNIT I

- **Data Communications & Networking Overview**: A Communications Model, Data Communications, Data Communication Networking.
- **Protocol Architecture**: The Need for a Protocol Architecture, A Simple Protocol Architecture, OSI, The TCP/IP Protocol Architecture.
- **Digital Data Communication Techniques**: Asynchronous & Synchronous Transmission, Types of Errors, Error Detection, Error Correction.
- **Data Link Control**: Flow Control, Error Control.

UNIT II

- **Network Layer Design Issues:** Store and Forward Packet Switching, Services Provided to the Transport Layer, Implementation of Connectionless Service, Implementation of Connection Oriented Service, Comparison of Virtual Circuit & Datagram Subnets.
- **Routing Algorithms:** The Optimality Principle, Shortest Path Routing, Flooding, Distance Vector Routing, Link State Routing, Hierarchical Routing.
- **Congestion Control Algorithms:** General Principles of Congestion Control, Congestion Prevention Policies, Congestion Control in Virtual Circuit Subnets, Congestion Control in Datagram Subnets, Load Shedding, Jitter Control.

UNIT III

- **Quality of Service:** Requirements, Techniques for Achieving Good Quality of Service
- **The Network Layer in the Internet:** The IP Protocol, IP Addresses, Internet Control Protocols.
- **The Transport Service:** Services Provided to the Upper Layers, Transport Service Primitives.
- **Elements of Transport Protocols:** Addressing, Connection Establishment, Connection Release, Flow Control and Buffering, Multiplexing, Crash Recovery.

UNIT IV

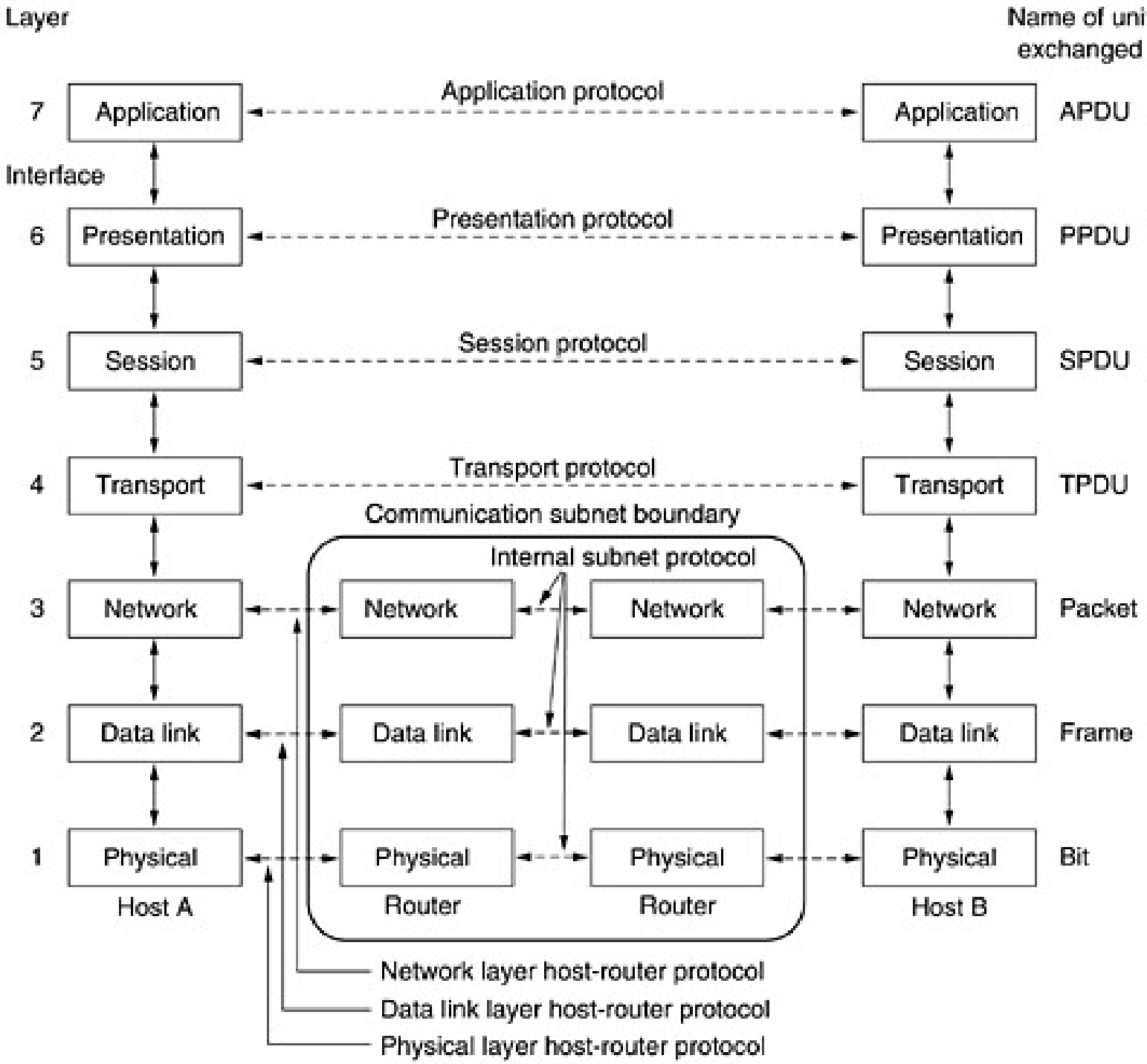
- **The Internet Transport Protocol (UDP):** Introduction to UDP, Remote Procedure Call, The Real Time Transport Protocol.
- **The Internet Transport Protocol (TCP):** Introduction to TCP, The TCP Service Model, The TCP Protocol, The TCP Segment Header, TCP Connection Establishment, TCP Connection Release, Modeling TCP Connection Management, TCP Transmission Policy, TCP Congestion Control, TCP Timer Management.
- **The Domain Name System (DNS):** The DNS Name Space, Resource Records, Name Servers.

TEXT BOOKS

1. William Stallings, Data and Computer Communications, 7th edition, Pearson Education, 2005. **(Unit - 1)**
2. Tanenbaum, Computer Networks, 4th edition, Pearson Education, 2008. **(Unit – 2 to 4)**

COMPUTER NETWORKS

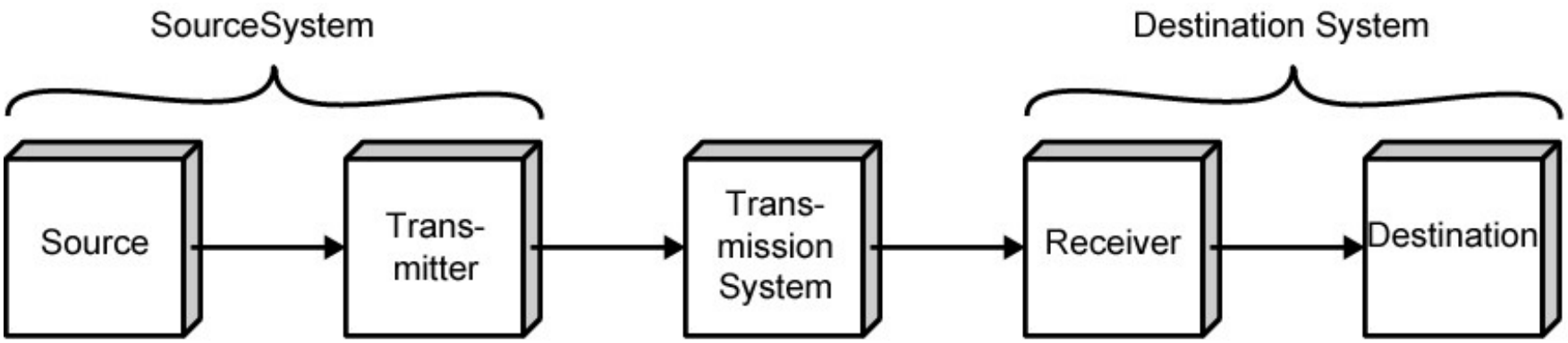
- It is a collection of autonomous computers “interconnected” by a single technology.
- Two computers are said to be interconnected if they are able to exchange information between them via copper wire, fiber optics, communication satellites.


Layer
Name of unit exchanged
7
Application
Application protocol
Application
APDU
Interface
6
Presentation
Presentation protocol
Presentation
PPDU
5
Session
Session protocol
Session
SPDU
4
Transport
Transport protocol
Transport
TPDU
Communication subnet boundary
Internal subnet protocol
3
Network
Network
Network
Network
Packet
2
Data link
Data link
Data link
Data link
Frame
1
Physical
Physical
Physical
Physical
Bit
Host A
Router
Router
Host B
Network layer host-router protocol
Data link layer host-router protocol
Physical layer host-router protocol

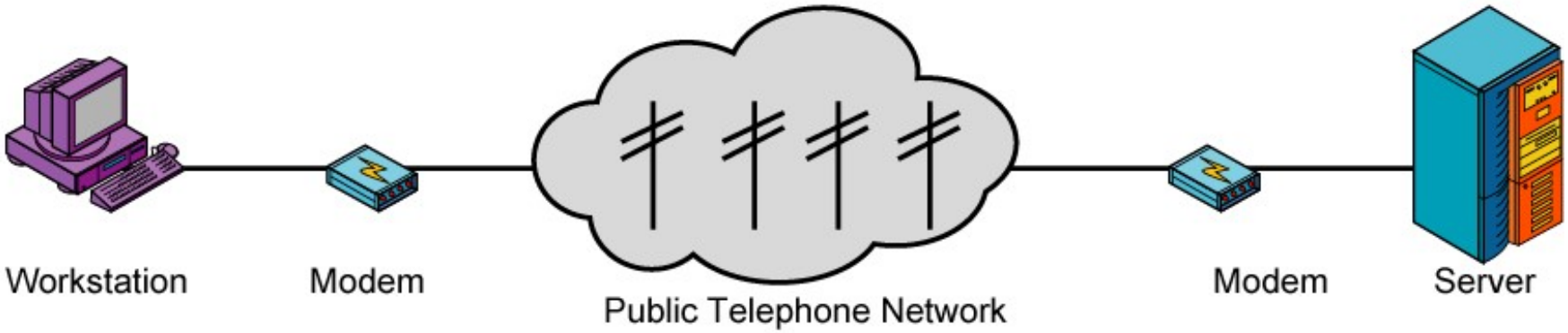
A Communications Model

- The fundamental purpose of a communications system is the exchange of data between two parties.
- A simple model of communications, illustrated by the block diagram in Figure (a).
- Figure (b) presents one particular example, which is communication between a workstation and a server over a public telephone network.
- Another example is the exchange of voice signals between two telephones over the same network.

A Communications Model



(a) General block diagram



(b) Example

A Communications Model

- The key elements of the model are as follows:
  - **Source -** This device generates the data to be transmitted; examples are telephones and personal computers.
  - **Transmitter -** Usually, the data generated by a source system are not transmitted directly in the form in which they were generated. Rather, a transmitter transforms and encodes the information in such a way as to produce electromagnetic signals that can be transmitted across some sort of transmission system. For example, a modem takes a digital bit stream from an attached device such as a personal computer and transforms that bit stream into an analog signal that can be handled by the telephone network.

A Communications Model

- **Transmission system -** This can be a single transmission line or a complex network connecting source and destination.
- **Receiver -** The receiver accepts the signal from the transmission system and converts it into a form that can be handled by the destination device. For example, a modem will accept an analog signal coming from a network or transmission line and convert it into a digital bit stream.
- **Destination -** Takes the incoming data from the receiver.

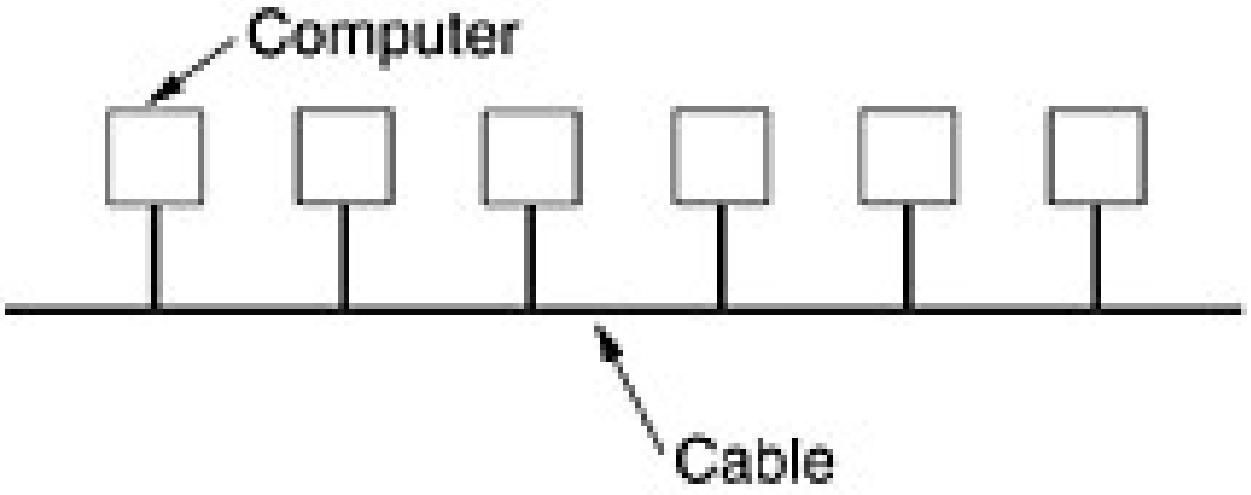
A Communications Model

- Below table lists some of the key tasks that must be performed in a data communications system.

Transmission system utilization	Addressing
Interfacing	Routing
Signal generation	Recovery
Synchronization	Message formatting
Exchange management	Security
Error detection and correction	Network management
Flow control	

A Communications Model

- The **transmission system utilization**, refers to the need to make efficient use of transmission facilities that are typically shared among a number of communicating devices.
- Various techniques (referred to as multiplexing) are used to allocate the total capacity of a transmission medium among a number of users.
- Congestion control techniques may be required to assure that the system is not overwhelmed by excessive demand for transmission services


Computer
Cable

A Communications Model

- To communicate, a device must **interface** with the transmission system. All the forms of communication depend on the use of electromagnetic signals propagated over a transmission medium.
- Once an interface is established, **signal generation** is required for communication. The properties of the signal, such as form and intensity, must be such that the signal is (1) capable of being propagated through the transmission system, and (2) interpretable as data at the receiver.

A Communications Model

- Not only must the signals be generated to conform to the requirements of the transmission system and receiver, but also there must be some form of **synchronization** between transmitter and receiver.
- The receiver must be able to determine when a signal begins to arrive and when it ends. It must also know the duration of each signal element.

A Communications Model

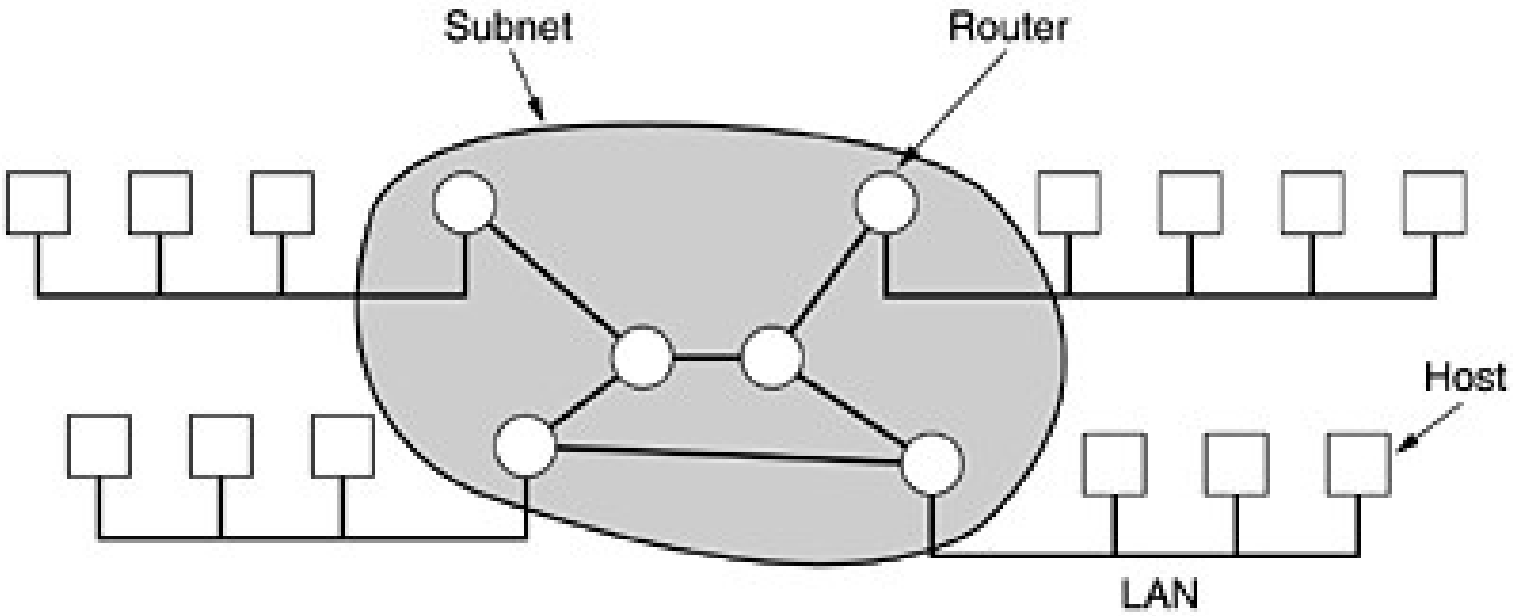
- Beyond the basic matter of deciding on the nature and timing of signals, there is a variety of requirements for communication between two parties that might be collected under the term **exchange management**.
- If data are to be exchanged in both directions over a period of time, the two parties must cooperate. For example, for two parties to engage in a telephone conversation, one party must dial the number of the other, causing signals to be generated that result in the ringing of the called phone. The called party completes a connection by lifting the receiver.
- For data processing devices, more will be needed than simply establishing a connection; certain conventions must be decided on. These conventions may include **whether both devices may transmit simultaneously or must take turns, the amount of data to be sent at one time, the format of the data, and what to do if certain contingencies such as an error arise.**

A Communications Model

- In all communications systems, there is a potential for error; transmitted signals are distorted to some extent before reaching their destination.
- **Error detection and correction** are required in circumstances where errors cannot be tolerated. This is usually the case with data processing systems. For example, in transferring a file from one computer to another, it is simply not acceptable for the contents of the file to be accidentally altered.
- **Flow control** is required to assure that the source does not overwhelm the destination by sending data faster than they can be processed and absorbed.

A Communications Model

- Next are the related but distinct concepts of **addressing** and **routing**.
- When more than two devices share a transmission facility, a source system must indicate the identity of the intended destination. The transmission system must assure that the destination system, and only that system, receives the data.
- Further, the transmission system may itself be a network through which various paths may be taken. A specific route through this network must be chosen.


Subnet
Router
Host
LAN

A Communications Model

- **Recovery** is a concept distinct from that of error correction.
- Recovery techniques are needed in situations in which an information exchange, such as a database transaction or file transfer, is interrupted due to a fault somewhere in the system.
- The objective is either to be able to **resume activity at the point of interruption** or at least to **restore the state of the systems involved to the condition prior to the beginning of the exchange**.

A Communications Model

- **Message formatting** has to do with an agreement between two parties as to the form of the data to be exchanged or transmitted, such as the binary code for characters.

A Communications Model

- Frequently, it is important to provide some measure of **security** in a data communications system.
- The sender of data may wish to be assured that only the intended receiver actually receives the data.
- And the receiver of data may wish to be assured that the received data have not been altered in transit and that the data actually come from the purported sender

A Communications Model

- A data communications facility is a complex system that cannot create or run itself.
- **Network management** capabilities are needed to configure the system, monitor its status, react to failures and overloads, and plan intelligently for future growth.

Data Communications

- The figure 2 provides a new perspective on the communications model of figure 1. We trace the details of this figure using electronic mail as an example.

Data Communications

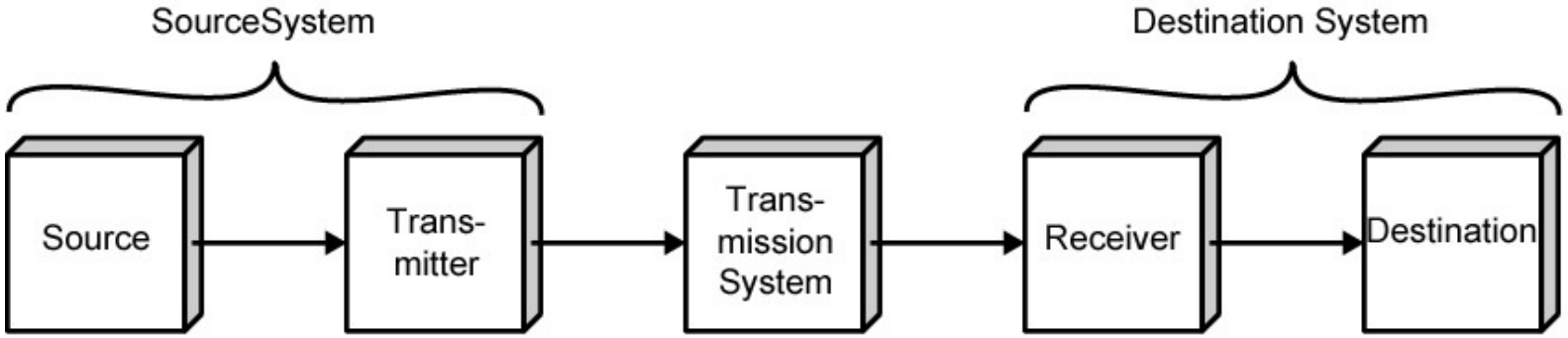


Figure 1: General Block Diagram

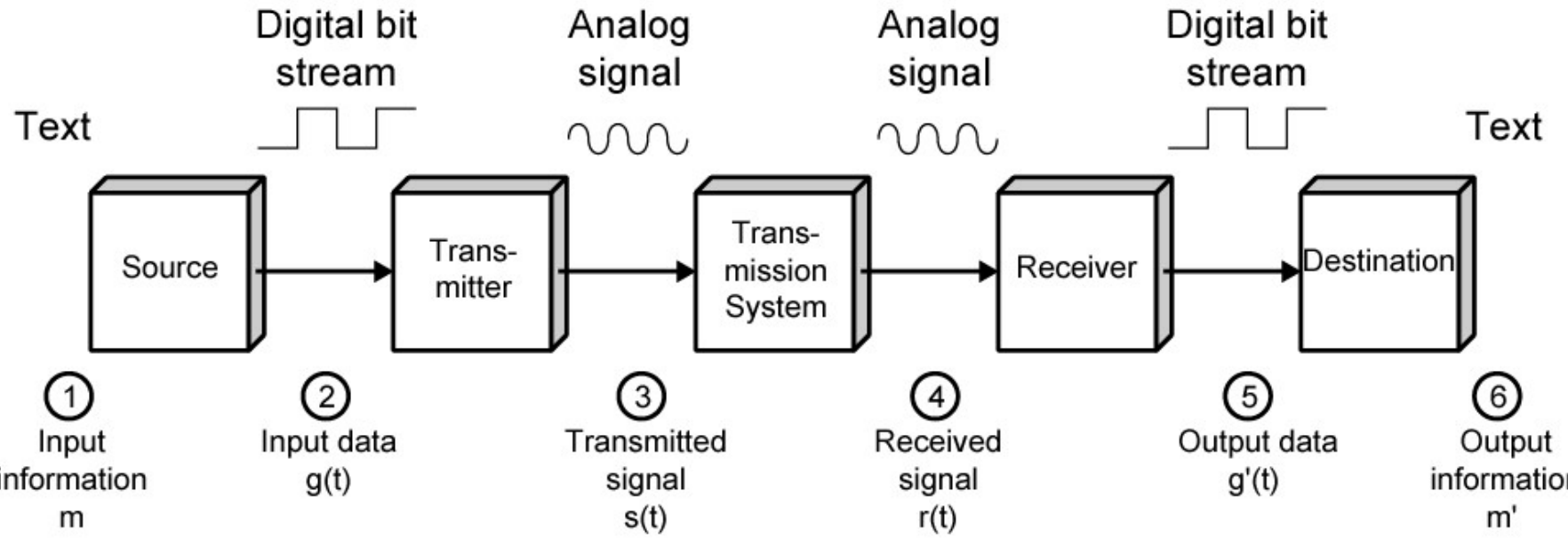


Figure 2: Simplified Data Communications Model

Data Communications

- Suppose that the input device and transmitter are components of a personal computer.
- The user of the PC wishes to send a message ***m*** to another user. The user activates the electronic mail package on the PC and enters the message via the keyboard (input device).
- The character string is briefly buffered in main memory. We can view it as a sequence of bits ***(g)*** in memory.
- The personal computer is connected to some transmission medium, such as a local network or a telephone line, by an I/O device (transmitter), such as a local network transceiver or a modem.
- The input data are transferred to the transmitter as a sequence of voltage shifts ***[g(t)]*** representing bits on some communications bus or cable. The transmitter is connected directly to the medium and converts the incoming stream ***[g(t)]*** into a signal ***[s(t)]*** suitable for transmission.

Data Communications

- The transmitted signal ***s(t)*** presented to the medium is subject to a number of impairments before it reaches the receiver.
- Thus, the received signal ***r(t)*** may differ from ***s(t)***. The receiver will attempt to estimate the original ***s(t)***, based on ***r(t)*** and its knowledge of the medium, producing a sequence of bits ***g'(t)***.
- These bits are sent to the output personal computer, where they are briefly buffered in memory as a block of bits ***(g')***.
- In many cases, the destination system will attempt to determine if an error has occurred and, if so, cooperate with the source system to eventually obtain a complete, error-free block of data.
- These data are then presented to the user via an output device, such as a printer or screen. The message ***(m')*** as viewed by the user will usually be an exact copy of the original message ***(m)***.

Data Communications

- Consider a telephone conversation.
- In this case the input to the telephone is a message ***(m)*** in the form of sound waves.
- The sound waves are converted by the telephone into electrical signals of the same frequency. These signals are transmitted without modification over the telephone line.
- Hence the input signal ***g(t)*** and the transmitted signal ***s(t)*** are identical. The signals ***s(t)*** will suffer some distortion over the medium, so that ***r(t)*** will not be identical to ***s(t)***.
- Nevertheless, the signal ***r(t)*** is converted back into a sound wave with no attempt at correction or improvement of signal quality. Thus, ***m'*** is not an exact replica of ***m***. However, the received sound message is generally comprehensible to the listener.

Data Communication Networking

- Wide Area Networks
  - Circuit Switching
  - Packet Switching
  - Frame Relay
  - ATM
- Local Area Networks
- Wireless Networks
- Metropolitan Area Networks

Data Communication Networking

- It is often impractical for two communicating devices to be directly, point-to-point connected. This is so for one (or both) of the following contingencies:
  - The devices are very far apart. It would be inordinately expensive, for example, to string a dedicated link between two devices thousands of kilometers apart.
  - There is a set of devices, each of which may require a link to many of the others at various times. Examples are all of the telephones in the world and all of the terminals and computers owned by a single organization. Except for the case of a very few devices, it is impractical to provide a dedicated wire between each pair of devices.

Data Communication Networking

- The solution to this problem is to attach each device to a communication network. There are two major categories into which communications networks are traditionally classified: Wide Area Networks (WANs) and Local Area Networks (LANs).
- The distinction between the two, both in terms of technology and application, has become somewhat blurred in recent years.

Wide Area Networks

- Wide Area Networks generally cover a large geographical area, require the crossing of public right-of-ways, and rely at least in part on circuits provided by a common carrier.
- WAN consists of a number of interconnected switching nodes. A transmission from any one device is routed through these internal nodes to the specified destination device.
- These nodes (including the boundary nodes) are not concerned with the content of the data; rather, their purpose is to provide a switching facility that will move the data from node to node until they reach their destination.
- Traditionally, WANs have been implemented using one of two technologies: **circuit switching and packet switching**. More recently, frame relay and ATM networks have assumed major roles.

Wide Area Networks

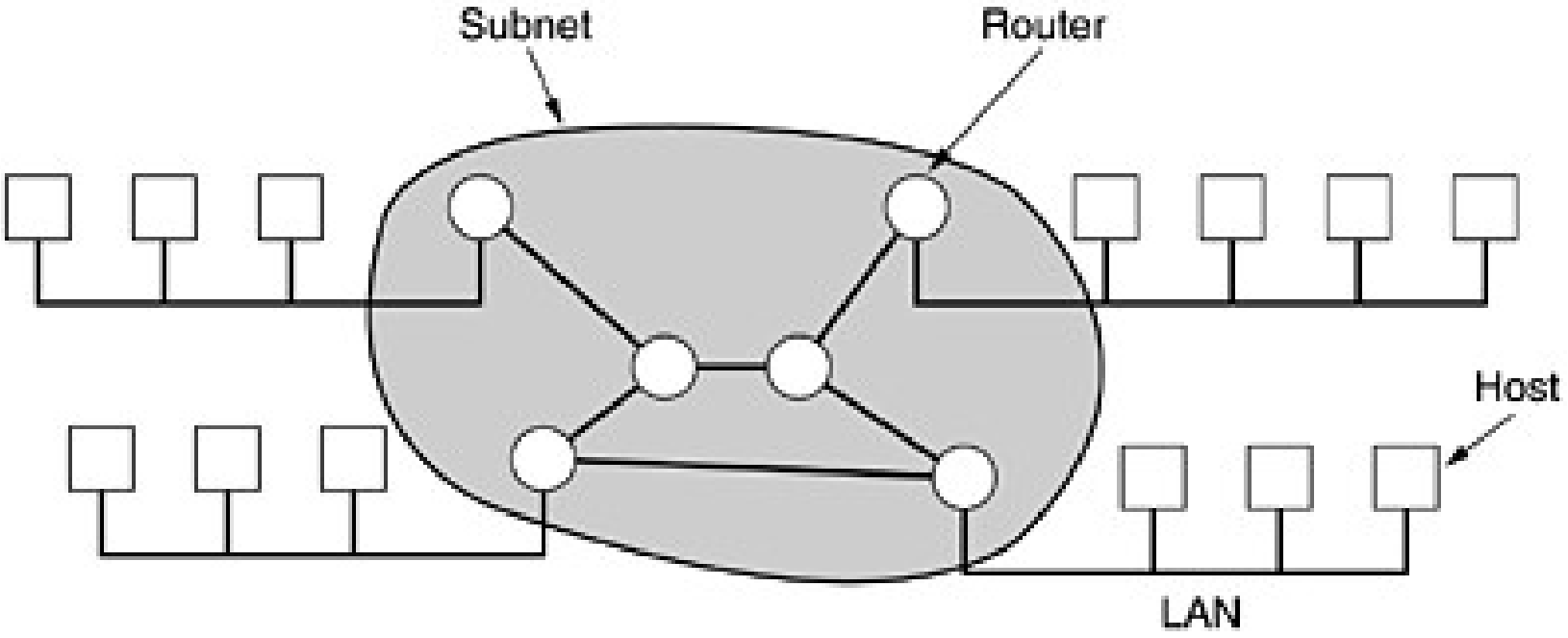
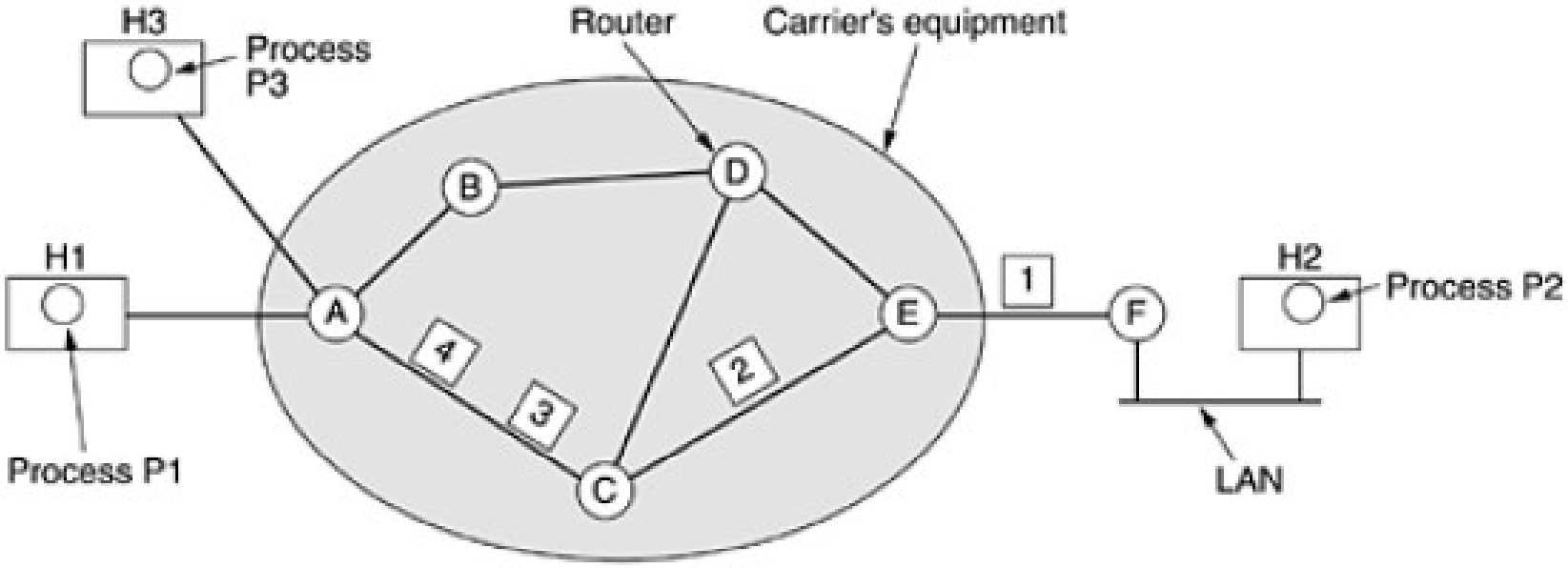


Figure: Relation between hosts on LANs and the subnet

Circuit Switching

- In a circuit-switching network, a dedicated communications path is established between two stations through the nodes of the network.
- That path is a connected sequence of physical links between nodes. On each link, a logical channel is dedicated to the connection.
- Data generated by the source station are transmitted along the dedicated path as rapidly as possible.
- At each node, incoming data are routed or switched to the appropriate outgoing channel without delay.
- The most common example of circuit switching is the telephone network.

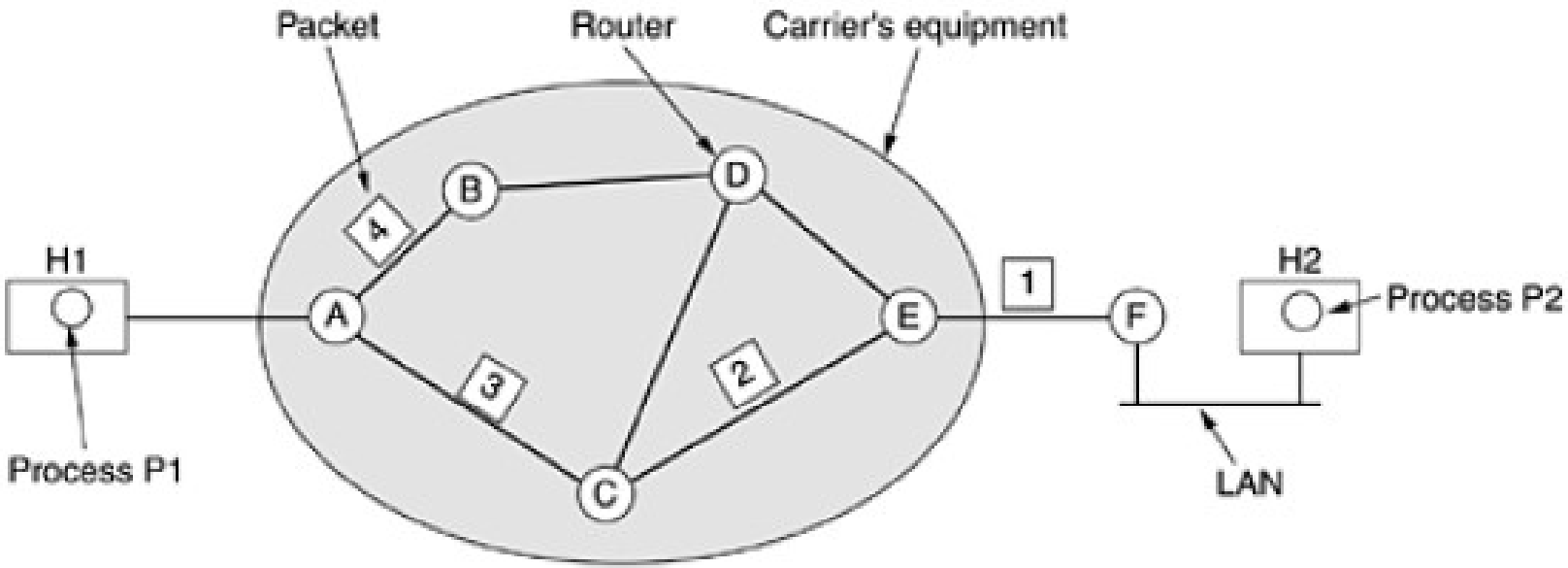
Circuit Switching



Packet Switching

- A quite different approach is used in a packet-switching network.
- In this case, it is not necessary to dedicate transmission capacity along a path through the network.
- Rather, data are sent out in a sequence of small chunks, called packets. Each packet is passed through the network from node to node along some path leading from source to destination.
- At each node, the entire packet is received, stored briefly, and then transmitted to the next node.
- Packet-switching networks are commonly used for terminal-to-computer and computer-to-computer communications.

Packet Switching



Circuit & Packet Switching

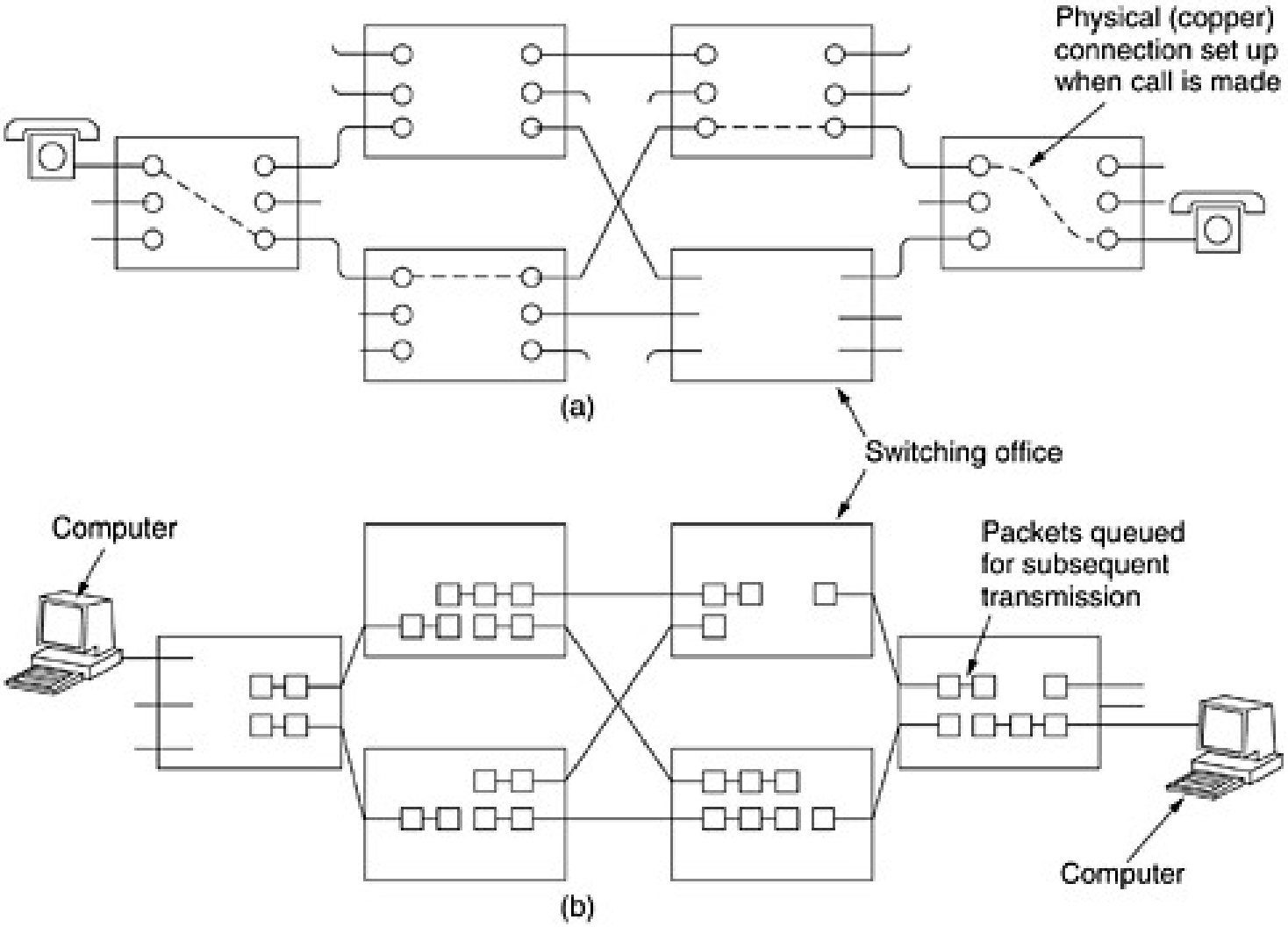


Figure: (a) Circuit switching. (b) Packet switching

Frame Relay

- Frame relay is a packet-switching telecommunications service designed for cost-efficient data transmission for intermittent traffic between local area networks (LANs) and between endpoints in wide area networks (WANs).
- Frame relay puts data in a variable-size unit called a frame and leaves any necessary error correction, or retransmission of data, up to the endpoints. This approach speeds up data transmission.

Frame Relay

- Frame relay was developed to take advantage of high data rates and low error rates.
- Whereas the original packet-switching networks were designed with a data rate to the end user of about 64 kbps, frame relay networks are designed to operate efficiently at user data rates of up to 2 Mbps.
- The key to achieving these high data rates is to strip out most of the overhead involved with error control.

ATM

- **Asynchronous transfer mode (ATM)**, is also referred as **cell relay**, is a culmination of developments in circuit switching and packet switching.
- ATM can be viewed as an evolution from **frame relay**.
- The difference between frame relay and ATM is that frame relay uses variable-length packets (53 bytes i.e., 5 byte header & 48 bytes of ATM payload), called **frames**, and ATM uses fixed-length packets, called **cells**.
- As with frame relay, ATM provides little overhead for error control, depending on the inherent reliability of the transmission system and on higher layers of logic in the end systems to catch and correct errors.
- By using a fixed packet length, the processing overhead is reduced even further for ATM compared to frame relay.
- The result is that ATM is designed to work in the range of 10s and 100s of Mbps, and in the Gbps range.

ATM

- ATM can also be viewed as an evolution from circuit switching. With circuit switching, only fixed-data-rate circuits are available to the end system.
- ATM allows the definition of multiple virtual channels with data rates that are dynamically defined at the time the virtual channel is created.
- By using small, fixed-size cells, ATM is so efficient that it can offer a constant-data-rate channel even though it is using a packet-switching technique.
- Thus, ATM extends circuit switching to allow multiple channels with the data rate on each channel dynamically set on demand.

Local Area Networks

- A LAN is a communications network that interconnects a variety of devices and provides a means for information exchange among those devices.

Local Area Networks

- There are several key distinctions between LANs and WANs:
  1. The scope of the LAN is small, typically a single building or a cluster of buildings. This difference in geographic scope leads to different technical solutions.
  2. It is usually the case that the LAN is owned by the same organization that owns the attached devices. For WANs, this is less often the case, or at least a significant fraction of the network assets is not owned. This has two implications. First, care must be taken in the choice of LAN, because there may be a substantial capital investment (compared to dial-up or leased charges for WANs) for both purchase and maintenance. Second, the network management responsibility for a LAN falls solely on the user.
  3. The internal data rates of LANs are typically much greater than those of WANs.

Local Area Networks

- LANs come in a number of different configurations.
- The most common are switched LANs and wireless LANs. The most common switched LAN is a switched Ethernet LAN, which may consist of a single switch with a number of attached devices, or a number of interconnected switches.
- Two other prominent examples are ATM LANs, which simply use an ATM network in a local area, and Fibre Channel.
- Wireless LANs use a variety of wireless transmission technologies and organizations.

Local Area Networks

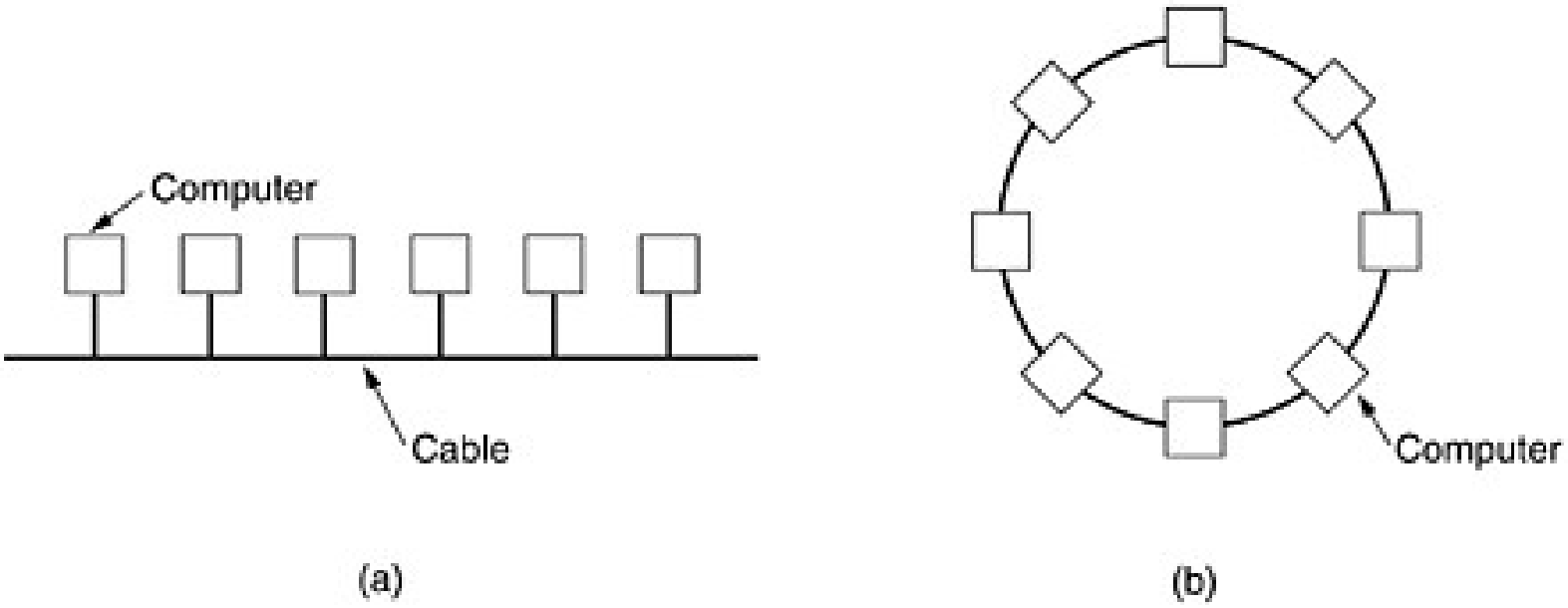


Figure: Two broadcast networks. (a) Bus. (b) Ring

Wireless Networks

- Wireless LANs are common, being widely used in business environments.
- Wireless technology is also common for both wide area voice and data networks.
- Wireless networks provide advantages in the areas of mobility and ease of installation and configuration.

Wireless Networks

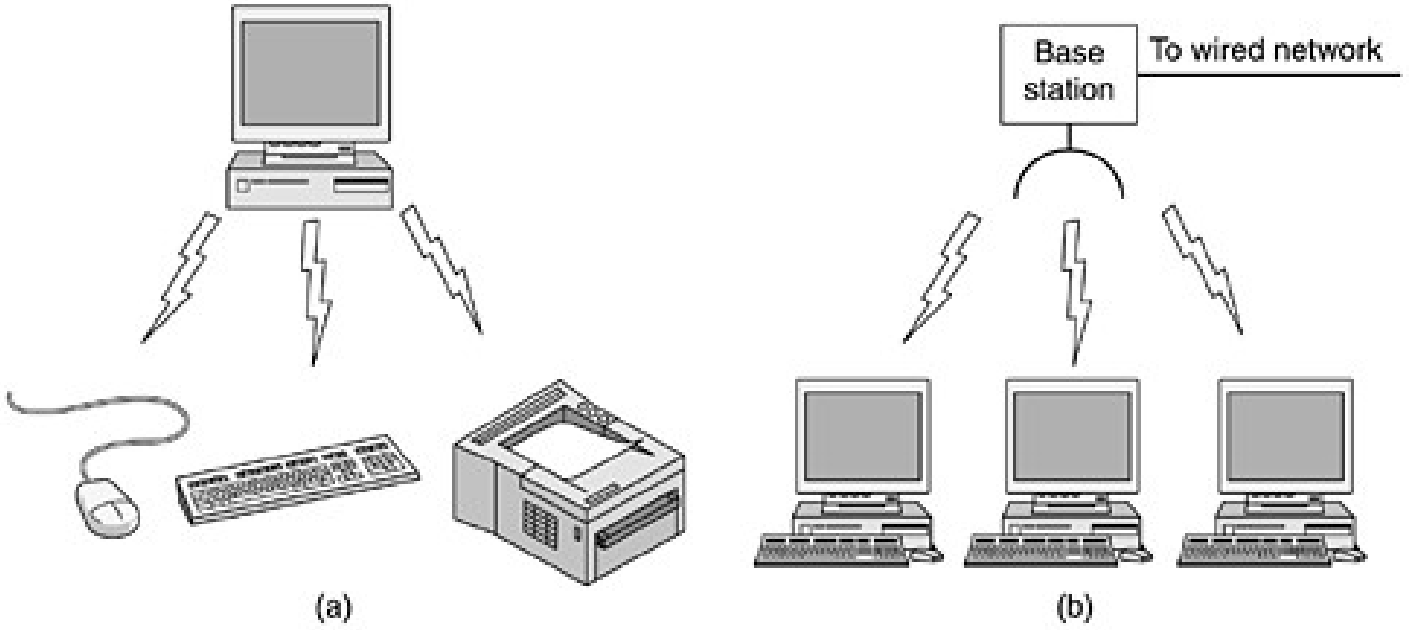


Figure: (a) Bluetooth configuration. (b) Wireless LAN.

Metropolitan Area Networks

- A MAN occupies a middle ground between LANs and WANs.
- Interest in MANs has come about as a result of a recognition that the traditional point-to-point and switched network techniques used in WANs may be inadequate for the growing needs of organizations.
- While frame relay and ATM promise to meet a wide range of high-speed needs, there is a requirement now for both private and public networks that provide high capacity at low costs over a large area.
- A number of approaches have been implemented, including wireless networks and metropolitan extensions to Ethernet.
- The primary market for MANs is the customer that has high-capacity needs in a metropolitan area.
- A MAN is intended to provide the required capacity at lower cost and greater efficiency than obtaining an equivalent service from the local telephone company.

Metropolitan Area Networks

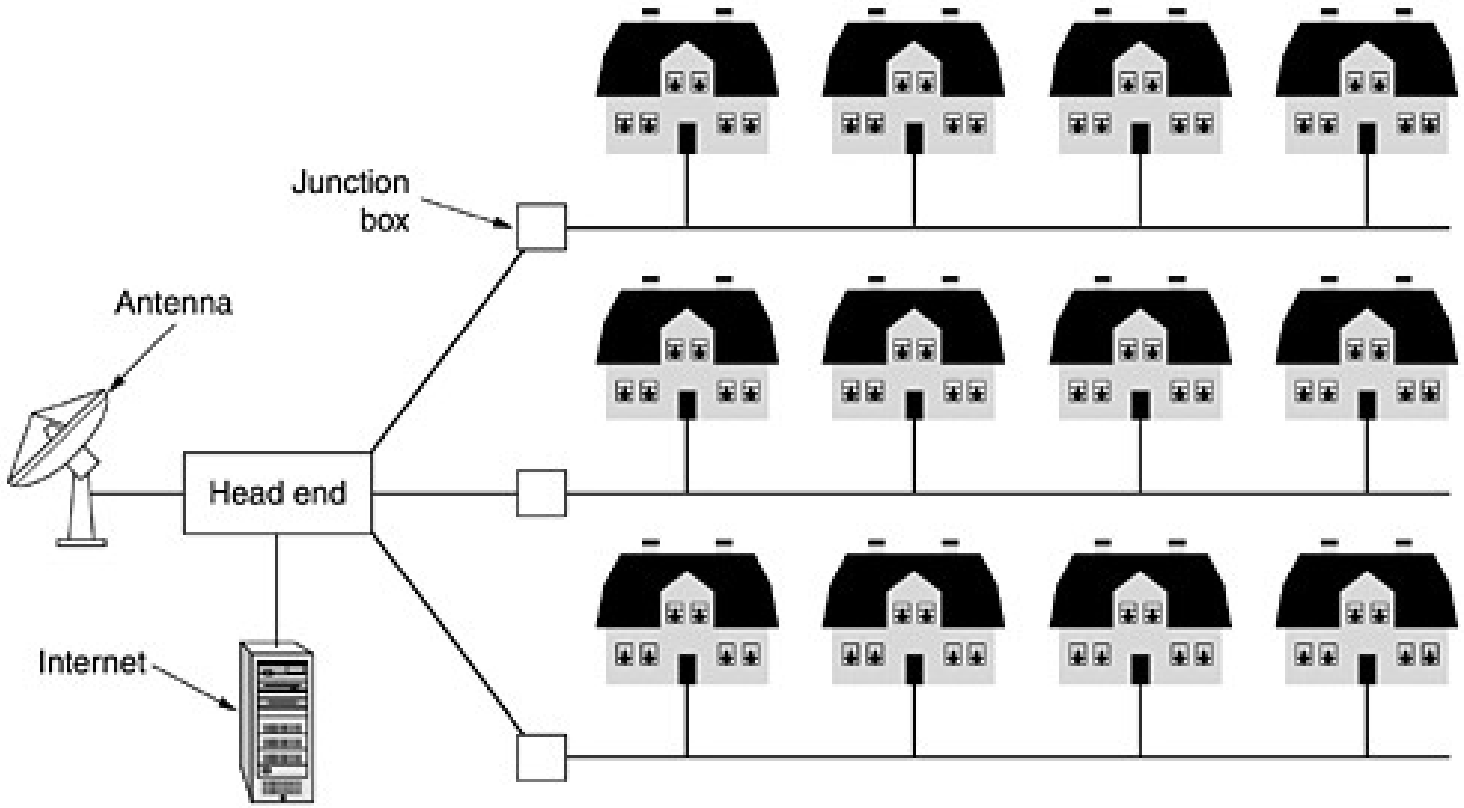


Figure: A metropolitan area network based on cable TV

Interprocessor distance	Processors located in same	Example
1 m	Square meter	Personal area network
10 m	Room	Local area network
100 m	Building	
1 km	Campus	
10 km	City	Metropolitan area network
100 km	Country	Wide area network
1000 km	Continent	
10,000 km	Planet	The Internet

Figure: Classification of interconnected processors by scale

UNIT 1 PART 2

Protocol Architecture: The Need for a Protocol Architecture, A Simple Protocol Architecture, OSI, The TCP/IP Protocol Architecture.

The Need for a Protocol Architecture

- When computers, terminals, and/or other data processing devices exchange data, the procedures involved can be quite complex.
- Consider, for example, the transfer of a file between two computers. There must be a data path between the two computers, either directly or via a communication network. But more is needed.

The Need for a Protocol Architecture

- Typical tasks to be performed are as follow:

1. **The source system must either activate the direct data communication path or inform the communication network of the identity of the desired destination system.**
2. **The source system must ascertain that the destination system is prepared to receive data.**
3. **The file transfer application on the source system must ascertain that the file management program on the destination system is prepared to accept and store the file for this particular user.**
4. **If the file formats used on the two systems are different, one or the other system must perform a format translation function.**

The Need for a Protocol Architecture

- It is clear that there must be a high degree of cooperation between the two computer systems.
- Instead of implementing the logic for this as a single module, the task is broken up into subtasks, each of which is implemented separately.
- In a protocol architecture, the modules are arranged in a vertical stack.
- Each layer in the stack performs a related subset of the functions required to communicate with another system.
- It relies on the next lower layer to perform more primitive functions and to conceal the details of those functions.
- It provides services to the next higher layer.
- Ideally, layers should be defined so that changes in one layer do not require changes in other layers.

The Need for a Protocol Architecture

- It takes two to communicate, so the same set of layered functions must exist in two systems.
- Communication is achieved by having the corresponding, or **peer**, layers in two systems communicate.
- The peer layers communicate by means of formatted blocks of data that obey a set of rules or conventions known as a **protocol**.
- The key features of a protocol are as follows:
  - **Syntax:** Concerns the format of the data blocks and signal levels
  - **Semantics:** Includes control information for coordination and error handling
  - **Timing:** Includes speed matching and sequencing

A Simple Protocol Architecture

- Instead of a single module for performing communications, there is a structured set of modules that implements the communications function. That structure is referred to as a **protocol architecture**.
- As an example, below figure suggests the way in which a file transfer facility could be implemented.
- **Three modules are used**.
- A **file transfer module** could perform tasks 3 and 4 in the preceding list.
- The **two modules** on the two systems exchange files and commands.

A Simple Protocol Architecture

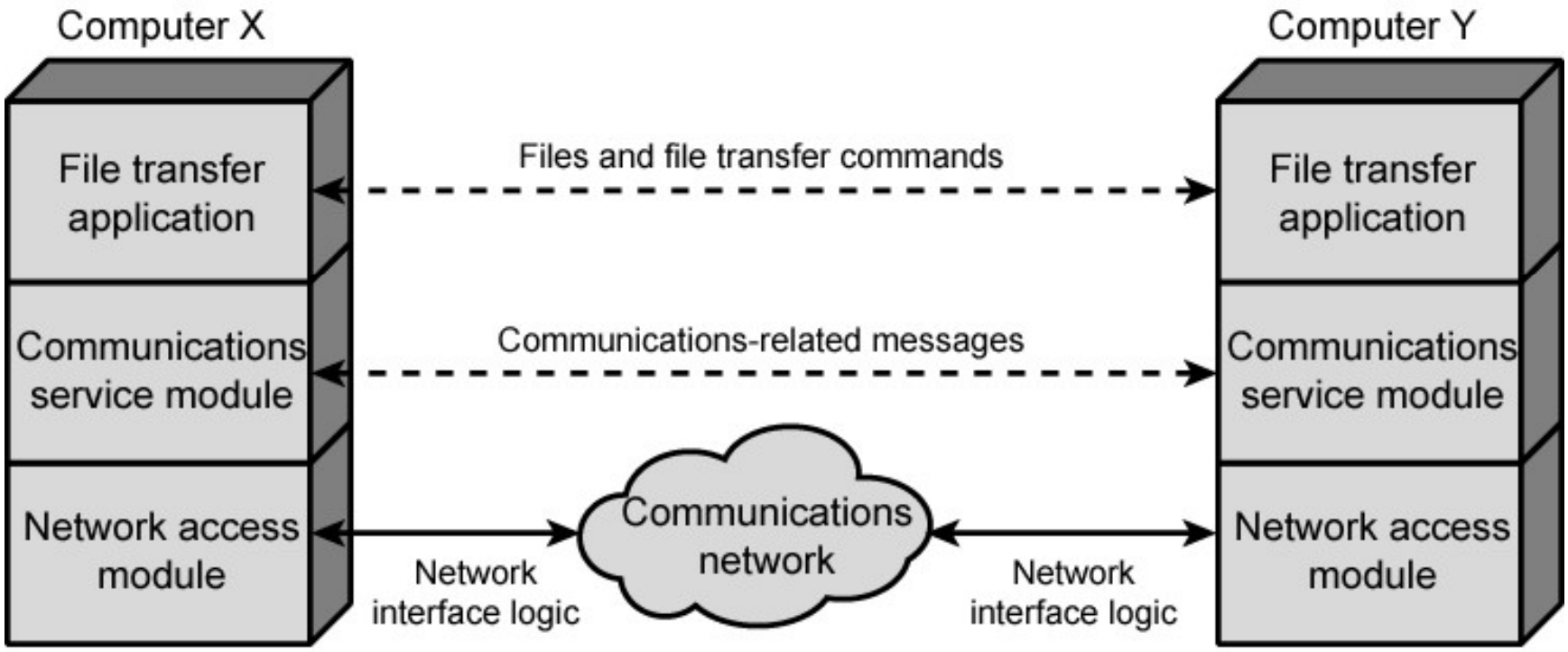


Figure: A Simplified Architecture for File Transfer

A Simple Protocol Architecture

- The **file transfer module** contains all of the logic that is unique to the file transfer application, such as transmitting passwords, file commands and file records.
- There is a need to transmit these files and commands reliably. However, the same sorts of reliability requirements are relevant to a variety of applications (e.g., electronic mail, document transfer). Therefore, a separate communications service module that can be used by a variety of applications meets these requirements.
- The **communications service module** is concerned with assuring that the two computer systems are active and ready for data transfer and for keeping track of the data that are being exchanged to assure delivery.
- However, these tasks are independent of the type of network that is being used. Therefore, the logic for actually dealing with the network is placed in a separate **network access module**. That way, if the network to be used is changed, only the network access module is affected.

A Three Layer Model

- In general terms, communications can be said to involve **three agents: applications, computers, and networks.**
- **Applications** execute on computers that typically support multiple simultaneous applications.
- **Computers** are connected to networks, and the data to be exchanged are transferred by the **network** from one computer to another.
- Thus, the transfer of data from one application to another involves first getting the data to the computer in which the application resides and then getting it to the intended application within the computer.
- With these concepts in mind, it appears natural to organize the communication task into three relatively independent layers:
  - Network Access Layer
  - Transport Layer
  - Application Layer

Network Access Layer

- The **network access layer** is concerned with the exchange of data between a computer and the network to which it is attached.
- The sending computer must provide the network with the address of the destination computer, so that the network may route the data to the appropriate destination.
- The sending computer may wish to invoke certain services, such as priority, that might be provided by the network.
- The specific software used at this layer depends on the type of network to be used; different standards have been developed for circuit switching, packet switching, LANs, and others.

Transport Layer

- Regardless of the nature of the applications that are exchanging data, there is usually a requirement that data be exchanged reliably.
- That is, we would like to be assured that all of the data arrive at the destination application and that the data arrive in the same order in which they were sent.
- As we shall see, the mechanisms for providing reliability are essentially independent of the nature of the applications.
- Thus, it makes sense to collect those mechanisms in a common layer shared by all applications; this is referred to as the **transport layer**.

Application Layer

- The **application layer** contains the logic needed to support the various user applications.
- For each different type of application, such as file transfer, a separate module is needed that is peculiar to that application.

A Three Layer Model

- Below figure shows three computers connected to a network.
- Each computer contains software at the network access and transport layers and software at the application layer for one or more applications.
- For successful communication, every entity in the overall system must have a unique address.

A Three Layer Model

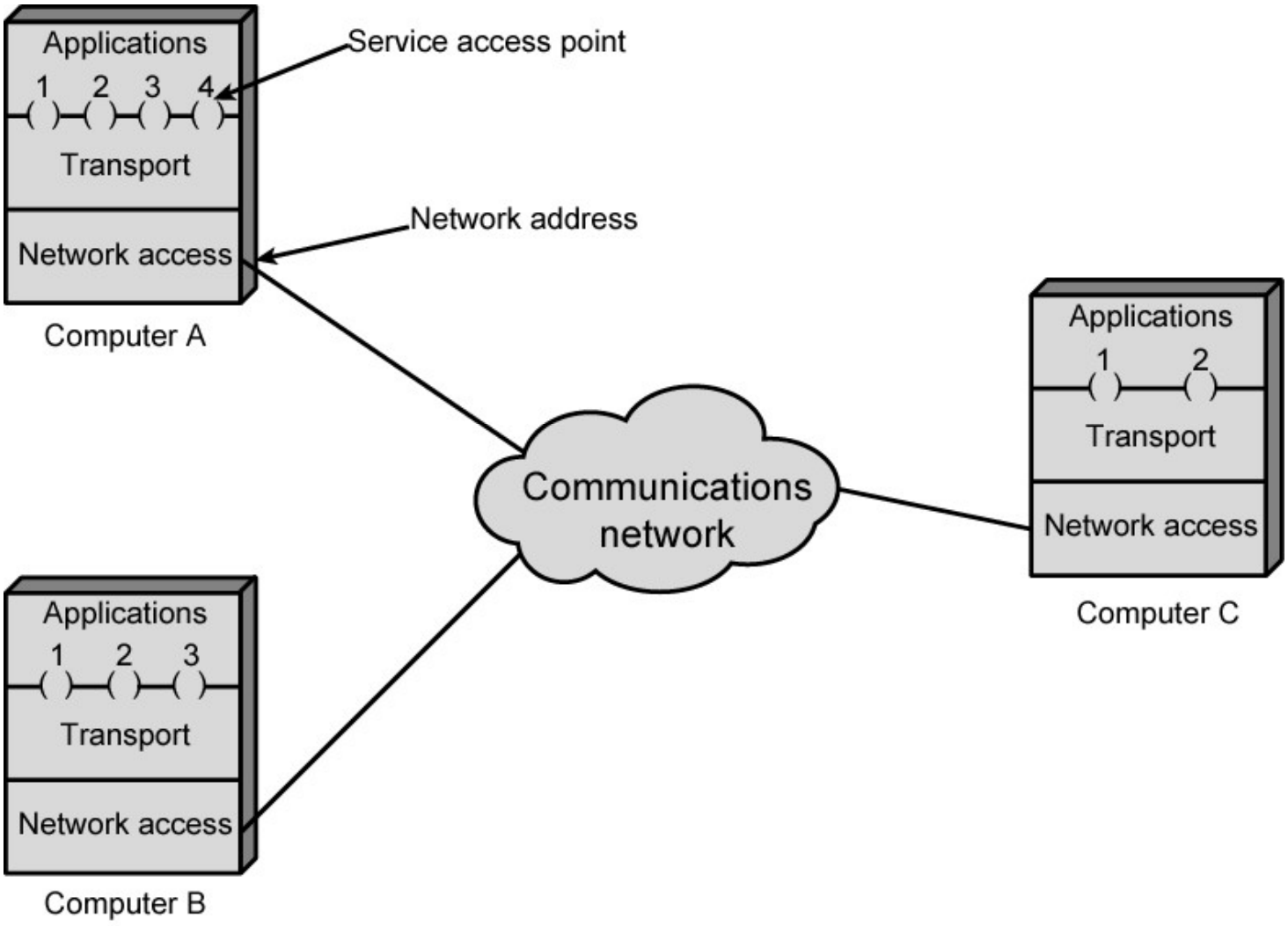


Figure: Protocol Architectures and Networks

Addressing Requirements

- Actually, two levels of addressing are needed.
- Each computer on the network must have a unique network address; this allows the network to deliver data to the proper computer.
- Each application on a computer must have an address that is unique within that computer; this allows the transport layer to support multiple applications at each computer.
- These latter addresses are known as **service access points (SAPs), or ports**, connoting the fact that each application is individually accessing the services of the transport layer.

Protocols in Simplified Architecture

- Below figure indicates that modules at the same level on different computers communicate with each other by means of a protocol.
- Let us trace a simple operation - Suppose that an application, associated with SAP 1 at computer X, wishes to send a message to another application, associated with SAP 2 at computer Y.
- The application at X hands the message over to its transport layer with instructions to send it to SAP 2 on computer Y.
- The transport layer hands the message over to the network access layer, which instructs the network to send the message to computer Y.
- Note that the network need not be told the identity of the destination service access point. All that it needs to know is that the data are intended for computer Y.

A Three Layer Model

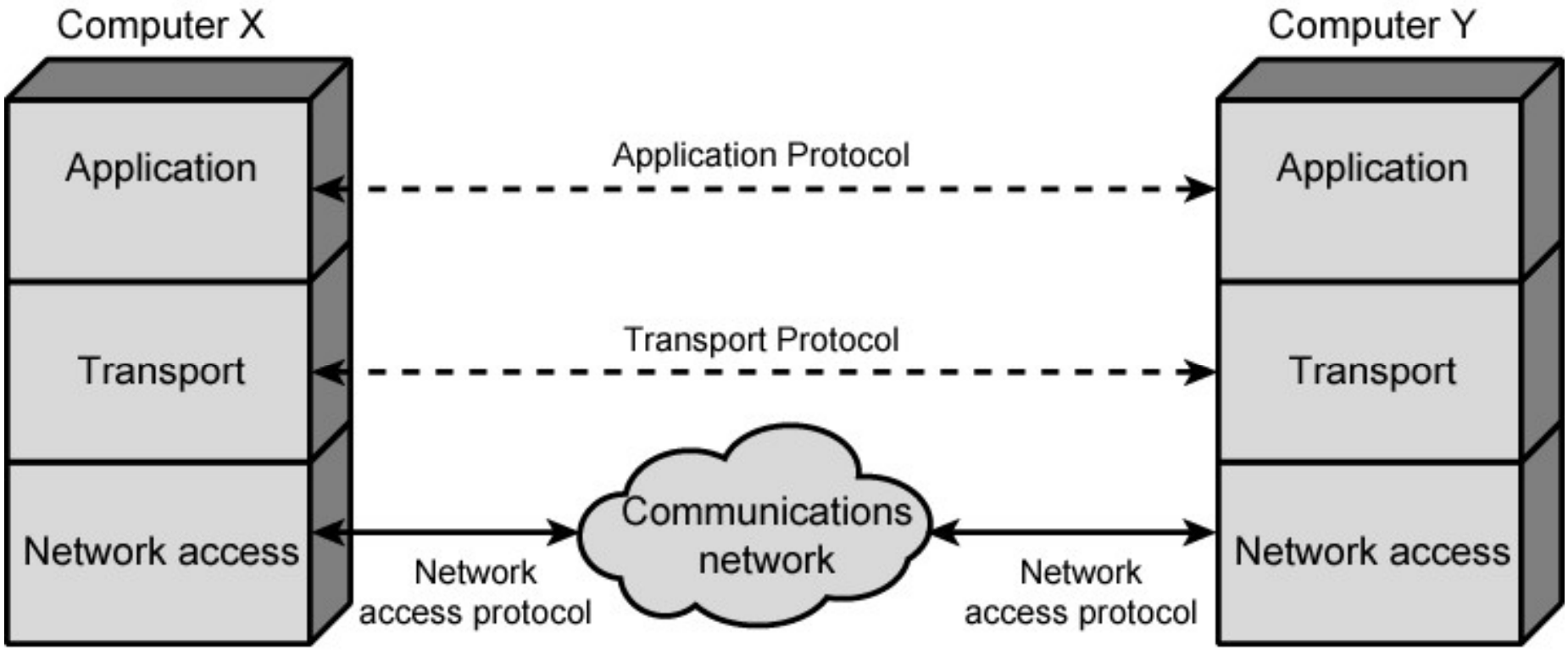
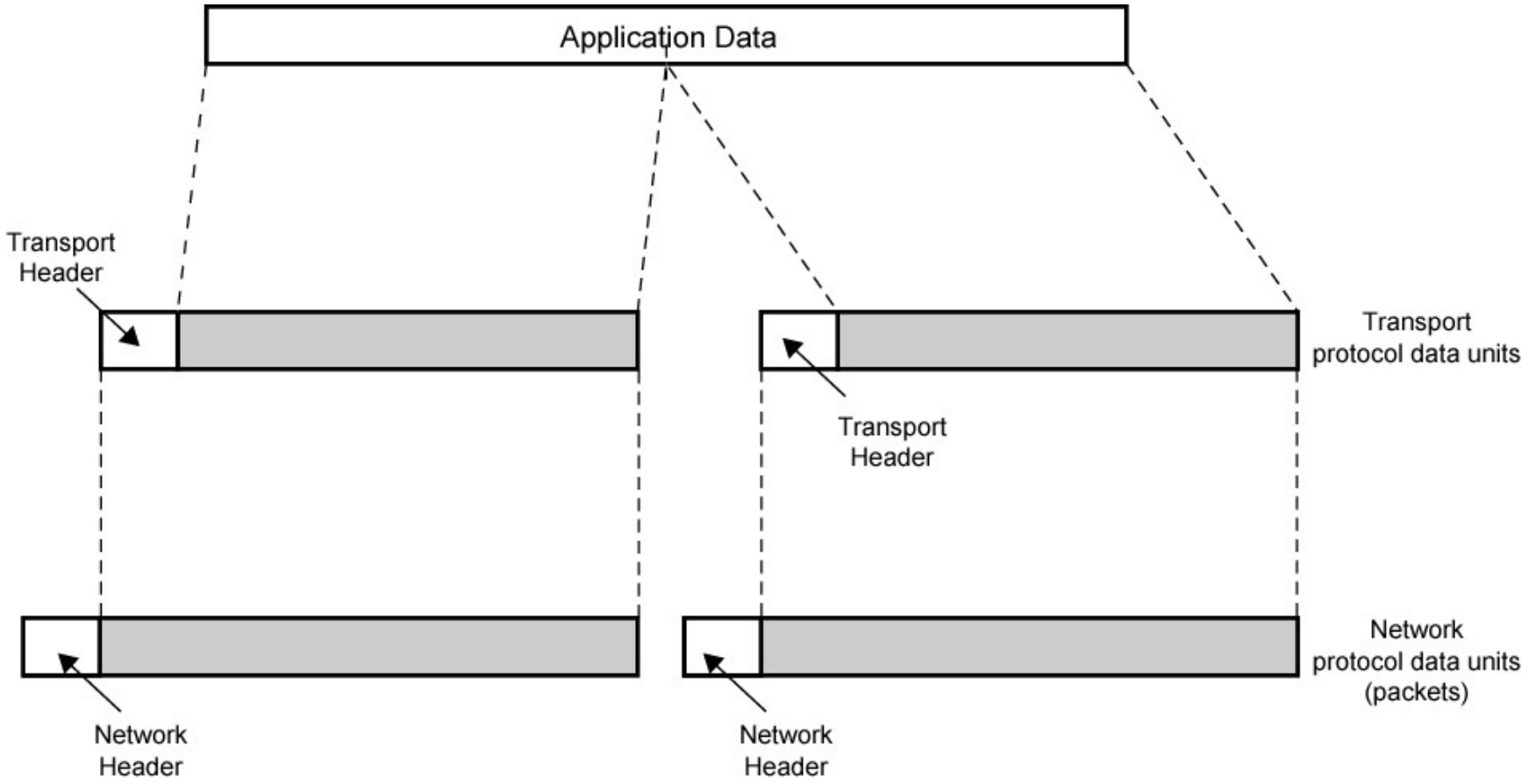


Figure: Protocols in a Simplified Architecture

Protocol Data Units (PDU)

- To control this operation, control information, as well as user data, must be transmitted, as suggested in below figure.
- Let us say that the sending application generates a block of data and passes this to the transport layer.
- The transport layer may break this block into two smaller pieces to make it more manageable.
- To each of these pieces the transport layer appends a transport header, containing protocol control information.
- The combination of data from the next higher layer and control information is known as a protocol data unit (PDU); in this case, it is referred to as a transport PDU.
- The header in each transport PDU contains control information to be used by the peer transport protocol at computer B.

Protocol Data Units (PDU)



Protocol Data Units (PDU)

- Examples of items that may be stored in this header include the following:
  - **Destination SAP**: When the destination transport layer receives the transport PDU, it must know to whom the data are to be delivered.
  - **Sequence number**: Because the transport protocol is sending a sequence of PDUs, it numbers them sequentially so that if they arrive out of order, the destination transport entity may reorder them.
  - **Error-detection code**: The sending transport entity may include a code that is a function of the contents of the remainder of the PDU. The receiving transport protocol performs the same calculation and compares the result with the incoming code. A discrepancy results if there has been some error in transmission. In that case, the receiver can discard the PDU and take corrective action.

Network PDU

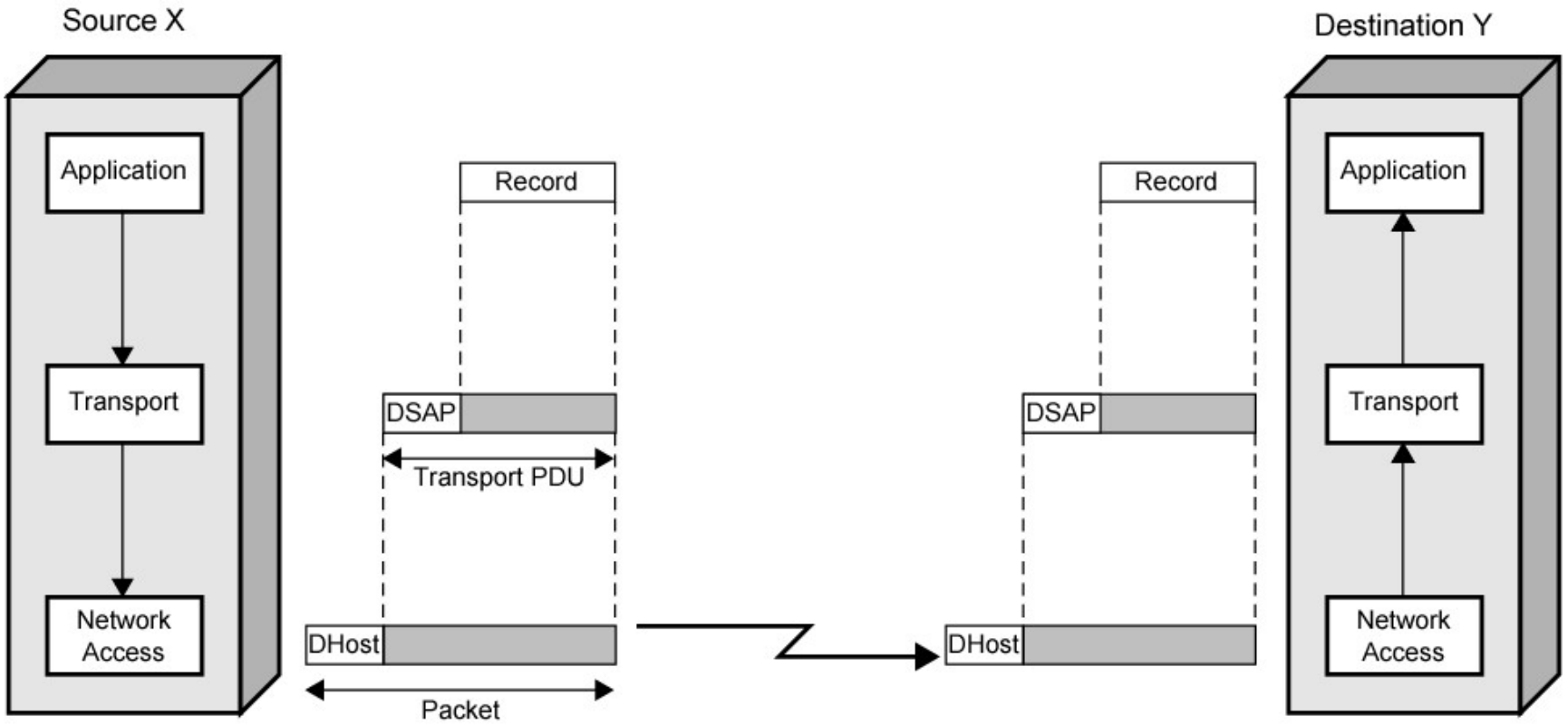
- The next step is for the transport layer to hand each PDU over to the network layer, with instructions to transmit it to the destination computer.
- To satisfy this request, the network access protocol must present the data to the network with a request for transmission.
- As before, this operation requires the use of control information. In this case, the network access protocol appends a network access header to the data it receives from the transport layer, creating a network access PDU.

Network PDU

- Examples of the items that may be stored in the header include the following:
  - **Destination computer address**: The network must know to whom (which computer on the network) the data are to be delivered.
  - **Facilities requests**: The network access protocol might want the network to make use of certain facilities, such as priority.

Operation of a Protocol Architecture

Below figure puts all of these concepts together, showing the interaction between modules to transfer one block of data.



DSAP = destination service access point
DHost = destination host

Standardized Protocol Architectures

- When communication is desired among computers from different vendors, the software development effort can be a nightmare.
- Different vendors use different data formats and data exchange protocols.
- Even within one vendor's product line, different model computers may communicate in unique ways.

Standardized Protocol Architectures

- As the use of computer communications and computer networking proliferates, a one-at-a-time special-purpose approach to communications software development is too costly to be acceptable.
- The only alternative is for computer vendors to adopt and implement a common set of conventions.
- For this to happen, standards are needed. Such standards would have two benefits:

Standardized Protocol Architectures

1. Vendors feel encouraged to implement the standards because of an expectation that, because of wide usage of the standards, their products would be less marketable without them.
2. Customers are in a position to require that any vendor wishing to propose equipment to them implement the standards.

Standardized Protocol Architectures

- Two protocol architectures have served as the basis for the development of interoperable protocol standards: the TCP/IP protocol suite and the OSI reference model.
- TCP/IP is by far the most widely used interoperable architecture.
- OSI, through well known, has never lived up to its early promise. There is also a widely used proprietary scheme: IBM's System Network Architecture (SNA).

Standardized Protocol Architectures

(Summary)

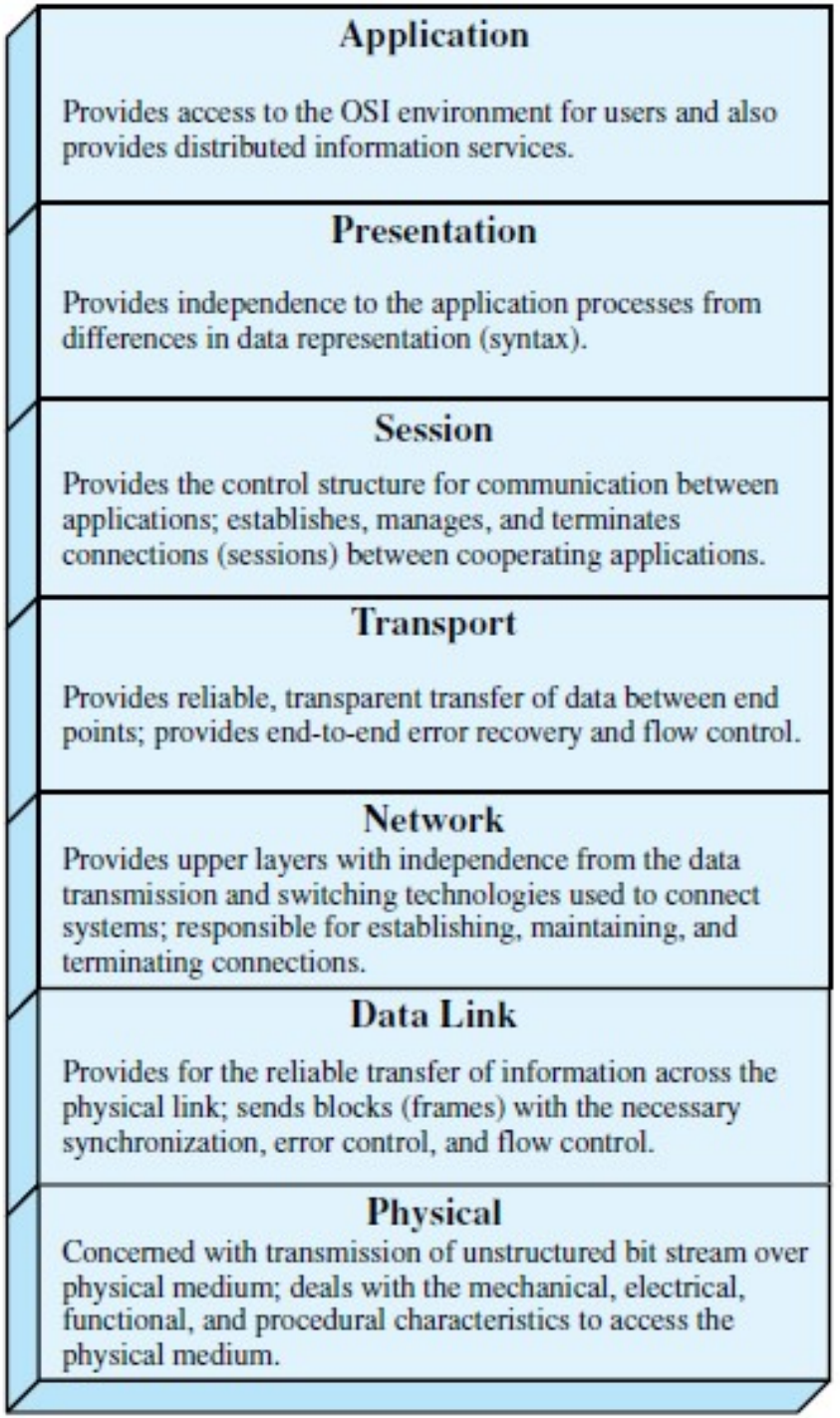
- Required for devices to communicate
- Vendors have more marketable products
- Customers can insist on standards based equipment
- Two standards:
  - OSI Reference model
    - Never lived up to early promises
  - TCP/IP protocol suite
    - Most widely used
- Also: IBM Systems Network Architecture (SNA)

The OSI Model

- The Open Systems Interconnection (OSI) reference model was developed by the International Organization for Standardization (ISO) as a model for a computer protocol architecture and as a framework for developing protocol standards.
- The OSI model consists of seven layers:
  - Application
  - Presentation
  - Session
  - Transport
  - Network
  - Data link
  - Physical

OSI Layers

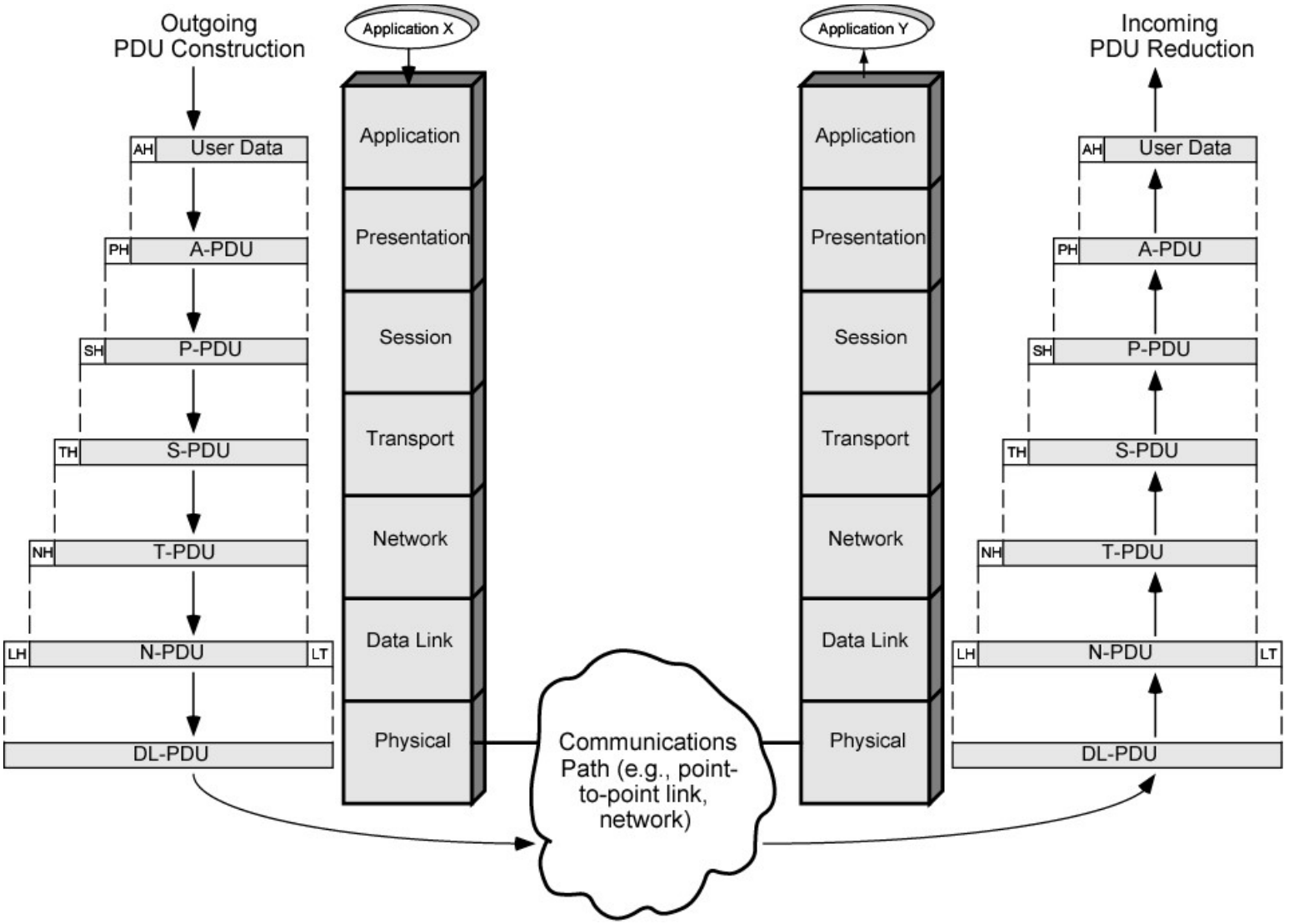
- Below figure illustrates the OSI model and provides a brief definition of the functions performed at each layer.
- The intent of the OSI model is that protocols be developed to perform the functions of each layer.



The OSI Model

- A layer model
- Each layer performs a subset of the required communication functions
- Each layer relies on the next lower layer to perform more primitive functions
- Each layer provides services to the next higher layer
- Changes in one layer should not require changes in other layers

The OSI Environment



Standardization within the OSI Framework

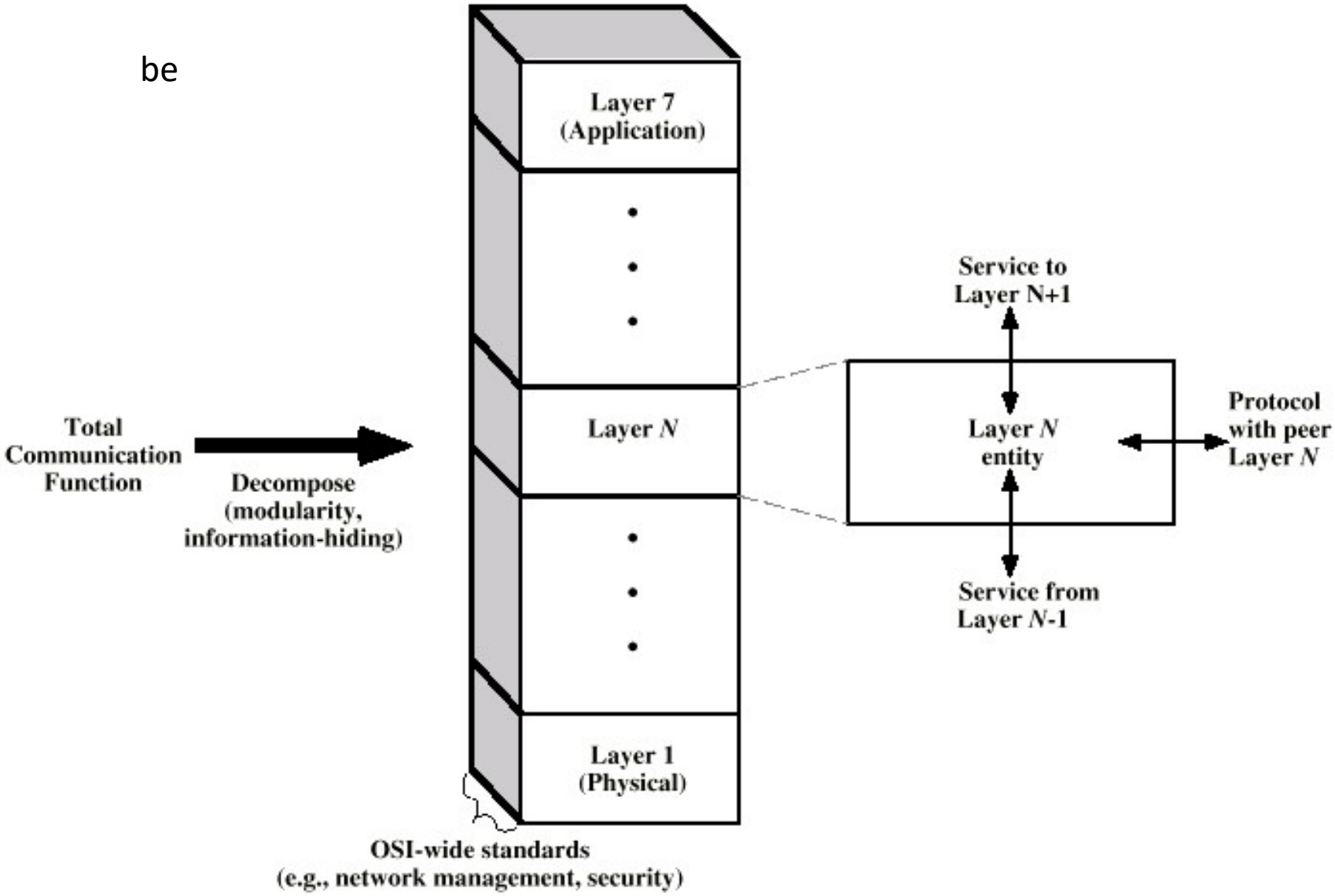
- The principal motivation for the development of the OSI model was to provide a framework for standardization. Within the model, one or more protocol standards can be developed at each layer.
- The model defines in general terms the functions to be performed at that layer and facilitates the standards-making process in two ways:
  - Because the functions of each layer are well defined, standards can be developed independently and simultaneously for each layer. This speeds up the standards-making process.
  - Because the boundaries between layers are well defined, changes in standards in one layer need not affect already existing software in another layer. This makes it easier to introduce new standards.

Standardization within the OSI Framework

- Below figure illustrates the use of the OSI model as such a framework.
- The overall communications function is decomposed into seven distinct layers. i.e., the overall function is broken up into a number of modules, making the interfaces between modules as simple as possible.
- In addition, the design principle of information hiding is used: Lower layers are concerned with greater levels of detail; upper layers are independent of these details.
- Each layer provides services to the next higher layer and implements a protocol to the peer layer in other systems.

OSI architecture as a Framework for Standardization

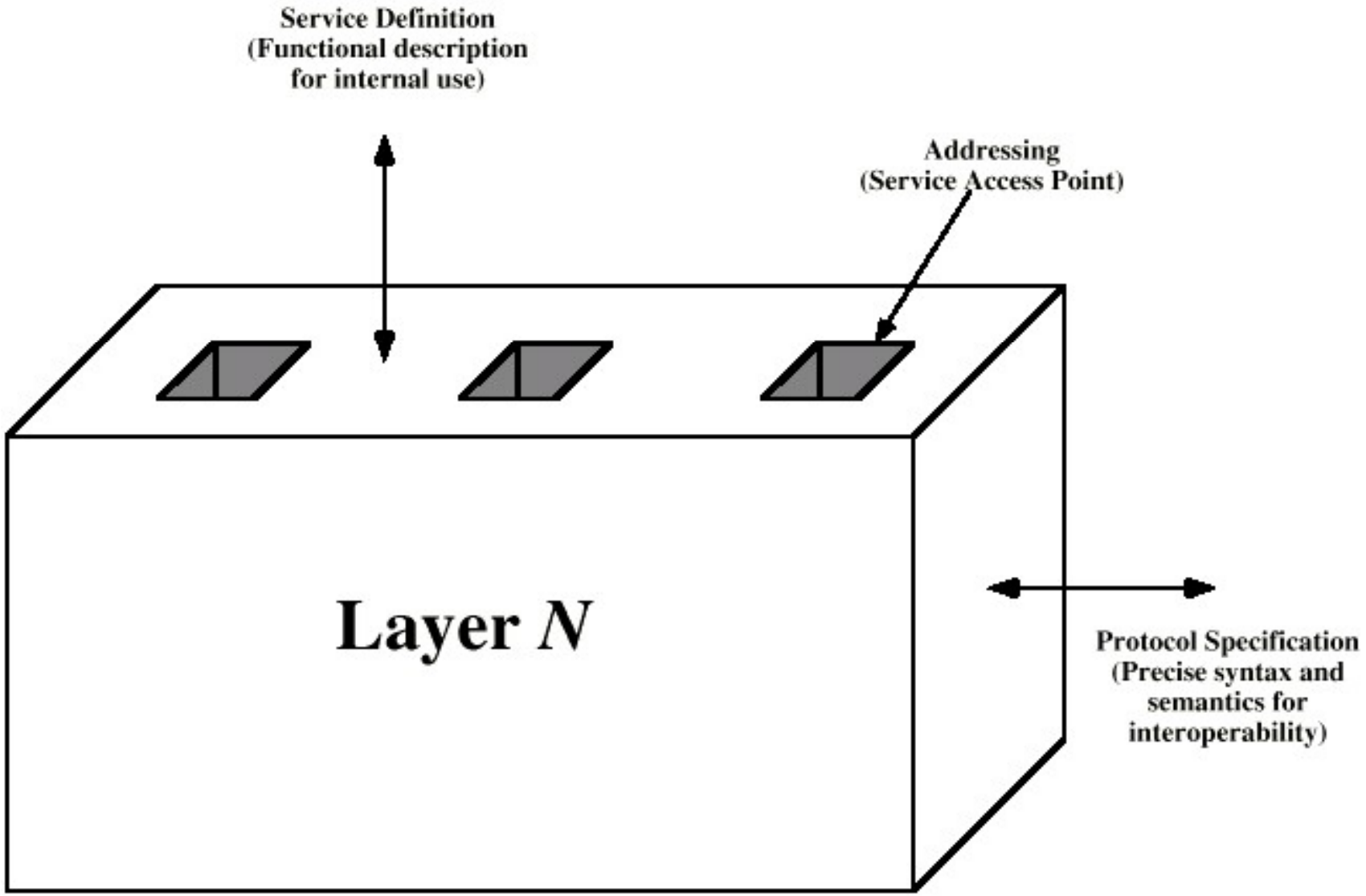
be



Standardization within the OSI Framework

- Below figure shows more specifically the nature of the standardization required at each layer. Three elements are key:
- **Protocol specification**: Two entities at the same layer in different systems cooperate and interact by means of a protocol. Because two different open systems are involved, the protocol must be specified precisely. This includes the format of the protocol data units exchanged, the semantics of all fields, and the allowable sequence of PDUs.
- **Service definition**: In addition to the protocol or protocols that operate at a given layer, standards are needed for the services that each layer provides to the next higher layer. Typically, the definition of services is equivalent to a functional description that defines what services are provided, but not how the services are to be provided.
- **Addressing**: Each layer provides services to entities at the next higher layer. These entities are referenced by means of a service access point (SAP). Thus, a network service access point (NSAP) indicates a transport entity that is a user of the network service.

Layer Specific Standards



Service Primitives and Parameters

- The services between adjacent layers in the OSI architecture are expressed in terms of primitives and parameters.
- A primitive specifies the function to be performed, and the parameters are used to pass data and control information.
- The actual form of a primitive is implementation dependent. An example is a procedure call.

Service Primitives and Parameters

- Four types of primitives are used in standards to define the interaction between adjacent layers in the architecture.
- These are defined in below Table.
- The layout of below figure(a) suggests the time ordering of these events. For example, consider the transfer of data from an (N) entity to a peer (N) entity in another system.

Primitive Types

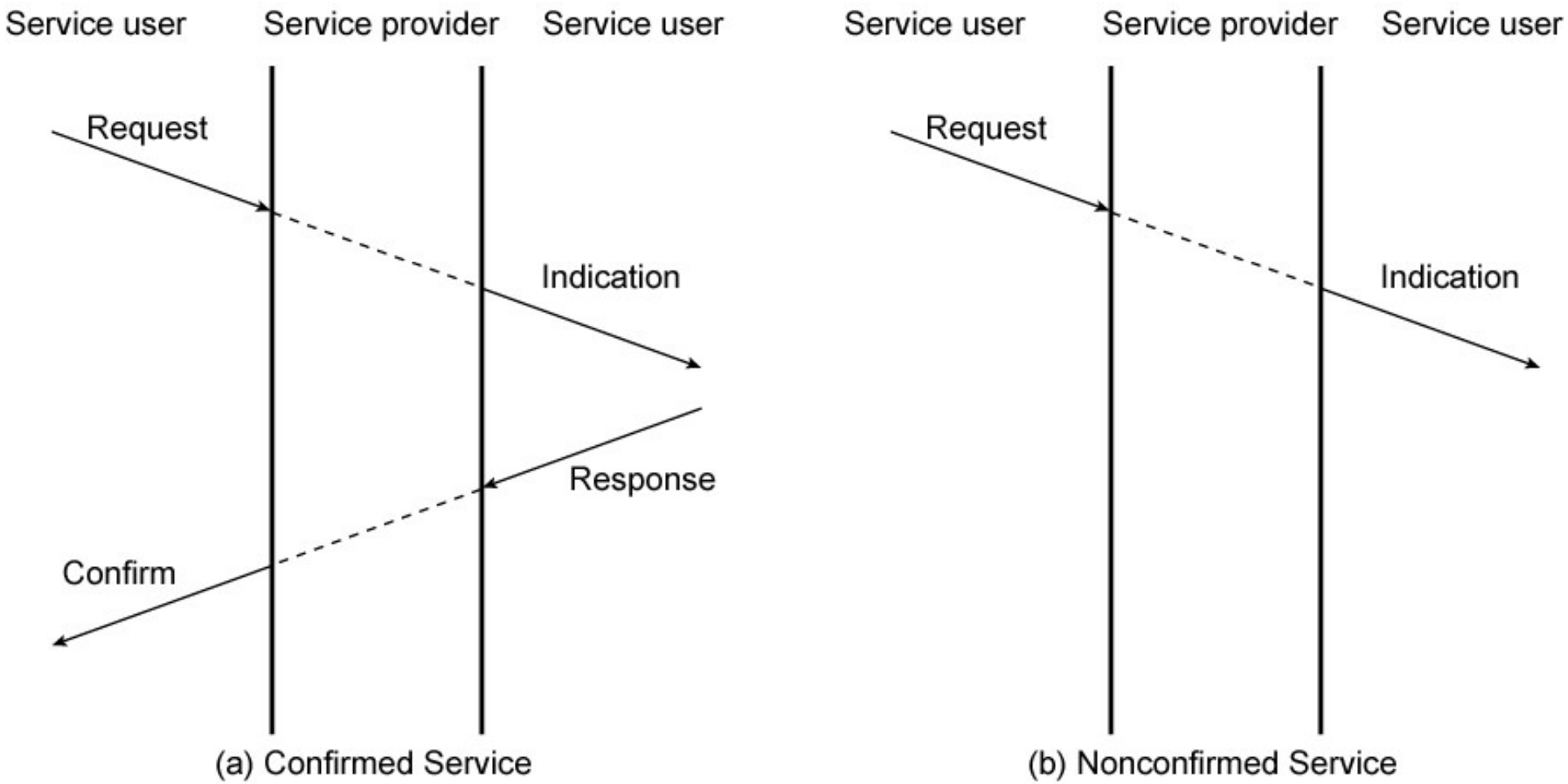
REQUEST	A primitive issued by a service user to invoke some service and to pass the parameters needed to specify fully the requested service
INDICATION	A primitive issued by a service provider either to: 1. indicate that a procedure has been invoked by the peer service user on the connection and to provide the associated parameters, or 2. notify the service user of a provider-initiated action
RESPONSE	A primitive issued by a service user to acknowledge or complete some procedure previously invoked by an indication to that user
CONFIRM	A primitive issued by a service provider to acknowledge or complete some procedure previously invoked by a request by the service user

Timing Sequence for Service Primitives

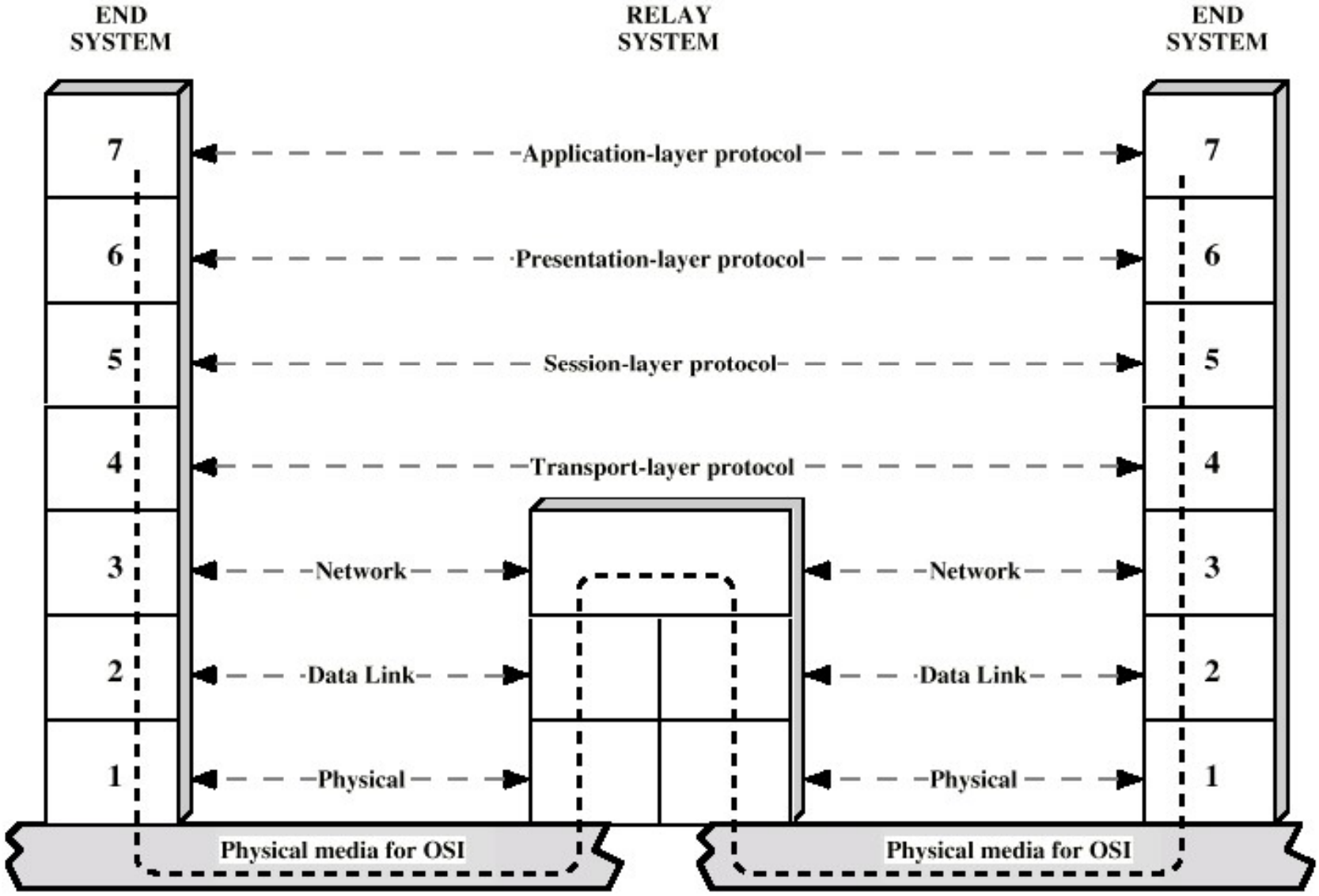
- The following steps occur:

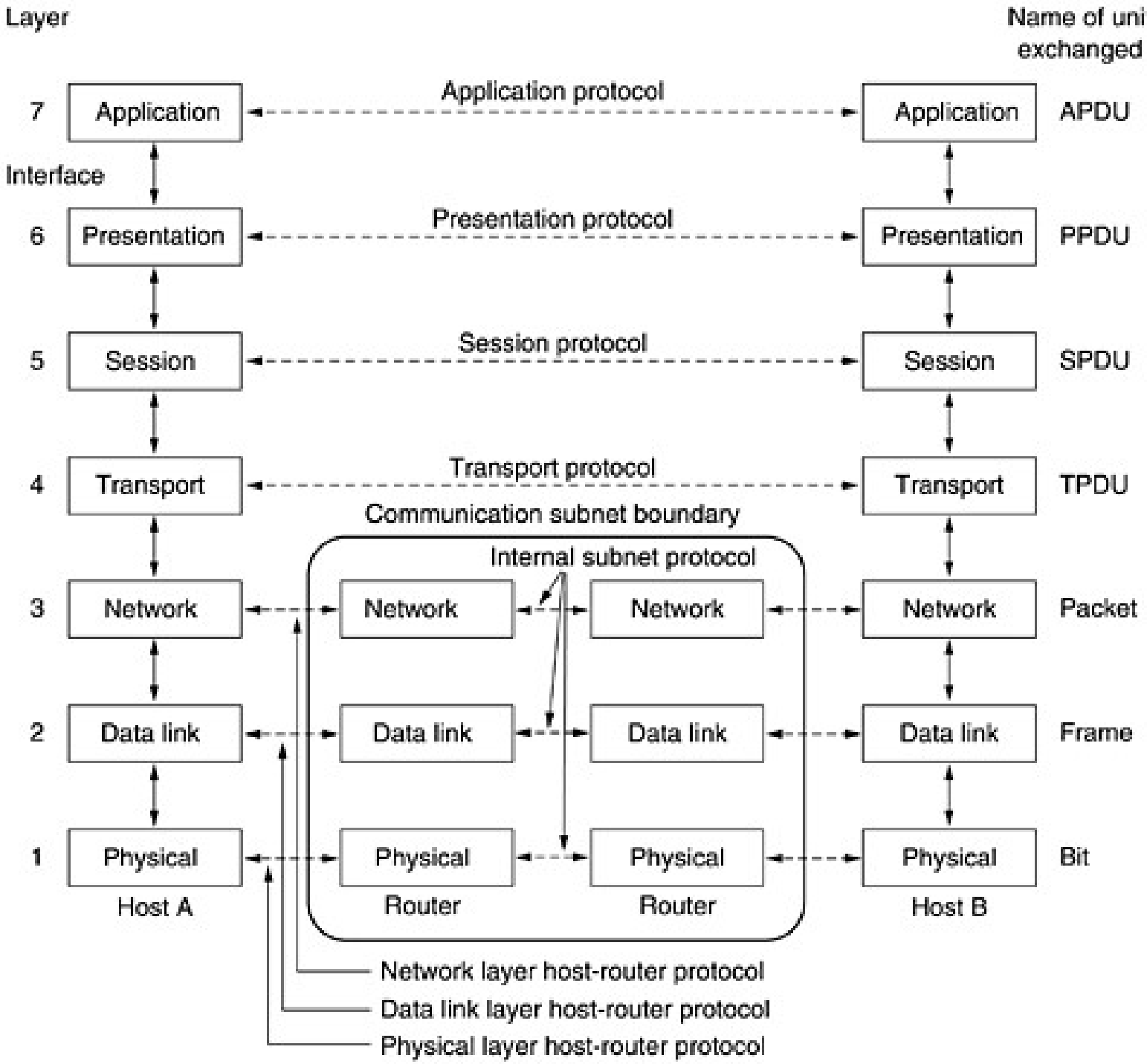
1. The source (N) entity invokes its (N – 1) entity with a request primitive. Associated with the primitive are the parameters needed, such as the data to be transmitted and the destination address.
2. The source (N – 1) entity prepares an (N – 1) PDU to be sent to its peer (N – 1) entity.
3. The destination (N– 1) entity delivers the data to the appropriate destination (N) entity via an indication primitive, which includes the data and source address as parameters.
4. If an acknowledgment is called for, the destination (N) entity issues a response primitive to its (N – 1) entity.
5. The (N – 1) entity conveys the acknowledgment in an (N – 1) PDU.
6. The acknowledgment is delivered to the (N) entity as a confirm primitive.

Timing Sequence for Service Primitives



The OSI Layers




Layer
Name of unit exchanged
7
Application
Application protocol
Application
APDU
Interface
6
Presentation
Presentation protocol
Presentation
PPDU
5
Session
Session protocol
Session
SPDU
4
Transport
Transport protocol
Transport
TPDU
Communication subnet boundary
Internal subnet protocol
3
Network
Network
Network
Network
Packet
2
Data link
Data link
Data link
Data link
Frame
1
Physical
Physical
Physical
Physical
Bit
Host A
Router
Router
Host B
Network layer host-router protocol
Data link layer host-router protocol
Physical layer host-router protocol

Physical Layer

- The physical layer covers the physical interface between devices and the rules by which bits are passed from one to another. The physical layer has four important characteristics:
  - **Mechanical**: Relates to the physical properties of the interface to a transmission medium. Typically, the specification is of a pluggable connector that joins one or more signal conductors, called circuits.
  - **Electrical**: Relates to the representation of bits (e.g., in terms of voltage levels) and the data transmission rate of bits.
  - **Functional**: Specifies the functions performed by individual circuits of the physical interface between a system and the transmission medium.
  - **Procedural**: Specifies the sequence of events by which bit streams are exchanged across the physical medium.

Data Link Layer

- The physical layer provides only a raw bit stream service, the data link layer attempts to make the physical link reliable and provides the means to activate, maintain, and deactivate the link.
- The principal service provided by the data link layer to higher layers is that of **error detection and control**.
- Thus, with a fully functional data link layer protocol, the next higher layer may assume error-free transmission over the link.
- However, if communication is between two systems that are not directly connected, the connection will comprise a number of data links in tandem, each functioning independently.
- Thus, the higher layers are not relieved of an error control responsibility.

Network Layer

- The network layer provides for the transfer of information between end systems across some sort of communications network.
- It relieves higher layers of the need to know anything about the underlying data transmission and switching technologies used to connect Systems.
- At this layer, the computer system engages in a dialogue with the network to specify the destination address and to request certain network facilities, such as priority.

Transport Layer

- The transport layer provides a mechanism for the exchange of data between end systems. The connection-oriented transport service ensures that data are delivered error free, in sequence, with no losses or duplications.
- The transport layer may also be concerned with optimizing the use of network services and providing a requested quality of service to session entities.
- For example, the session entity may specify acceptable error rates, maximum delay, priority, and security.
- The size and complexity of a transport protocol depend on how reliable or un-reliable the underlying network and network layer services are.
- Accordingly, ISO has developed a family of five transport protocol standards, each oriented toward a different underlying service.
- In the TCP/IP protocol-suite, there are two common transport-layer protocols: the connection oriented TCP (Transmission Control Protocol) and the connectionless UDP (User Datagram Protocol).

Session Layer

- The session layer provides the mechanism for controlling the dialogue between applications in end systems.
- In many cases, there will be little or no need for session-layer services, but for some applications, such services are used.
- The key services provided by the session layer include the following:
- **Dialogue discipline**: This can be two-way simultaneous (full duplex) or two-way alternate (half duplex).
- **Grouping:** The flow of data can be marked to define groups of data. For example, if a retail store is transmitting sales data to a regional office, the data can be marked to indicate the end of the sales data for each department. This would signal the host computer to finalize running totals for that department and start new running counts for the next department.
- **Recovery:** The session layer can provide a checkpointing mechanism, so that if a failure of some sort occurs between checkpoints, the session entity can retransmit all data since the last checkpoint.

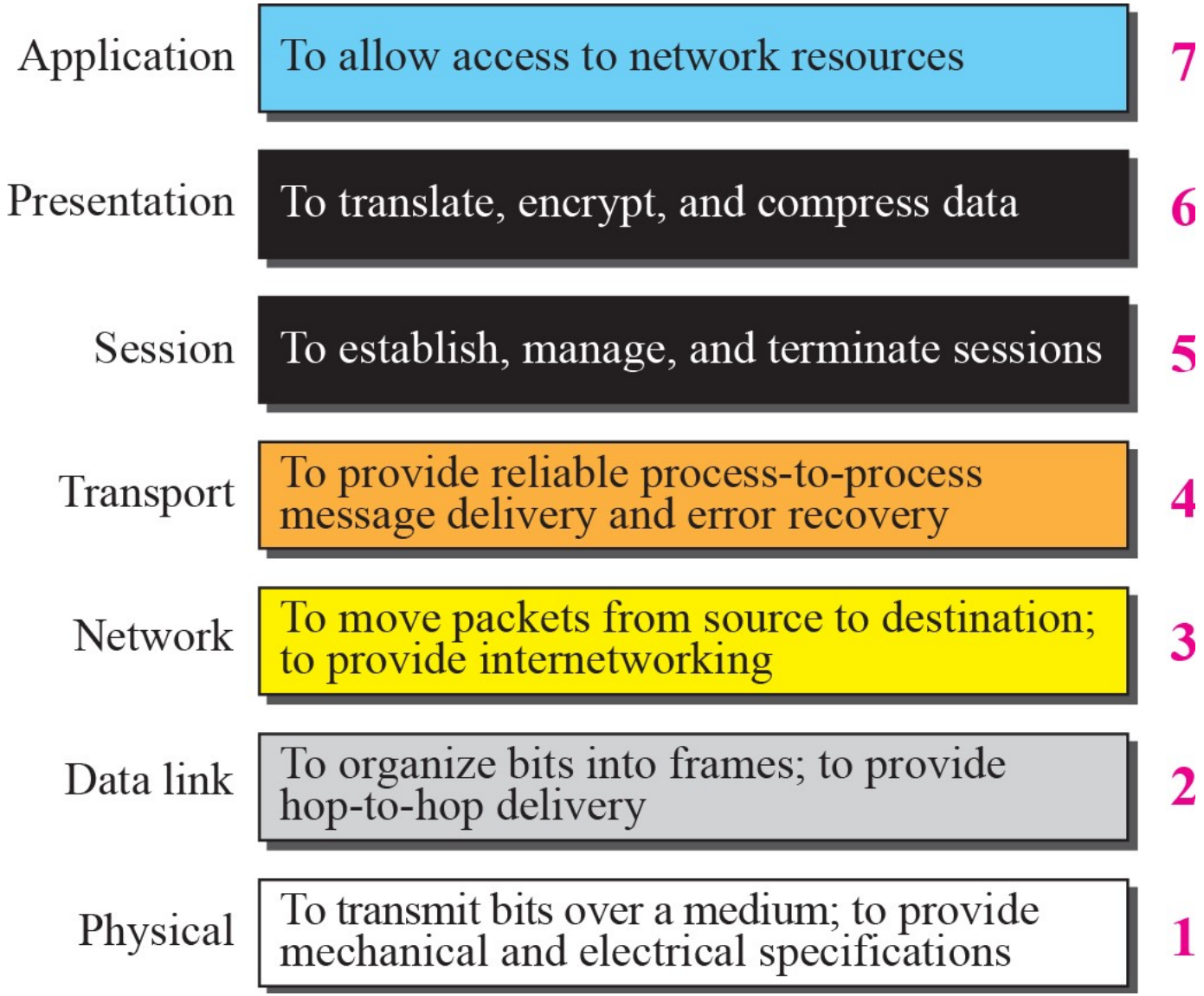
Presentation Layer

- The presentation layer defines the format of the data to be exchanged between applications and offers application programs a set of data transformation services.
- The presentation layer defines the syntax used between application entities and provides for the selection and subsequent modification of the representation used.
- Examples of specific services that may be performed at this layer include data compression and encryption.

Application Layer

- The application layer provides a means for application programs to access the OSI environment.
- This layer contains management functions and generally useful mechanisms to support distributed applications.
- In addition, general-purpose applications such as file transfer, electronic mail, and terminal access to remote computers are considered to reside at this layer.

Summary of OSI Layers



TCP/IP Protocol Architecture

- The TCP/IP protocol architecture is a result of protocol research and development conducted on the experimental packet-switched network, ARPANET, funded by the Defense Advanced Research Projects Agency (DARPA), and is generally referred to as the TCP/IP protocol suite.
- This protocol suite consists of a large collection of protocols that have been issued as Internet standards by the Internet Activities Board (IAB).

The TCP/IP Layers

- The TCP/IP model organizes the communication task into five relatively independent layers.
  - Physical layer
  - Network access layer
  - Internet layer
  - Host-to-host, or transport layer
  - Application layer

Physical Layer

- The physical layer covers the physical interface between a data transmission device (e.g., workstation, computer) and a transmission medium or network.
- This layer is concerned with specifying the characteristics of the transmission medium, the nature of the signals, the data rate, and related matters.

Network Access Layer

- The network access layer is concerned with the exchange of data between an end system (server, workstation, etc.) and the network to which it is attached.
- The sending computer must provide the network with the address of the destination computer, so that the network may route the data to the appropriate destination.

Internet Layer

- The network access layer is concerned with access to and routing data across a network for two end systems attached to the same network.
- In those cases where two devices are attached to different networks, procedures are needed to allow data to traverse multiple interconnected networks.
- This is the function of the internet layer.
- The Internet Protocol (IP) is used at this layer to provide the routing function across multiple networks.
- This protocol is implemented not only in the end systems but also in routers.
- A router is a processor that connects two networks and whose primary function is to relay data from one network to the other on its route from the source to the destination end system.

Host-to-Host Layer, or Transport Layer

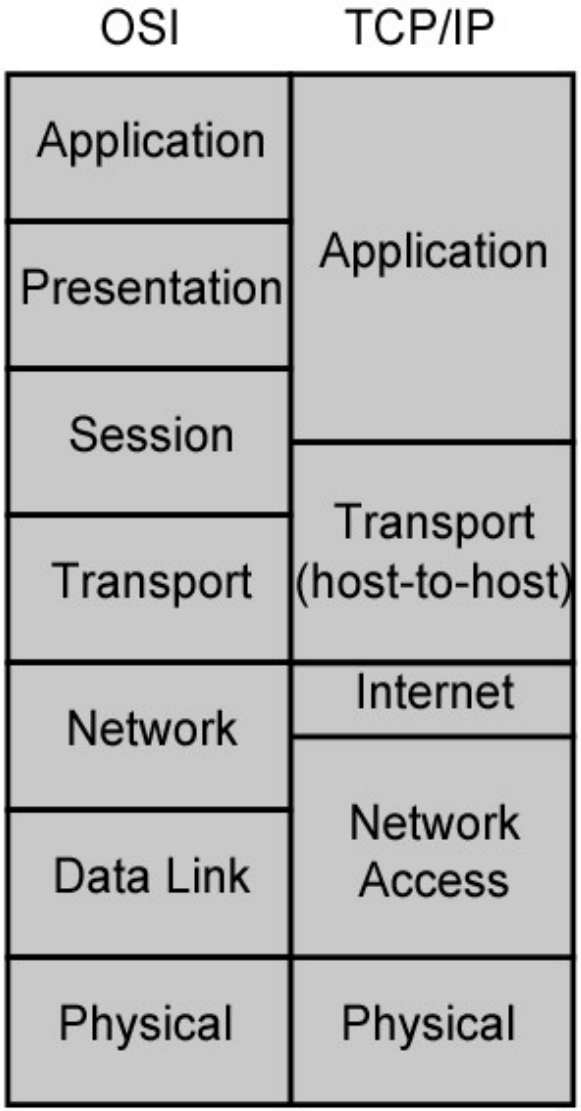
- Regardless of the nature of the applications that are exchanging data, there is usually a requirement that data be exchanged reliably.
- That is, we would like to be assured that all of the data arrive at the destination application and that the data arrive in the same order in which they were sent.
- As we shall see, the mechanisms for providing reliability are essentially independent of the nature of the applications.
- Thus, it makes sense to collect those mechanisms in a common layer shared by all applications; this is referred to as the host-to-host layer, or transport layer.
- The Transmission Control Protocol (TCP) is the most commonly used protocol to provide this functionality.

Application Layer

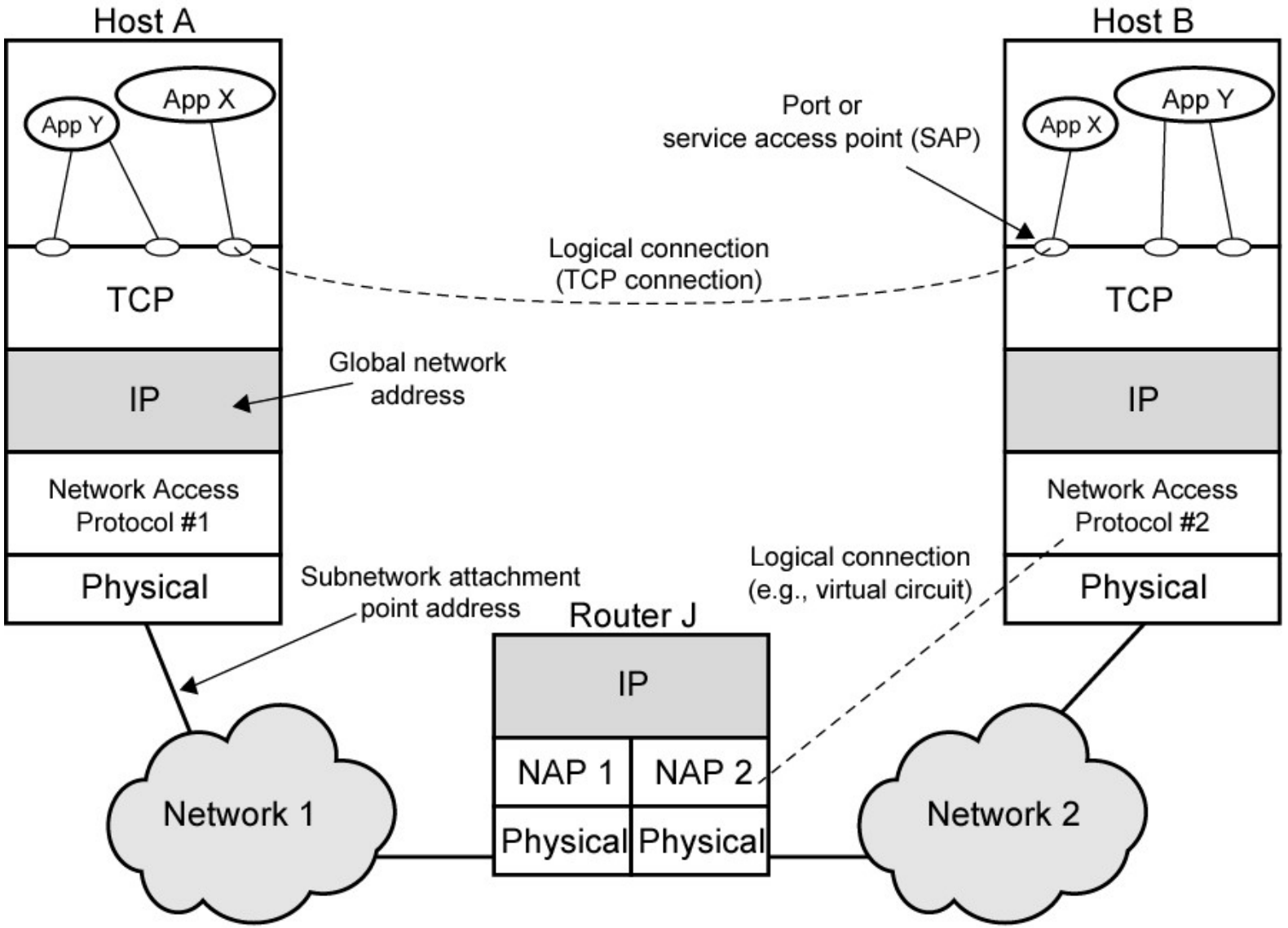
- The application layer contains the logic needed to support the various user applications.
- For each different type of application, such as file transfer, a separate module is needed that is peculiar to that application.

OSI v TCP/IP

The below figure illustrates the layers of the TCP/IP and OSI architectures, showing roughly the correspondence in functionality between the two.



TCP/IP Concepts



Addressing level

- For successful communication, every entity in the overall system must have a unique address.
- Actually, two levels of addressing are needed.
- Each host on a subnetwork must have a unique global **internet address**; this allows the data to be delivered to the proper host.
- Each process with a host must have an address that is unique within the host; this allows the host-to-host protocol (TCP) to deliver data to the proper process.
- These latter addresses are known as **ports.**

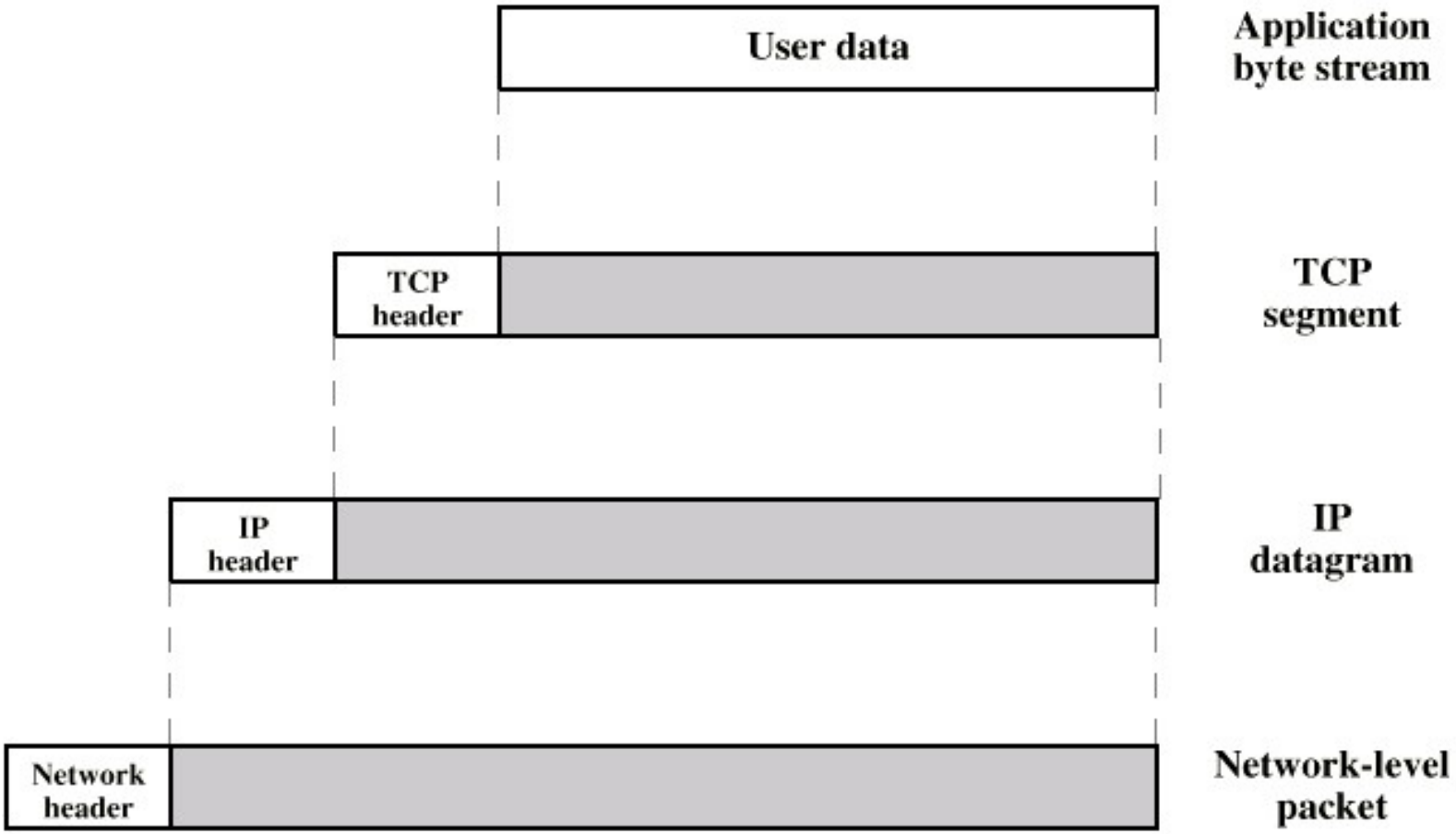
Trace of Simple Operation

- Suppose that a process, associated with port 3 at host A, wishes to send a message to another process, associated with port 2 at host B.
- The process at A hands the message down to TCP with instructions to send it to host B, port 2.
- TCP hands the message down to IP with instructions to send it to host B.
- Note that IP need not be told the identity of the destination port.
- All it needs to know is that the data are intended for host B.
- Next, IP hands the message down to the network access layer (e.g., Ethernet logic) with instructions to send it to router J (the first hop on the way to B).

Operation of TCP and IP

- To control this operation, control information as well as user data must be transmitted, as suggested in below figure.
- Let us say that the sending process generates a block of data and passes this to TCP.
- TCP may break this block into smaller pieces to make it more manageable. To each of these pieces, TCP appends control information known as the TCP header, forming a TCP segment. The control information is to be used by the peer TCP protocol entity at host B.
- **Examples of items in this header include:**
  - **Destination port**
  - **Sequence number**
  - **Checksum**

PDUs in TCP/IP



Operation of TCP and IP

- Next, TCP hands each segment over to IP, with instructions to transmit it to B.
- These segments must be transmitted across one or more subnetworks and relayed through one or more intermediate routers.
- This operation, too, requires the use of control information.
- Thus IP appends a header of control information to each segment to form an IP datagram.
- An example of an item stored in the IP header is the destination host address (in this example, B).

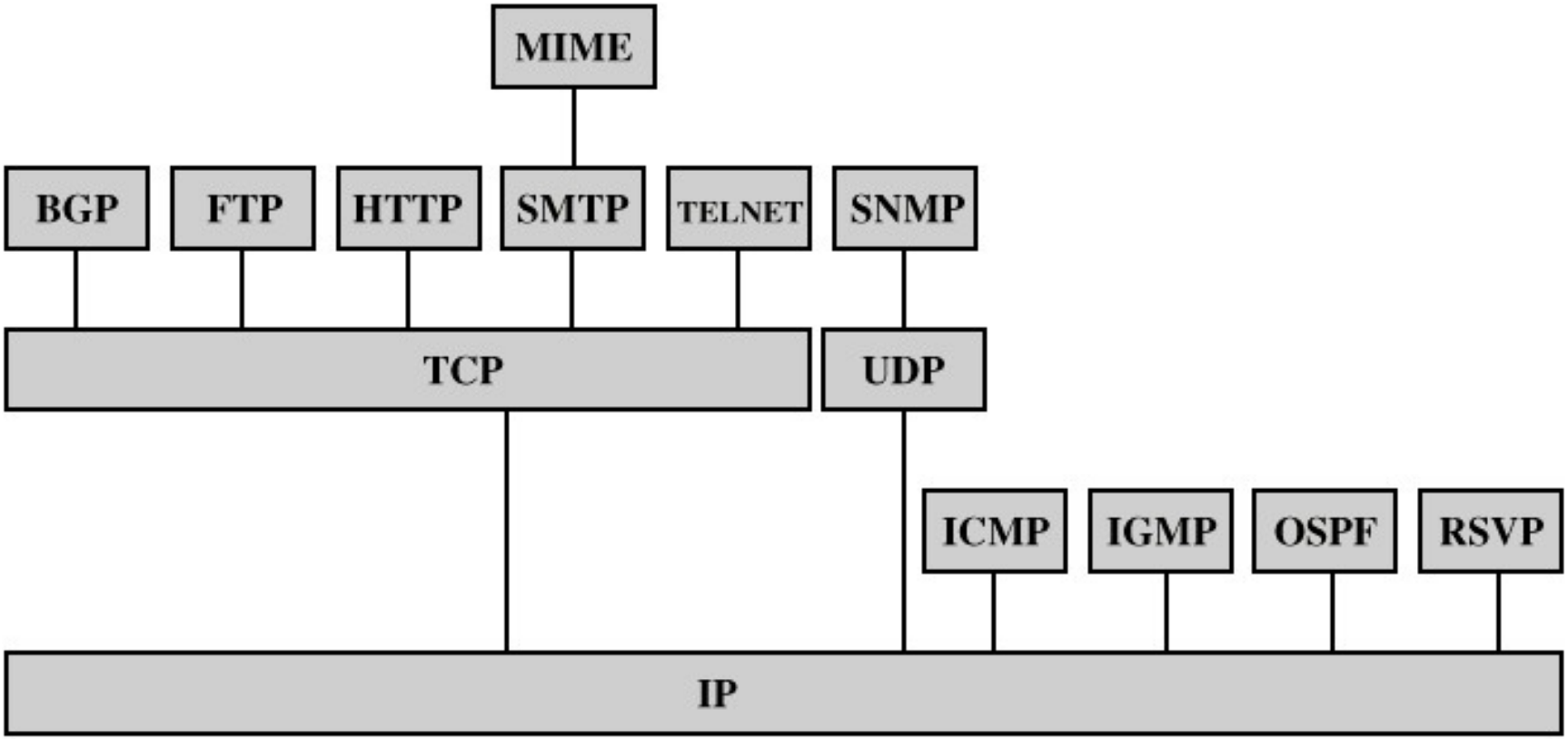
Operation of TCP and IP

- Finally, each IP datagram is presented to the network access layer for transmission across the first subnetwork in its journey to the destination.
- The network access layer appends its own header, creating a packet, or frame.
- The packet is transmitted across the subnetwork to router J.
- The packet header contains the information that the subnetwork needs to transfer the data across the subnetwork.
- Examples of items that may be contained in this header include:
  - Destination subnetwork address
  - Facilities requests

TCP/IP Applications

- A number of applications have been standardized to operate on top of TCP. We mention three of the most common here.
- The **Simple Mail Transfer Protocol (SMTP)** provides a basic electronic mail transport facility. It provides a mechanism for transferring messages among separate hosts. Features of SMTP include mailing lists, return receipts, and forwarding.
- The **File Transfer Protocol (FTP)** is used to send files from one system to another under user command. Both text and binary files are accommodated, and the protocol provides features for controlling user access.
- **TELNET** provides a remote logon capability, which enables a user at a terminal or personal computer to logon to a remote computer and function as if directly connected to that computer.

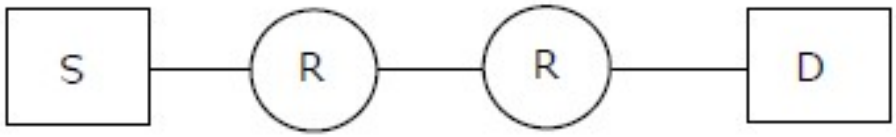
Some Protocols in TCP/IP Suite



BGP = Border Gateway Protocol
FTP = File Transfer Protocol
HTTP = Hypertext Transfer Protocol
ICMP = Internet Control Message Protocol
IGMP = Internet Group Management Protocol
IP = Internet Protocol
MIME = Multi-Purpose Internet Mail Extension
OSPF = Open Shortest Path First
RSVP = Resource ReSerVation Protocol
SMTP = Simple Mail Transfer Protocol
SNMP = Simple Network Management Protocol
TCP = Transmission Control Protocol
UDP = User Datagram Protocol

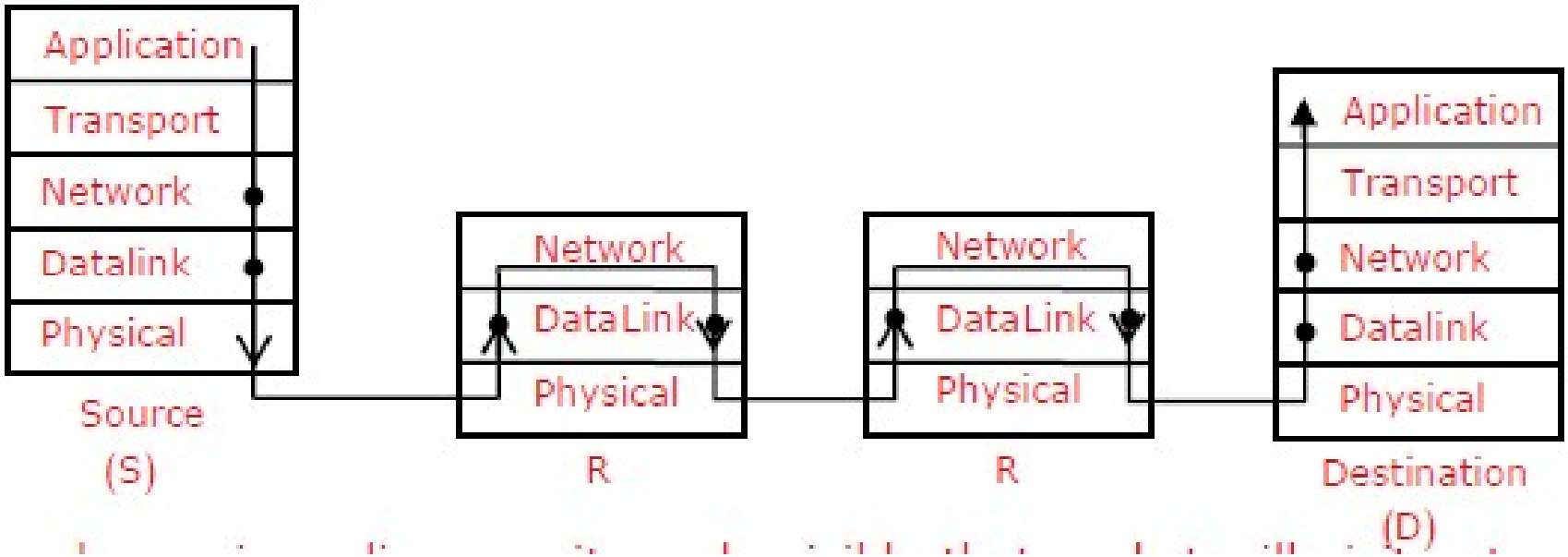
Gate-2013

Assume that source S and destination D are connected through two intermediate routers labeled R. Determine how many times each packet has to visit the network layer and the data link layer during a transmission from S to D.



(A) Network layer – 4 times and Data link layer-4 times
(B) Network layer – 4 times and Data link layer-3 times
(C) Network layer – 4 times and Data link layer-6 times
(D) Network layer – 2 times and Data link layer-6 times

Answer


Application
Transport
Network
Datalink
Physical
Source
(S)
Network
DataLink
Physical
R
Network
DataLink
Physical
R
Application
Transport
Network
Datalink
Physical
Destination
(D)

- Which layer performs encryption and decryption
- A) TL
- B)NL
- C) AL
- D) DLL

UNIT 1 PART 3

Digital Data Communication Techniques: Asynchronous & Synchronous Transmission, Types of Errors, Error Detection, Error Correction.

Introduction

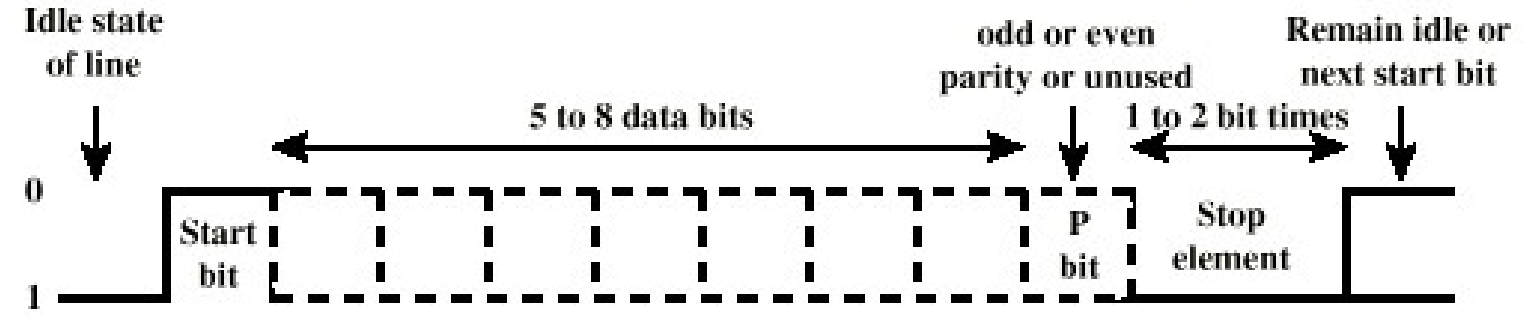
- The transmission of a stream of bits from one device to another across a transmission link involves a great deal of cooperation and agreement between the two sides.
- One of the most fundamental requirements is **synchronization**.
- The receiver must know the rate at which bits are being received so that it can sample the line at appropriate intervals to determine the value of each received bit.
- Two techniques are in common use for this purpose — **asynchronous and synchronous transmission**.

Asynchronous & Synchronous Transmission

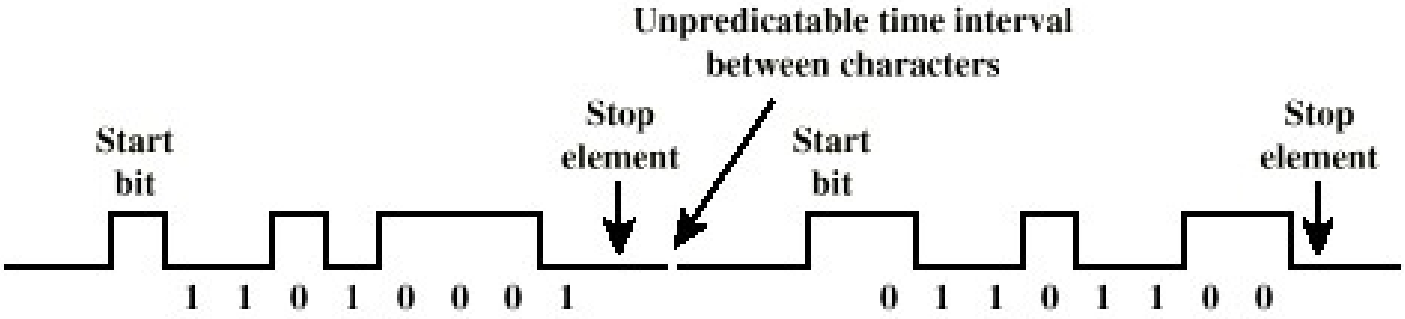
- The reception of digital data involves sampling the incoming signal once per bit time to determine the binary value.
- This is compounded by a timing difficulty: In order for the receiver to sample the incoming bits properly, it must know the arrival time and duration of each bit that it receives.
- Typically, the receiver will attempt to sample the medium at the center of each bit time, at intervals of one bit time. If the receiver times its samples based on its own clock, then there will be a problem if the transmitter's and receiver's clocks are not precisely aligned. If there is a drift in the receiver's clock, then after enough samples, the receiver may be in error because it is sampling in the wrong bit time.
- For smaller timing differences, the error would occur later, but eventually the receiver will be out of step with the transmitter if the transmitter sends a sufficiently long stream of bits and if no steps are taken to synchronize the transmitter and receiver.

Asynchronous Transmission

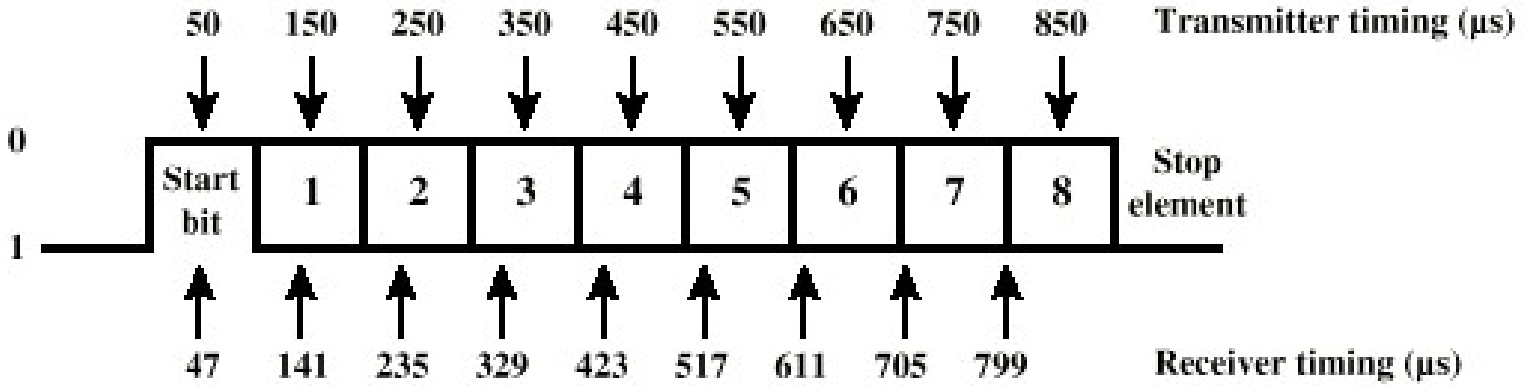
- The strategy with this scheme is to avoid the timing problem by not sending long, uninterrupted streams of bits.
- Instead, data are transmitted one character at a time, where each character is five to eight bits in length.
- Timing or synchronization must only be maintained within each character; the receiver has the opportunity to resynchronize at the beginning of each new character.



(a) Character format



(b) 8-bit asynchronous character stream



(c) Effect of timing error

Asynchronous Transmission

- Above figure illustrates this technique. When no character is being transmitted, the line between transmitter and receiver is in an ***idle* state** (binary 1 level).
- The beginning of a character is signaled by a ***start bit*** with a value of binary 0.
- This is followed by the 5 to 8 bits that actually make up the character. The bits of the character are transmitted beginning with the least significant bit.
- Then the data bits are usually followed by a parity bit, set by the transmitter such that the total number of ones in the character, including the parity bit, is even (even parity) or odd (odd parity). The receiver uses this bit for error detection.
- The final element is a ***stop element***, which is a binary 1.

Asynchronous Transmission

- Figure (c) shows the effects of a timing error of sufficient magnitude to cause an error in reception.
- In this example we assume a data rate of 10,000 bits per second (10 kbps); therefore, each bit is of 0.1 millisecond (ms), or 100 μs, duration.
- Assume that the receiver is fast by 6%, or 6 μs per bit time. Thus, the receiver samples the incoming character every 94 μs (based on the transmitter's clock).
- As can be seen, the last sample is erroneous.

Asynchronous Transmission

- Asynchronous transmission is simple and cheap but requires an overhead of two to three bits per character.
- For example, for an 8-bit character with no parity bit, using a 1-bit-long stop element, two out of every ten bits convey no information but are there merely for synchronization; thus the overhead is 20%.
- The percentage overhead could be reduced by sending larger blocks of bits between the start bit and stop element.
- However, the figure (c) indicates, the larger the block of bits, the greater the cumulative timing error.
- To achieve greater efficiency, a different form of synchronization, known as synchronous transmission, is used.

Summary

- In **asynchronous transmission**, each character of data is treated independently.
- Each character begins with a start bit that alerts the receiver that a character is arriving. The receiver samples each bit in the character and then looks for the beginning of the next character.
- This technique would not work well for long blocks of data because the receiver's clock might eventually drift out of synchronization with the transmitter's clock.
- However, sending data in large blocks is more efficient than sending data one character at a time.

Synchronous Transmission

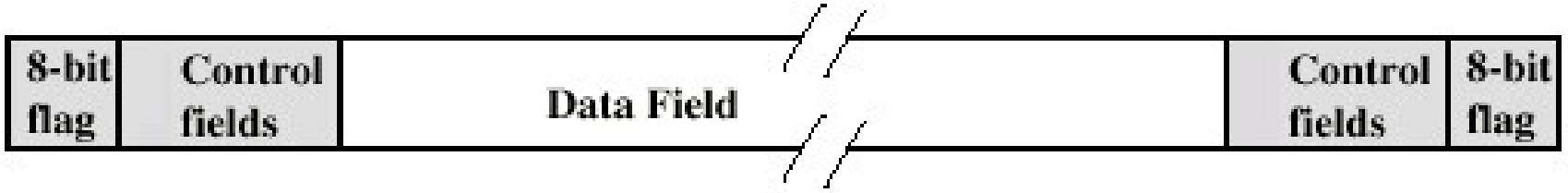
- With **synchronous transmission**, a block of bits is transmitted in a steady stream without start and stop codes.
- The block may be many bits in length. To prevent timing drift between transmitter and receiver, their clocks must somehow be synchronized.
- One possibility is to provide a separate clock line between transmitter and receiver.
- The other alternative is to embed the clocking information in the data signal.
- For digital signals, this can be accomplished with Manchester or differential Manchester encoding.
- For analog signals, a number of techniques can be used; for example, the carrier frequency itself can be used to synchronize the receiver based on the phase of the carrier.

Synchronous Transmission

- With synchronous transmission, there is another level of synchronization required, to allow the receiver to determine the beginning and end of a block of data.
- To achieve this, each block begins with a ***preamble*** bit pattern and generally ends with a ***postamble*** bit pattern.
- In addition, other bits are added to the block that convey control information used in the data link control procedures.
- The data plus preamble, postamble, and control information are called a **frame**.
- The exact format of the frame depends on which data link control procedure is being used.

Synchronous Transmission

- Below figure shows a typical frame format for synchronous transmission.
- It starts with a **preamble** called a **flag**, which is 8 bits long.
- The same flag is used as a **postamble**.
- This is followed by some number of control fields (containing data link control protocol information), then a data field (variable length for most protocols), more control fields, and finally the flag is repeated.
- For sizable blocks of data, synchronous transmission is far more efficient than asynchronous.
- Asynchronous transmission requires 20% or more overhead.
- The control information, preamble, and postamble in synchronous transmission are typically less than 100 bits.



Summary

- For large blocks, **synchronous transmission** is used.
- Each block of data is formatted as a frame that includes a starting and an ending flag.
- Some form of synchronization, such as the use of Manchester encoding, is employed.

Types of Errors

- In digital transmission systems, an error occurs when a bit is altered between transmission and reception; that is, a binary 1 is transmitted and a binary 0 is received, or a binary 0 is transmitted and a binary 1 is received. Two general types of errors can occur: **single-bit errors** and **burst errors.**
- A **single-bit error** is an isolated error condition that alters one bit but does not affect nearby bits. A single-bit error can occur in the presence of white noise, when a slight random deterioration of the signal-to-noise ratio is sufficient to confuse the receiver's decision of a single bit.
- A **burst error** of length *B* is a contiguous sequence of *B* bits in which the first and last bits and any number of intermediate bits are received in error. Burst errors are more common and more difficult to deal with. Burst errors can be caused by impulse noise, and by fading in a mobile wireless environment.
- **Note** that the effects of burst errors are greater at higher data rates.

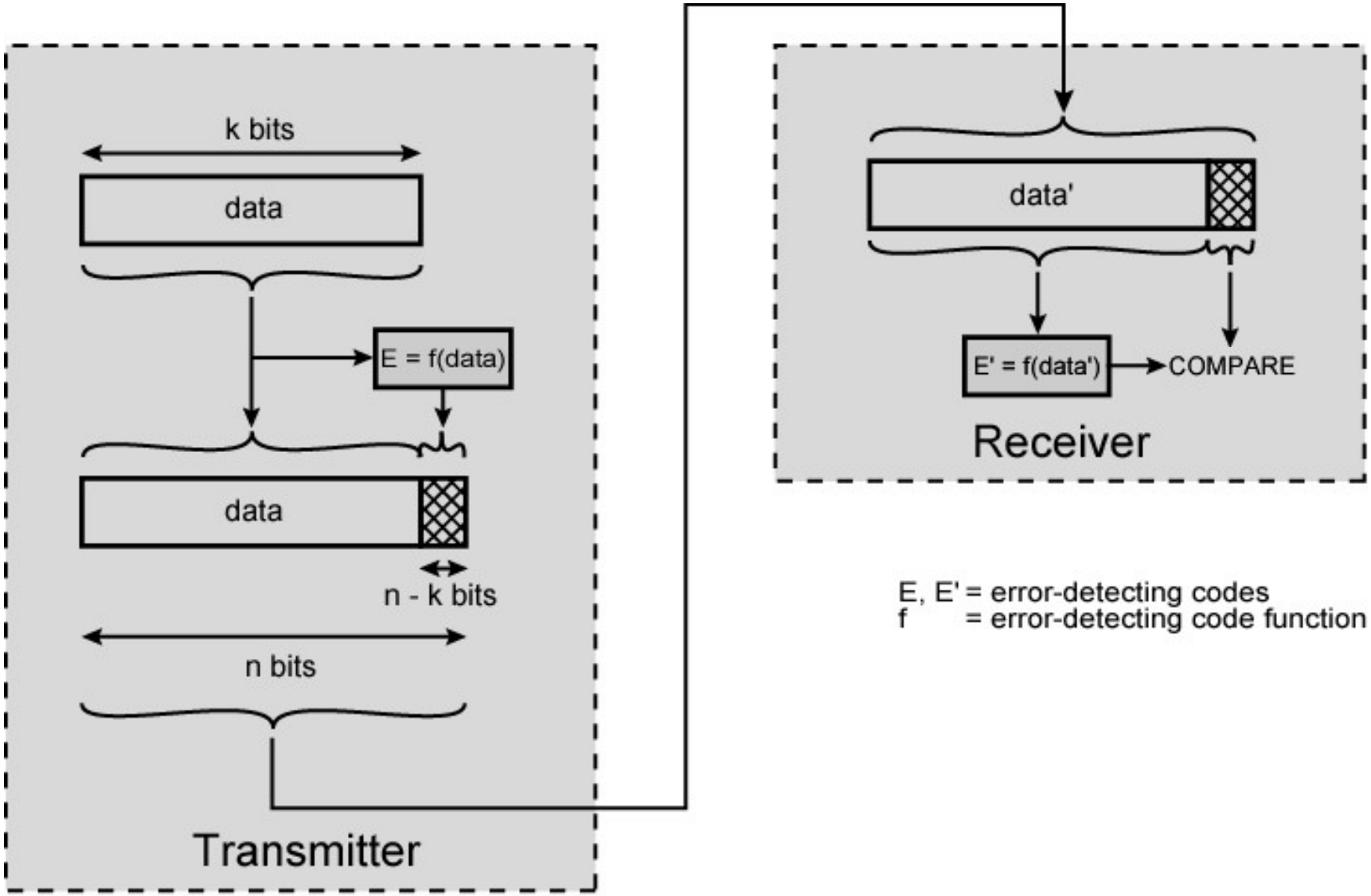
Error Detection

- Regardless of the design of the transmission system, there will be errors, resulting in the change of one or more bits in a transmitted frame.
- **Error detection** is performed by calculating an error-detecting code that is a function of the bits being transmitted.
- The code is appended to the transmitted bits.
- The receiver calculates the code based on the incoming bits and compares it to the incoming code to check for errors.
- A detected error occurs if and only if there is a mismatch. There is a probability that a frame contains errors and that the error-detecting scheme will detect that fact.
- *Also have a* residual error rate, being the probability that an error will be undetected despite the use of an error-detecting scheme.

Error Detection

- Below figure illustrates the error detection process.
- For a given frame of bits, additional bits that constitute an **error-detecting code** are added by the transmitter.
- This code is calculated as a function of the other transmitted bits.
- Typically, for a data block of k bits, the error-detecting algorithm yields an error-detecting code of $n – k$ bits, where $(n – k) < k$.
- The error-detecting code, also referred to as the **check bits**, is appended to the data block to produce a frame of n bits, which is then transmitted.
- The receiver separates the incoming frame into the k bits of data and $(n – k)$ bits of the error-detecting code.
- The receiver performs the same error-detecting calculation on the data bits and compares this value with the value of the incoming error-detecting code.
- A detected error occurs if and only if there is a mismatch.

Error Detection Process



Parity Check

- The simplest error-detecting scheme is to append a parity bit to the end of a block of data. The value of this bit is selected so that the character has an even number of 1s (even parity) or an odd number of 1s (odd parity).
- Note, however, that if two (or any even number) of bits are inverted due to error, an undetected error occurs.
- Typically, even parity is used for synchronous transmission and odd parity for asynchronous transmission.
- The use of the parity bit is not foolproof, as noise impulses are often long enough to destroy more than one bit, particularly at high data rates.

Cyclic Redundancy Check (CRC)

- One of the most common, and one of the most powerful, error-detecting codes is the Cyclic Redundancy Check (CRC), which can be described as follows.
- Given a *k*-bit block of bits, or message, the transmitter generates an (*n* – *k*)-bit sequence, known as a Frame Check Sequence (FCS), such that the resulting frame, consisting of *n* bits, is exactly divisible by some predetermined number.
- The receiver then divides the incoming frame by that number and, if there is no remainder, assumes there was no error.
- Can state this procedure in three equivalent ways:
  - **modulo 2 arithmetic**
  - **polynomials**
  - digital logic

Cyclic Redundancy Check (CRC)

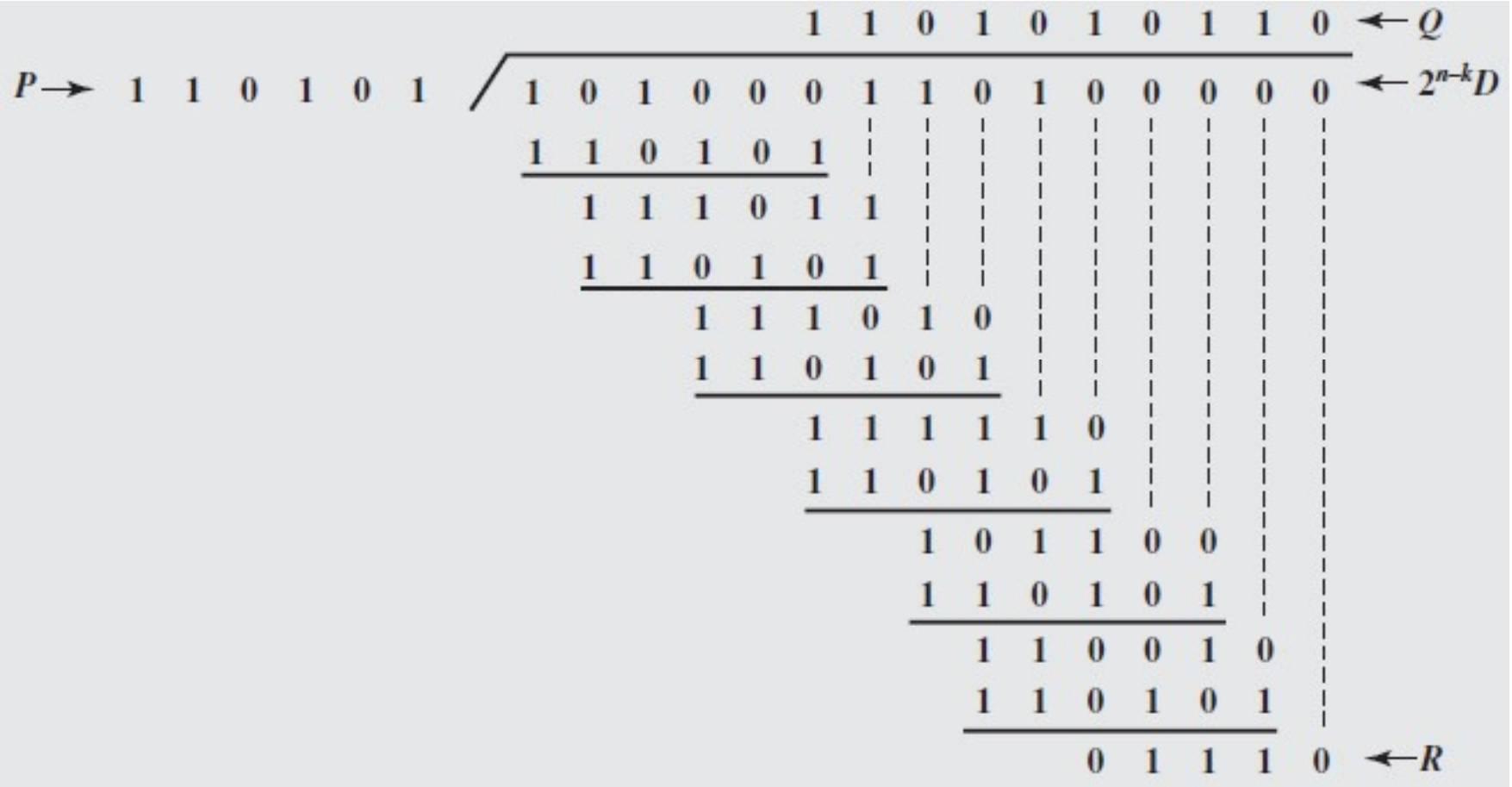
Given

Message D=1010001101 (10 bits)

Pattern P=($x^5+x^4+x^2+1$) 110101 (6 bits)

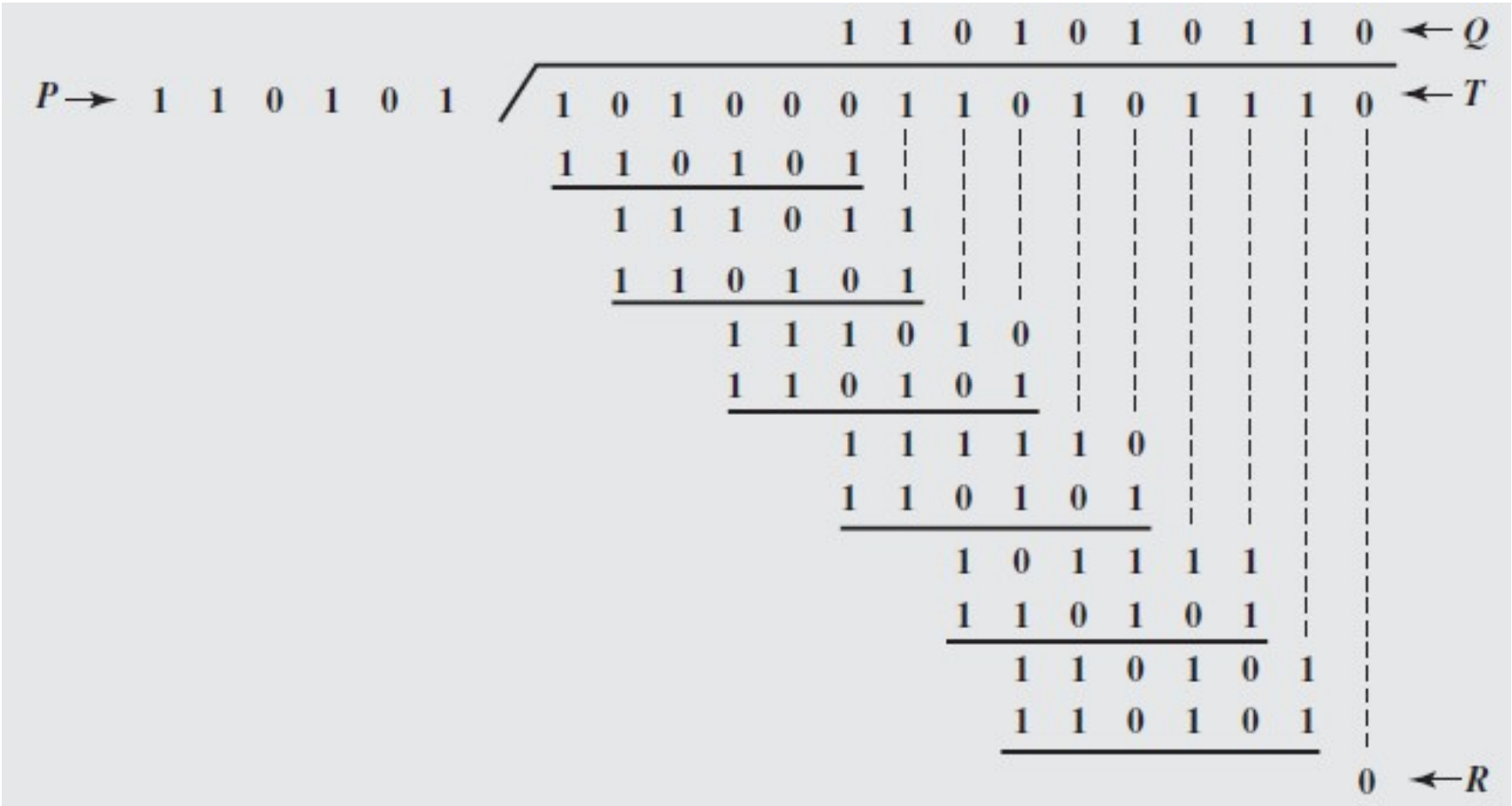
FCS R=to be calculated (5 bits) i.e, P-1

Cyclic Redundancy Check (CRC)



Now the Transmitted Frame T=1010001101**01110**

Cyclic Redundancy Check (CRC)



Because there is no remainder, it is assumed that there have been no errors.

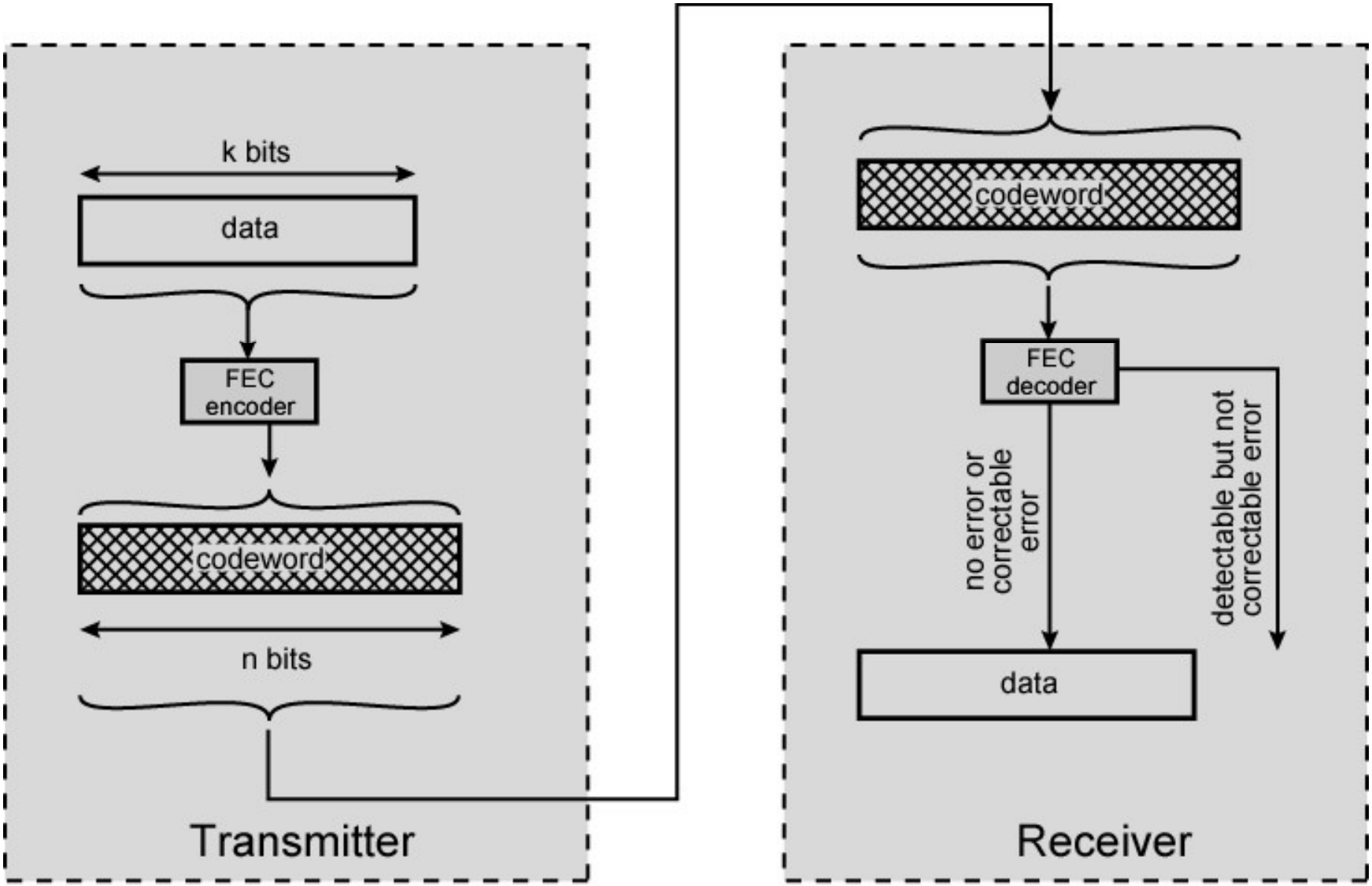
Error Correction

- Error detection is a useful technique, found in data link control protocols, such as HDLC, and in transport protocols, such as TCP. However, correction of errors using an error-detecting code, requires that block of data be retransmitted.
- For wireless applications this approach is inadequate for two reasons:

1. The bit error rate on a wireless link can be quite high, which would result in a large number of retransmissions.
2. In some cases, especially satellite links, the propagation delay is very long compared to the transmission time of a single frame. With a long data link, an error in a single frame necessitates retransmitting many frames.

- Instead, it would be desirable to enable the receiver to correct errors in an incoming transmission on the basis of the bits in that transmission. **Error correction** operates in a fashion similar to error detection but is capable of correcting certain errors in a transmitted bit stream.

Error Correction Process Diagram



Error Correction Process

- Above figure shows in general how this is done. On the transmission end, each *k*-bit block of data is mapped into an *n*-bit block (*n* > *k*) called a **codeword**, using an **FEC (Forward Error Correction)** encoder. The codeword is then transmitted. During transmission, the signal is subject to impairments, which may produce bit errors in the signal. At the receiver, the incoming signal is demodulated to produce a bit string that is similar to the original codeword but may contain errors. This block is passed through an FEC decoder, with one of four possible outcomes:

1. If there are no bit errors, the input to the FEC decoder is identical to the original codeword, and the decoder produces the original data block as output.
2. For certain error patterns, it is possible for the decoder to detect and correct those errors, the FEC decoder is able to map this block into the original data block.
3. For certain error patterns, the decoder can detect but not correct the errors, the decoder simply reports an uncorrectable error.
4. For certain, typically rare, error patterns, the decoder does not detect that any errors have occurred and maps the incoming data block into a block different from the original.

Working of Error Correction

- In essence, error correction works by adding redundancy to the transmitted message. The redundancy makes it possible for the receiver to deduce what the original message was, even in the face of a certain level of error rate.
- In this section we look at a widely used form of error-correcting code known as a block error-correcting code. If wish to transmit blocks of data of length k bits, so map each k-bit sequence into a unique n-bit codeword, which differ significantly from each other.
- Typically, each valid codeword reproduces the original k data bits and adds to them $(n - k)$ check bits to form the n-bit codeword. Then if an invalid codeword is received, assume the valid codeword is the one that is closest to it, and use the input bit sequence associated with it.
- The ratio of redundant bits to data bits, $(n - k)/k$, is called the **redundancy** of the code, and the ratio of data bits to total bits, k/n, is called the **code rate**. The code rate is a measure of how much additional bandwidth is required to carry data at the same data rate as without the code. For example, a code rate of 1/2 requires double the transmission capacity of an uncoded system to maintain the same data rate.

Error Correction Techniques

1. **Hamming codes**
2. Binary convolutional codes
3. Reed-Solomon codes
4. Low-Density Parity Check codes

Hamming Codes

- The Hamming Code method is a network technique designed by R.W.Hamming, for damage and error detection during data transmission between multiple network channels.
- The Hamming Code method is one of the most effective ways to detect single-data bit errors in the original data at the receiver end. It is not only used for **error detection** but is also for **correcting errors** in the data bit.

Important Terms for Hamming Code

- To begin with, the steps involved in the detection and correction of data using hamming code, we need to understand some important terms and expressions, which are:

1. **Redundant Bits** - These are the extra binary bits added externally into the original data bit to prevent damage to the transmitted data and are also needed to recover the original data.

- The expression applied to deduce the redundant value is, $2^r >= d+r+1$

  Where,

  d - “Data Bits”

  r - “Redundant Bits”, r = {1, 2, 3, …….. n}

Important Terms for Hamming Code

- **Example**: **Assuming the number of data bits is 7, find the number of redundant bits.**

Ans:-

Given d=7

2^r >= d+r+1

2^1>=7+1+1, 2>=9 [r=1]

2^2>=7+2+1, 4>=10 [r=2]

2^3>=7+3+1, 8>=11 [r=3]

2^4 >=7+4+1, 16>=12 [r=4]

The number of redundant bits = 4.

Important Terms for Hamming Code

2. **Parity Bits** - The parity bit is the method to append binary bits to ensure that the total count of 1's in the original data is even bit or odd. It is also applied to detect errors on the receiver side and correct them.

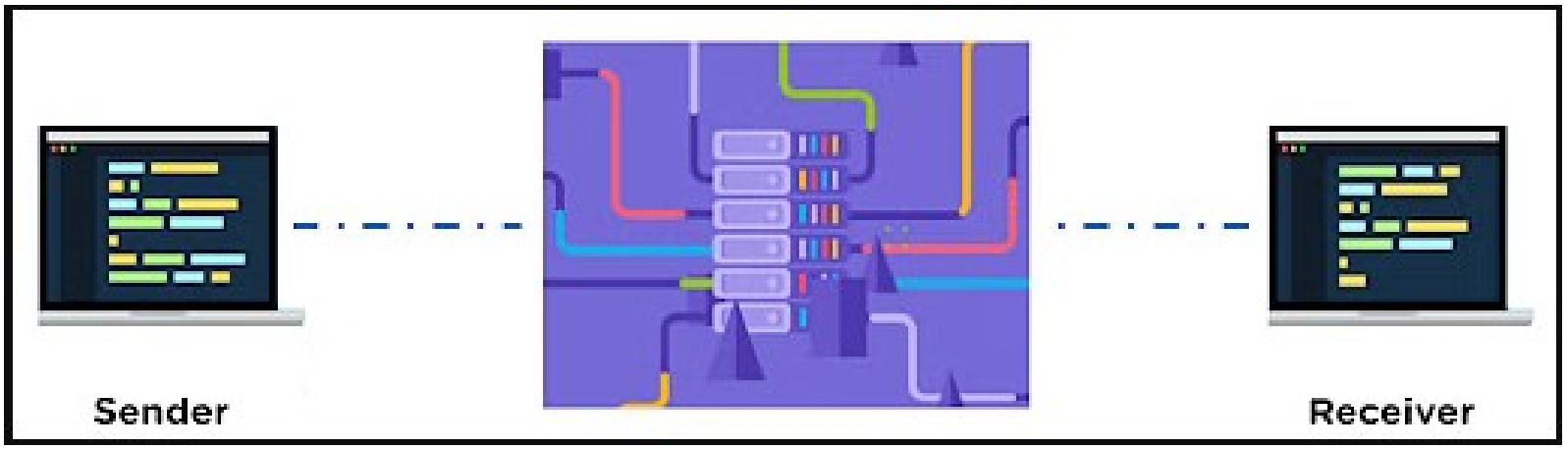
Types of parity bits:

- **Odd Parity bits** - In this parity type, the total number of 1's in the data bit should be odd in count, then the parity value is 0, and the value is 1.
- **Even Parity bits** - In this parity type, the total number of 1's in the data bit should be even in count; then the parity value is 0, and the value is 1.

Working of Hamming Code

- To solve the data bit issue with the hamming code method, some steps need to be followed:
  - Step 1 - The position of the data bits and the number of redundant bits in the original data. The number of redundant bits is deduced from the expression [2^r >= d+r+1].
  - Step 2 - Fill in the data bits and redundant bit, and find the parity bit value using the expression [2^p, where, p - {0,1,2, …… n}].
  - Step 3 - Fill the parity bit obtained in the original data and transmit the data to the receiver side.
  - Step 4 - Check the received data using the parity bit and detect any error in the data, and in case damage is present, use the parity bit value to correct the error.

Example for Hamming Code



•To better understand the working of the hamming code, the following example is to be solved:

•The data bit to be transmitted is 1011010, to be solved using the hamming code method.

Determining the Number of Redundant Bits and Position in the Data

- The data bits = 7
  The redundant bit,
  2^r >= d+r+1
  2^4 >= 7+4+1
  16 >= 12, [So, the value of r = 4.]
- Position of the redundant bit, applying the 2^p expression:
  2^0 - P1
  2^1 - P2
  2^2 - P4
  2^3 - P8
- Total no.of bits to be transmitted =d+r=7+4=11

Finding the Parity Bits, for ”Even parity bits”

1. P1 parity bit is deduced by checking all the bits with 1’s in the least significant location.
   P1: 1, 3, 5, 7, 9, 11
   P1 - P1, 1, 0, 1, 0, 0
   P1 - 0
2. P2 parity bit is deduced by checking all the bits with 1’s in the second significant location.
   P2: 2, 3, 6, 7, 10, 11
   P2 - P2, 1, 1, 1, 1, 0
   P2 – 0
3. P4 parity bit is deduced by checking all the bits with 1’s in the third significant location.
   P4: 4, 5, 6, 7
   P4 - P4, 0, 1, 1
   P4 – 0
4. P8 parity bit is deduced by checking all the bits with 1’s in the fourth significant location.
   P8: 8, 9, 10, 11
   P8 – P8, 0, 1, 0
   P8 - 1

d=1011010(7 bits)

1	2	3	4	5	6	7	8	9	10	11
P1	P2	M1	P4	M2	M3	M4	P8	M5	M6	M7
		1		0	1	1		0	1	0

So, the original data to be transmitted to the receiver side is:

1	2	3	4	5	6	7	8	9	10	11
P1	P2	M1	P4	M2	M3	M4	P8	M5	M6	M7
0	0	1	0	0	1	1	1	0	1	0

00100111010

1	2	3	4	5	6	7	8	9	10	11
0	0	1	0	0	1	1	1	0	1	0

C1 (1, 3, 5, 7, 9, 11) = 0, 1, 0, 1, 0, 0 = 0

C2 (2, 3, 6, 7, 10, 11) = 0, 1, 1, 1, 1, 0 = 0

C4 (4, 5, 6, 7) = 0, 0, 1, 1 = 0

C8 (8, 9, 10, 11) = 1, 0, 1, 0 = 0

All C8 C4 C2 C1=0000 NO ERROR

Placed Error at 7th Position

00100101010

1	2	3	4	5	6	7	8	9	10	11
0	0	1	0	0	1	0	1	0	1	0

C1 (1, 3, 5, 7, 9, 11) = 0, 1, 0, 0, 0, 0 = 1

C2 (2, 3, 6, 7, 10, 11) = 0, 1, 1, 0, 1, 0 = 1

C4 (4, 5, 6, 7) = 0, 0, 1, 0 = 1

C8 (8, 9, 10, 11) = 1, 0, 1, 0 = 0

C8 C4 C2 C1=0111=7 so 7th bit error Replace 0 with 1 in 7th Position

UNIT 1 PART 4

Data Link Control: Flow Control, Error Control.

Data Link Control Protocols

- In this chapter, we shift our emphasis to that of *sending data over a data communications link*.
- To achieve the necessary control, a layer of logic is added above the physical layer, referred to as **data link control** or a **data link control protocol**.

Data Link Control Protocols

- Some of the requirements and objectives for effective data communication between two directly connected transmitting-receiving stations:
  - **Frame synchronization:** Data are sent in blocks called frames. The beginning and end of each frame must be recognizable.
  - **Flow control:** The sending station must not send frames at a rate faster than the receiving station can absorb them.
  - **Error control:** Bit errors introduced by the transmission system should be corrected.
  - **Addressing:** On a shared link, such as a local area network (LAN), the identity of the two stations involved in a transmission must be specified.
  - **Control and data on same link:** the receiver must be able to distinguish control information from the data being transmitted.
  - **Link management:** Procedures for the management of initiation, maintenance, and termination of a sustained data exchange over a link.

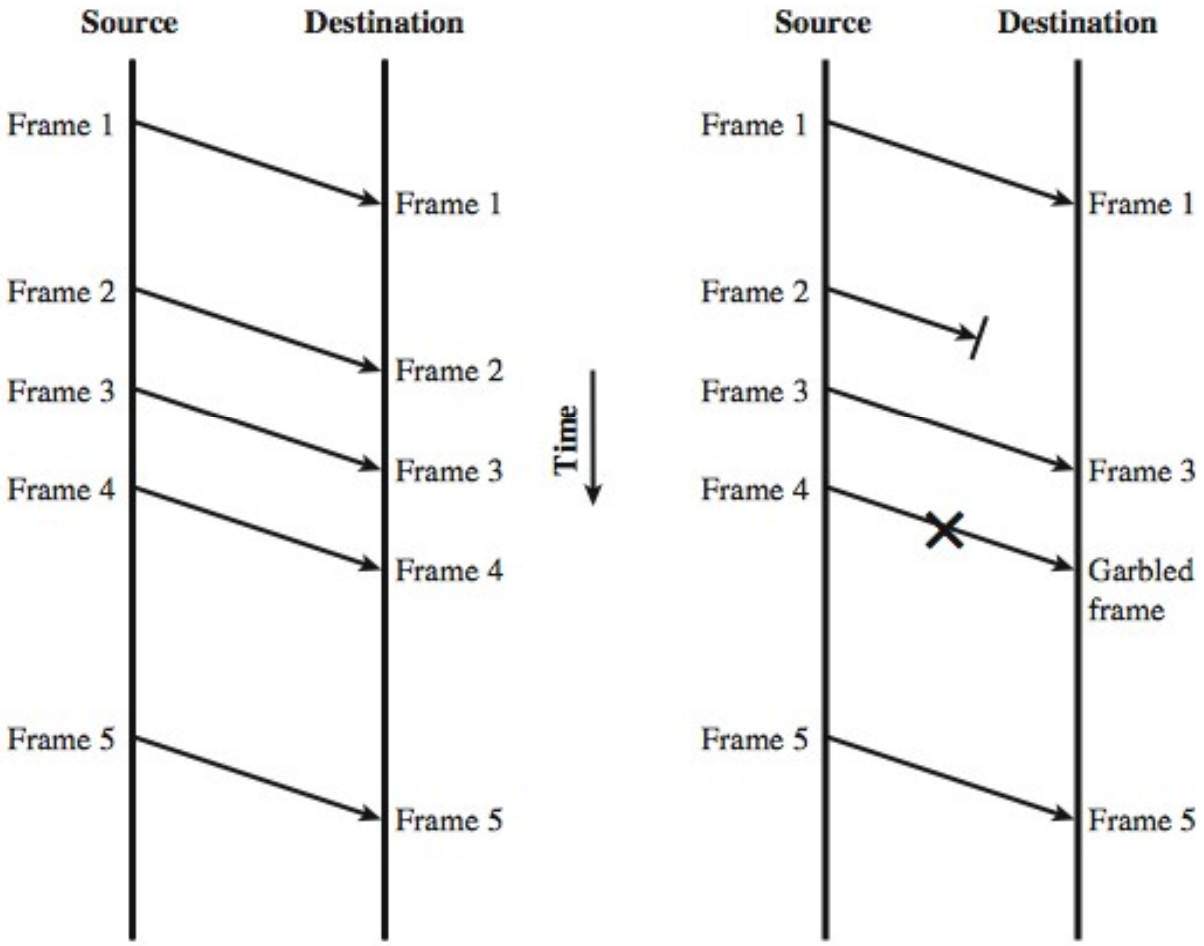
Flow Control

- Flow control is a technique for assuring that a transmitting entity does not overwhelm a receiving entity with data.
- The receiving entity typically allocates a data buffer of some maximum length for a transfer.
- When data are received, the receiver must do a certain amount of processing before passing the data to the higher-level software.
- In the absence of flow control, the receiver's buffer may fill up and overflow while it is processing old data.

Flow Control

- To begin, we examine mechanisms for flow control in the absence of errors.
- The model we will use is depicted in below figure, which is a vertical-time sequence diagram.
- It has the advantages of showing time dependencies and illustrating the correct send-receive relationship.
- Each arrow represents a single frame transiting a data link between two stations.
- The data are sent in a sequence of frames, with each frame containing a portion of the data and some control information.

Flow Control



(a) Error-free transmission

(b) Transmission with losses and errors

Flow Control

- The time it takes for a station to emit all of the bits of a frame onto the medium is the **transmission time**; this is proportional to the length of the frame.
- The **propagation time** is the time it takes for a bit to traverse the link between source and destination.
- For this section, we assume that all frames that are transmitted are successfully received; no frames are lost, they arrive in order sent, and none arrive with errors.
- However, each transmitted frame suffers an arbitrary and variable amount of delay before reception.

Flow Control

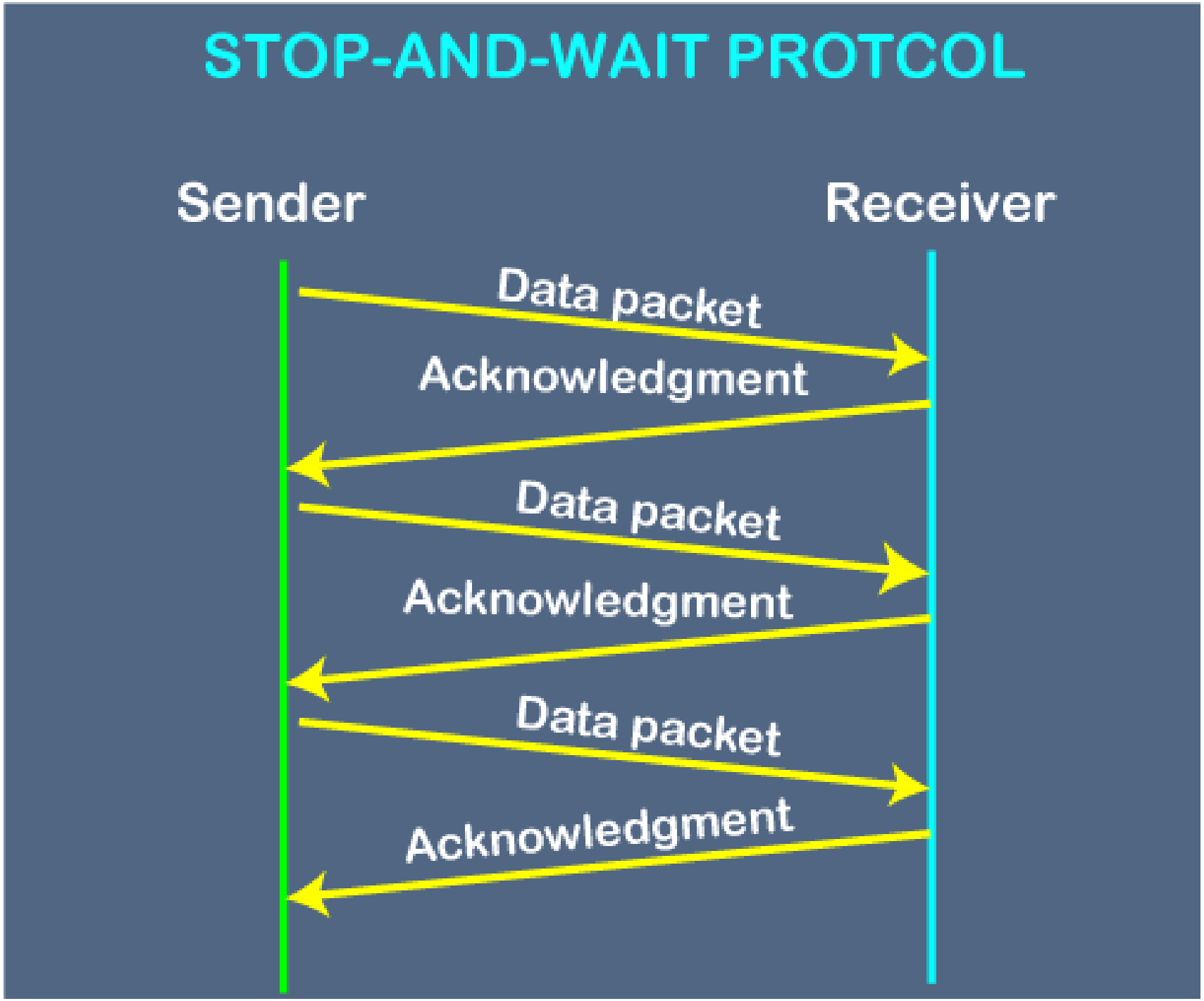
- Stop-and-Wait Flow Control
- Sliding-Window Flow Control

Stop-and-Wait Flow Control

- The simplest form of flow control, known as stop-and-wait flow control, works as follows.
- A source entity transmits a frame. After the destination entity receives the frame, it indicates its willingness to accept another frame by sending back an acknowledgment to the frame just received.
- The source must wait until it receives the acknowledgment before sending the next frame. The destination can thus stop the flow of data simply by withholding acknowledgment.
- This procedure works fine and, indeed, can hardly be improved upon when a message is sent in a few large frames.

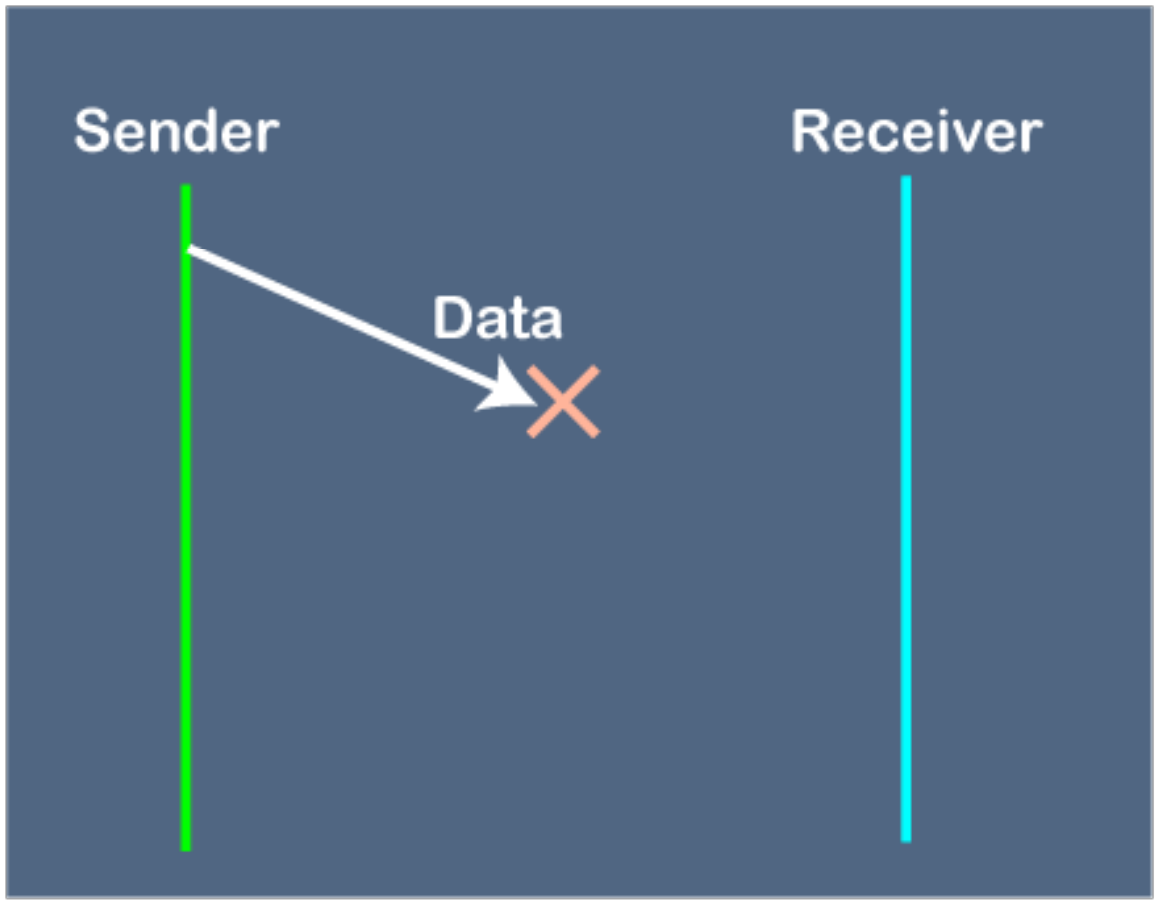
Stop-and-Wait Flow Control

- However, it is often the case that a source will break up a large block of data into smaller blocks and transmit the data in many frames **(because of limited buffer size, errors detected sooner with less to resend, to prevent media hogging).**
- This is done for the following reasons:
  - The buffer size of the receiver may be limited.
  - The longer the transmission, the more likely that there will be an error, necessitating retransmission of the entire frame. With smaller frames, errors are detected sooner, and a smaller amount of data needs to be retransmitted.
  - On a shared medium, such as a LAN, it is usually desirable not to permit one station to occupy the medium for an extended period, thus causing long delays at the other sending stations.
- With the use of multiple frames for a single message, the stop-and-wait procedure may be inadequate, mainly since only one frame at a time can be in transit.


STOP-AND-WAIT PROTCOL
Sender
Receiver
Data packet
Acknowledgment
Data packet
Acknowledgment
Data packet
Acknowledgment

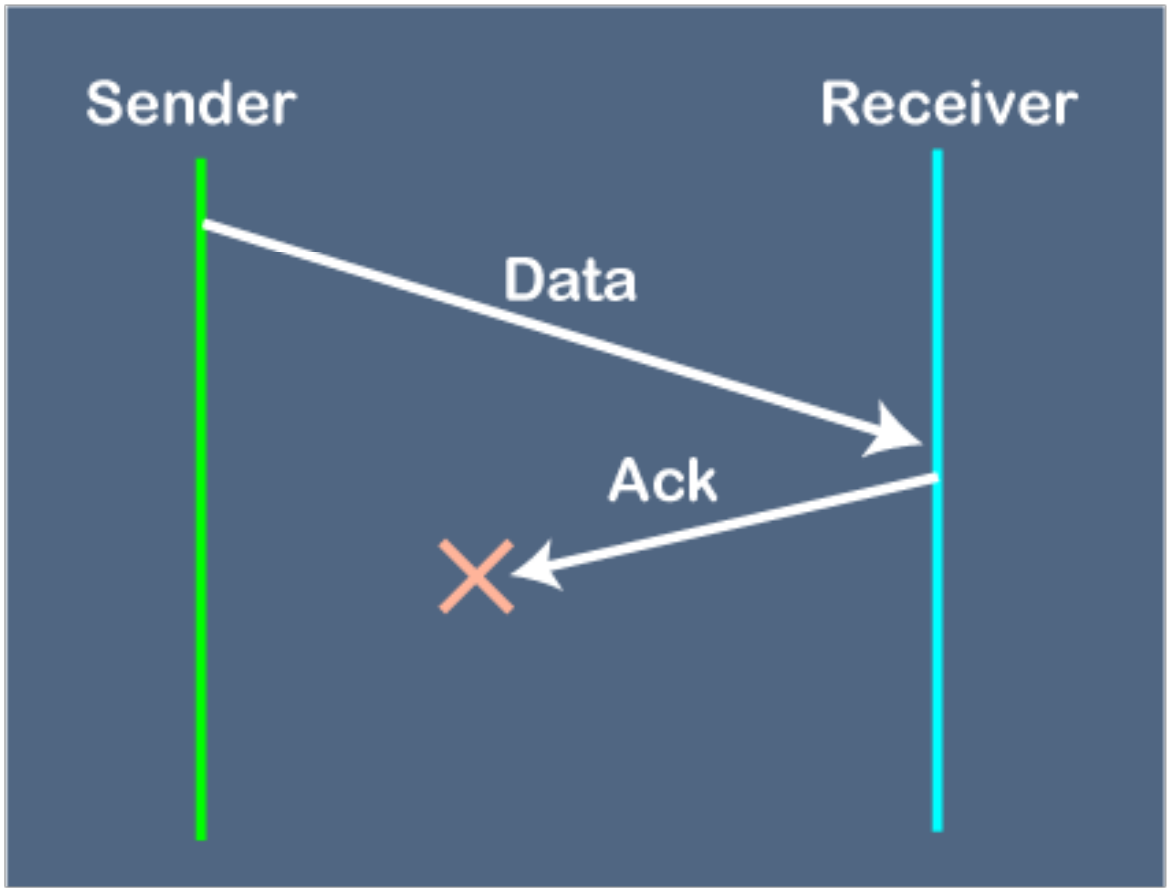
The following are the problems associated with a stop and wait protocol

1. **Problems occur due to lost data**



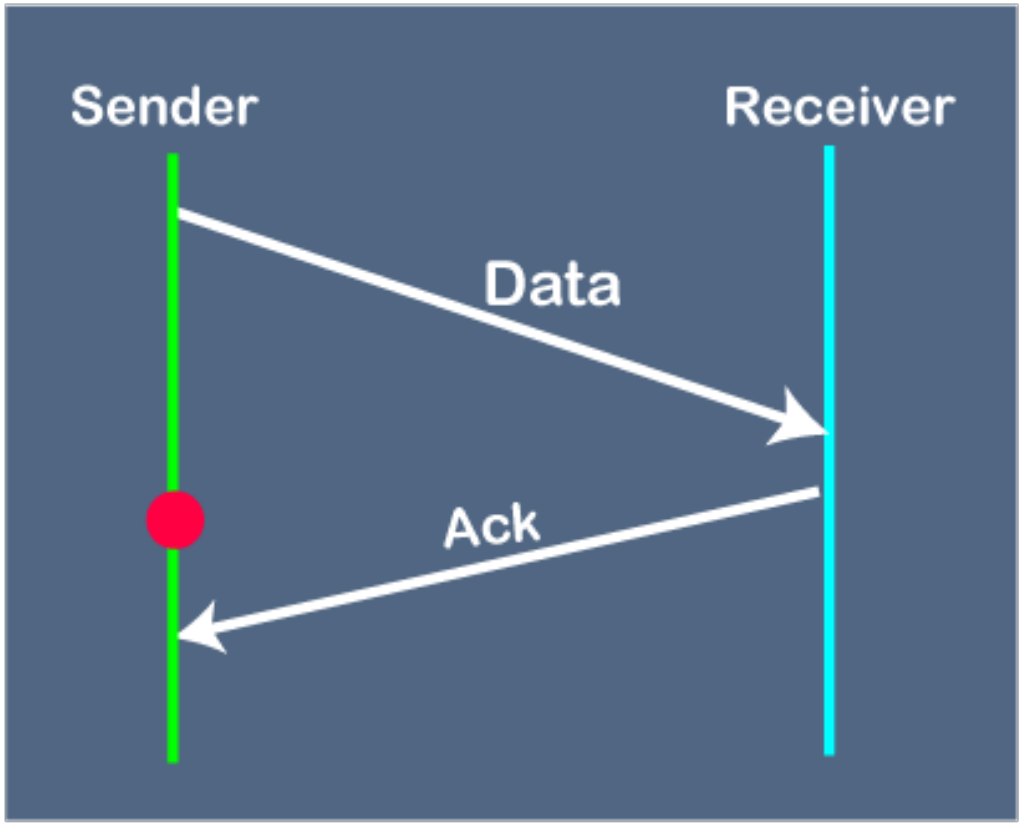
- **In this case, two problems occur:**
  - Sender waits for an infinite amount of time for an acknowledgment.
  - Receiver waits for an infinite amount of time for a data.

2. Problems occur due to lost acknowledgment



- **In this case, one problem occurs:**
  - Sender waits for an infinite amount of time for an acknowledgment.

3. **Problem due to the delayed data or acknowledgment**



Stop-and-Wait Flow Control

- To show this, start by define the **bit length B of a link** as follows:

$$B = R \times \frac{d}{V}$$

where

B = length of the link in bits; this is the number of bits present on the link at an instance in time when a stream of bits fully occupies the link

R = data rate of the link, in bps

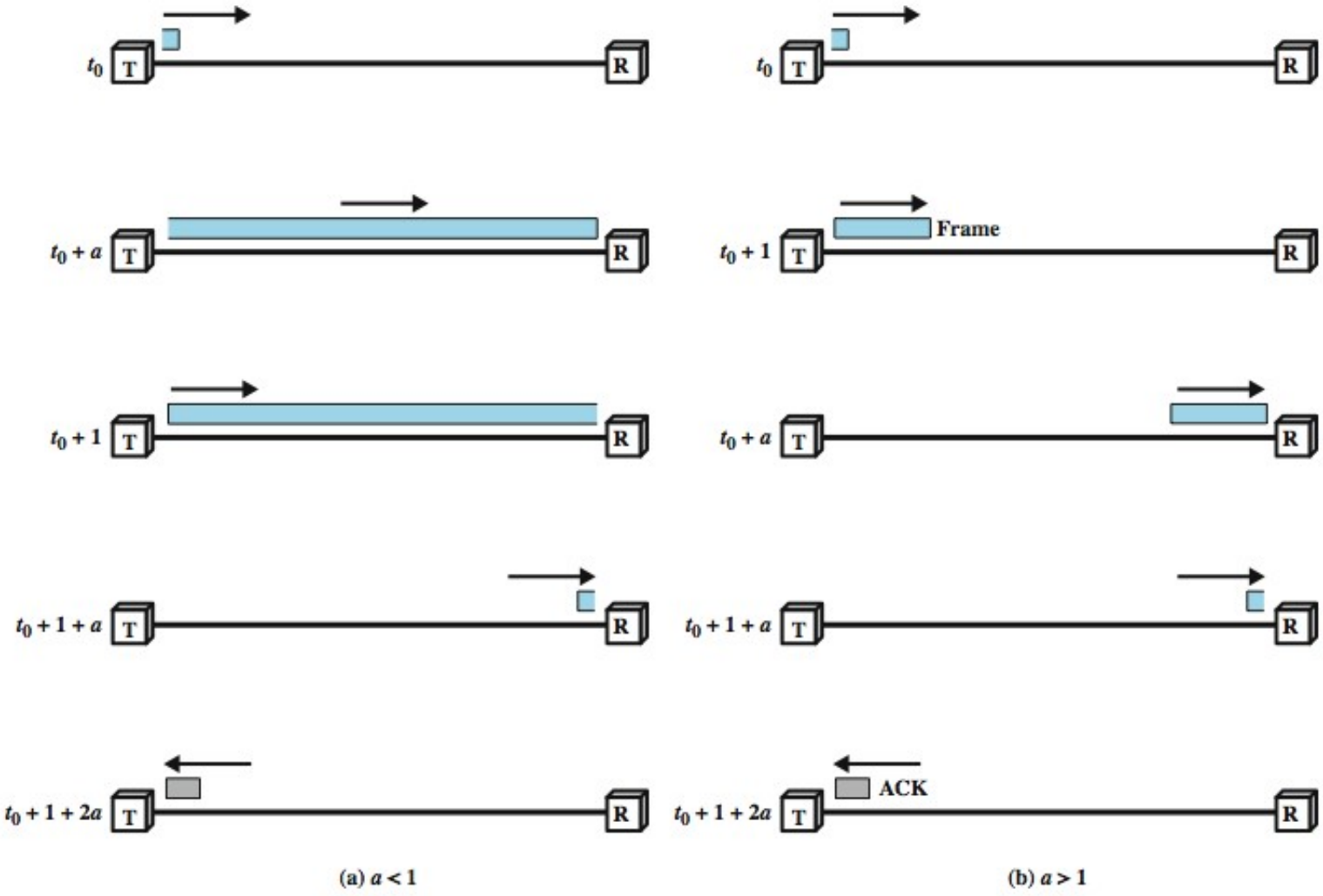
d = length, or distance, of the link in meters

V = velocity of propagation, in m/s

Stop-and-Wait Flow Control

- In situations where the bit length of the link is greater than the frame length, serious inefficiencies result, as shown in below figure.
- In the figure, the transmission time (the time it takes for a station to transmit a frame) is normalized to one, and the propagation delay (the time it takes for a bit to travel from sender to receiver) is expressed as the variable $a = B / L$, where L is the number of bits in the frame.

Stop and Wait Link Utilization



Stop-and-Wait Flow Control

- When *a* is less than 1, the propagation time is less than the transmission time.
  - In this case, the frame is sufficiently long that the first bits of the frame have arrived at the destination before the source has completed the transmission of the frame.
- When *a* is greater than 1, the propagation time is greater than the transmission time.
  - In this case, the sender completes transmission of the entire frame before the leading bits of that frame arrive at the receiver.

Stop-and-Wait Flow Control

- Both parts of above figure (a and b) consist of a sequence of snapshots, the first four show the process of transmitting a data frame, and the last shows the return of a small acknowledgment frame.
- Note that for $a > 1$, the line is always underutilized and even for $a < 1$, the line is inefficiently utilized.
- In essence, for very high data rates, for very long distances between sender and receiver, stop-and-wait flow control provides inefficient line utilization.

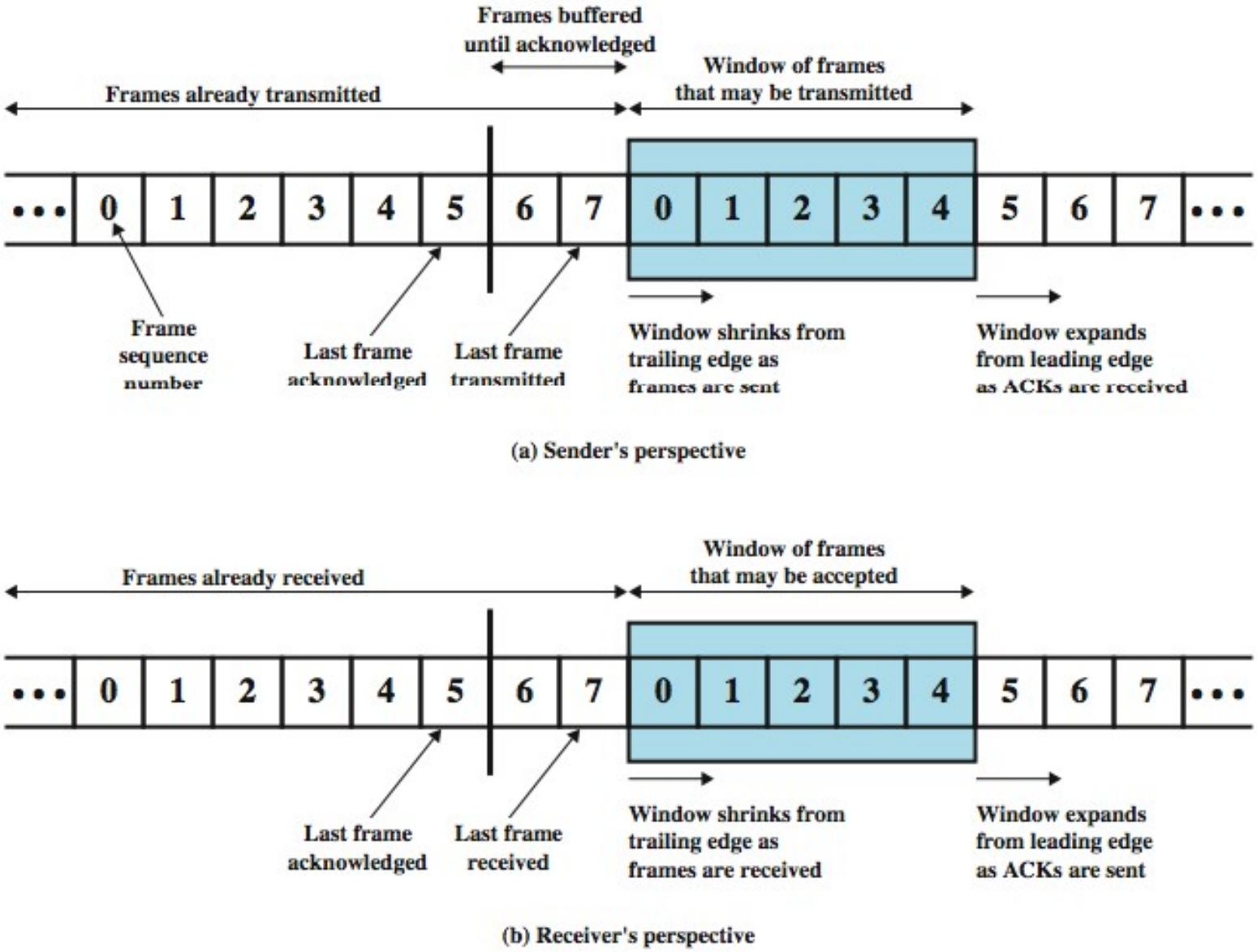
Sliding-Window Flow Control

- The essence of the problem described so far is that only one frame at a time can be in transit.
- Efficiency can be greatly improved by allowing multiple frames to be in transit at the same time.
- Consider two stations, A and B, connected via a **full-duplex** link.
- Station B allocates buffer space for *W* frames. Thus, B can accept *W* frames, and A is allowed to send *W* frames without waiting for any acknowledgments.

Sliding-Window Flow Control

- To keep track of which frames have been acknowledged, each is labeled with a k-bit sequence number.
- This gives a range of sequence numbers of 0 through $2^k - 1$, and frames are numbered modulo 2^k, with a maximum window size of $2^k - 1$.
- The window size need not be the maximum possible size for a given sequence number length k.
- B acknowledges a frame by sending an acknowledgment that includes the sequence number of the next frame expected.
- This acknowledgment also implicitly announces that B is prepared to receive the next W frames, beginning with the number specified.
- This scheme can also be used to acknowledge multiple frames, and is referred to as **sliding-window flow control**.

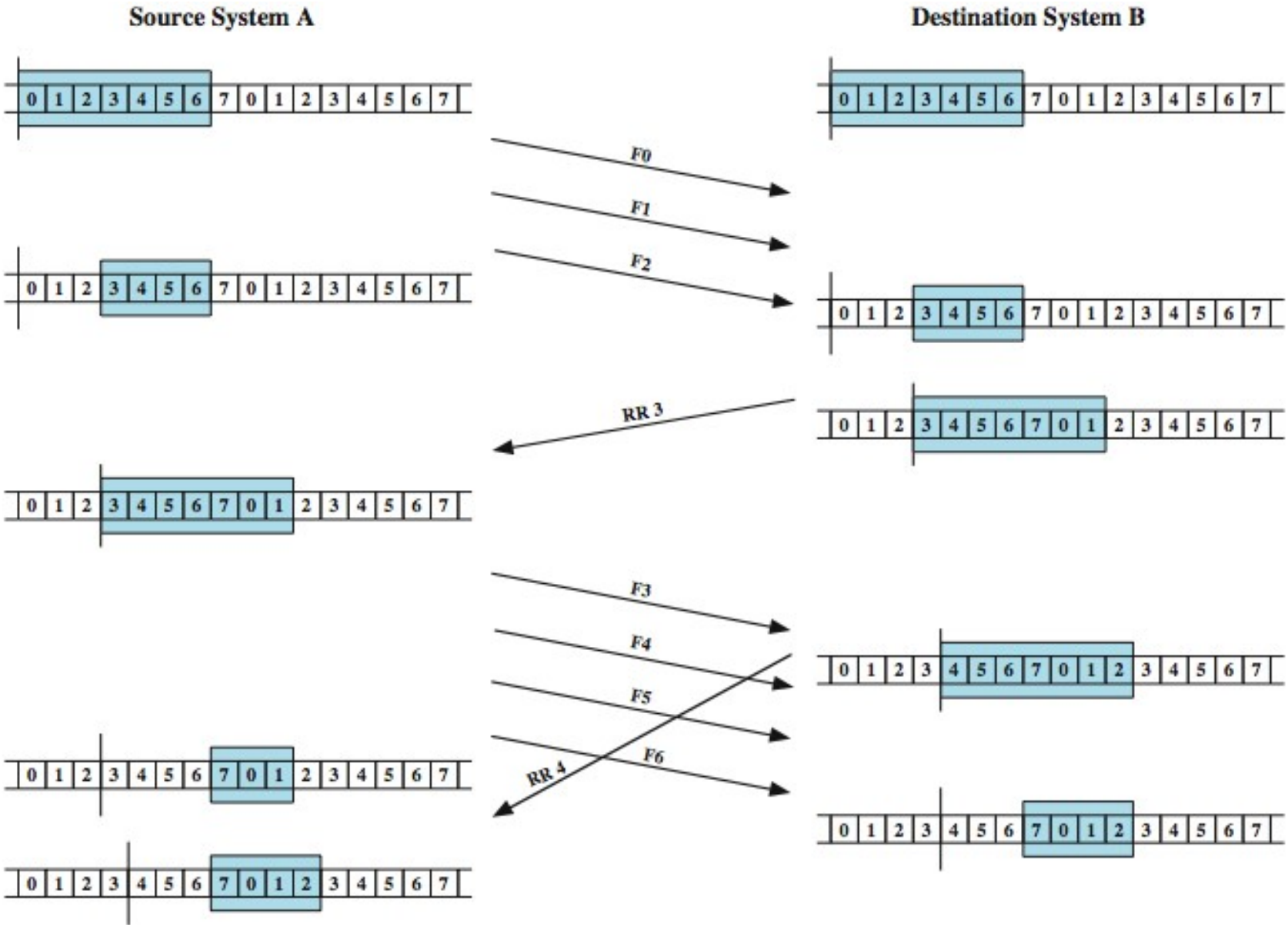
Sliding-Window Flow Control



Sliding-Window Flow Control

- Above figure depicts the sliding-window process. It assumes the use of a 3-bit sequence number, so that frames are numbered sequentially from 0 through 7, and then the same numbers are reused for subsequent frames.
- The shaded rectangle indicates the frames that may be sent; in this figure, the sender may transmit five frames, beginning with frame 0.
- Each time a frame is sent, the shaded window shrinks; each time an acknowledgment is received, the shaded window grows. Frames between the vertical bar and the shaded window have been sent but not yet acknowledged. As we shall see, the sender must buffer these frames in case they need to be retransmitted.
- Sliding-window flow control is potentially much more efficient than stop-and-wait flow control. The reason is that, with sliding-window flow control, the transmission link is treated as a pipeline that may be filled with frames in transit. In contrast, with stop-and-wait flow control, only one frame may be in the pipe at a time.

Sliding-Window Flow Control



Sliding-Window Flow Control

- An example is shown in above Figure.
- The example assumes a 3-bit sequence number field and a maximum window size of seven frames.
- Initially, A and B have windows indicating that A may transmit seven frames, beginning with frame 0 (F0).
- After transmitting three frames (F0, F1, F2) without acknowledgment, A has shrunk its window to four frames and maintains a copy of the three transmitted frames.
- The window indicates that A may transmit four frames, beginning with frame number 3.
- B then transmits an **RR (receive ready)** 3, which means "I have received all frames up through frame number 2 and am ready to receive frame number 3; in fact, I am prepared to receive seven frames, beginning with frame number 3."
- With this acknowledgment, A is back up to permission to transmit seven frames, still beginning with frame 3; also A may discard the buffered frames that have now been acknowledged.
- A proceeds to transmit frames 3, 4, 5, and 6. B returns RR 4, which acknowledges F3, and allows transmission of F4 through the next instance of F2.
- By the time this RR reaches A, it has already transmitted F4, F5, and F6, and therefore A may only open its window to permit sending four frames beginning with F7.

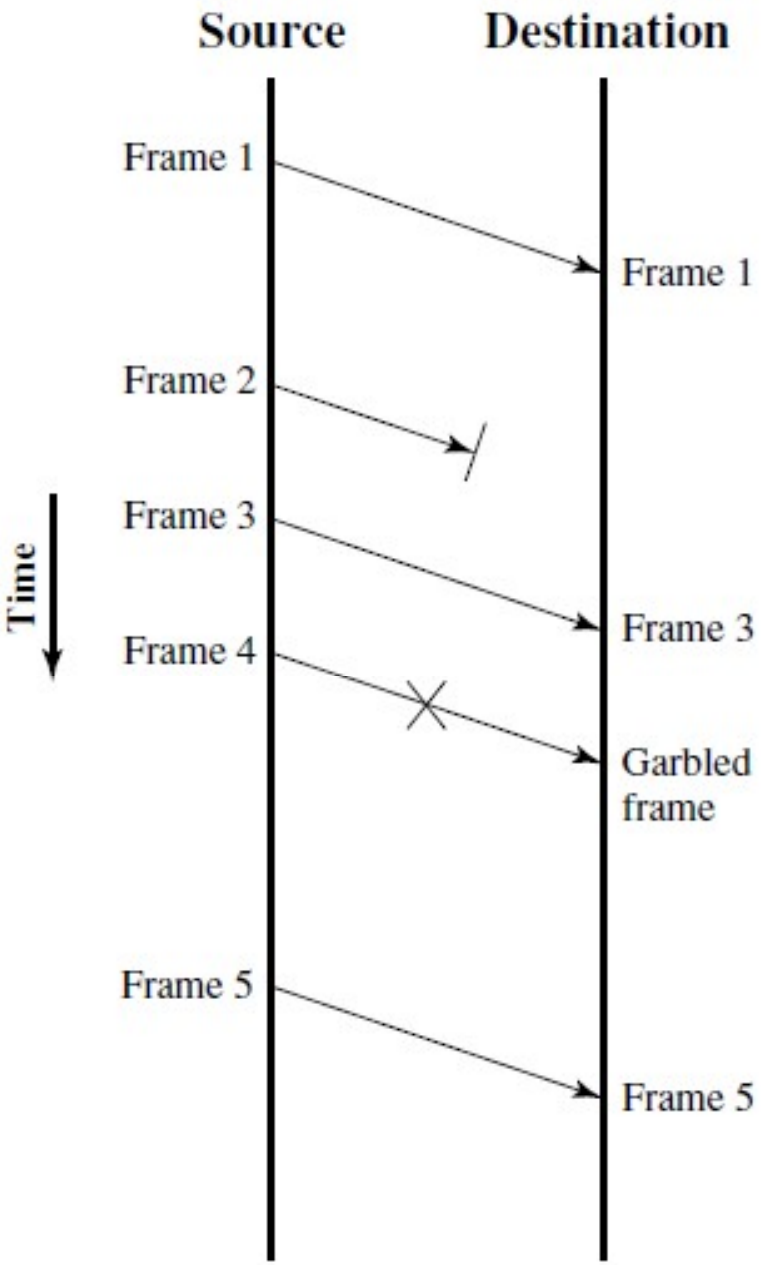
Sliding Window Enhancements

- Most data link control protocols also allow a station to cut off the flow of frames from the other side by sending a **Receive Not Ready (RNR)** message, which acknowledges former frames but forbids transfer of future frames. At some subsequent point, the station must send a normal acknowledgment to reopen the window.
- If two stations exchange data, each needs to maintain two windows, one for transmit and one for receive, and each side needs to send the data and acknowledgments to the other. To provide efficient support for this requirement, a feature known as **piggybacking** is typically provided.
- Each **data frame** includes a field that holds the sequence number of that frame plus a field that holds the sequence number used for acknowledgment.

Error Control

- **Error control refers to mechanisms to detect and correct errors that occur in the transmission of frames.**
- The model that we will use, which covers the typical case, was illustrated earlier in below figure
- As before, data are sent as a sequence of frames; frames arrive in the same order in which they are sent; and each transmitted frame suffers an arbitrary and potentially variable amount of delay before reception.

Error Control



(b) Transmission with losses and errors

Error Control

- In addition, we admit the possibility of two types of errors:
  - **Lost frame:** A frame fails to arrive at the other side. Ex- a noise burst may damage a frame so badly it is not recognized
  - **Damaged frame:** A recognizable frame does arrive, but some of the bits are in error (have been altered during transmission).
- The most common techniques for error control are based on some or all of the following ingredients:
  - **Error detection:** As discussed in earlier (Parity & CRC)
  - **Positive acknowledgment:** The destination returns a positive acknowledgment to successfully received, error-free frames.
  - **Retransmission after timeout:** The source retransmits a frame that has not been acknowledged after a predetermined amount of time.
  - **Negative acknowledgment and retransmission:** The destination returns a negative acknowledgment to frames in which an error is detected. The source retransmits such frames.

Error Control

- Collectively, these mechanisms are all referred to as **Automatic Repeat Request** (ARQ); the effect of ARQ is to turn an unreliable data link into a reliable one.
- Three versions of ARQ have been standardized:
  - Stop-and-wait ARQ
  - Go-back-N ARQ
  - Selective-reject ARQ
- All of these forms are based on the use of the flow control techniques.

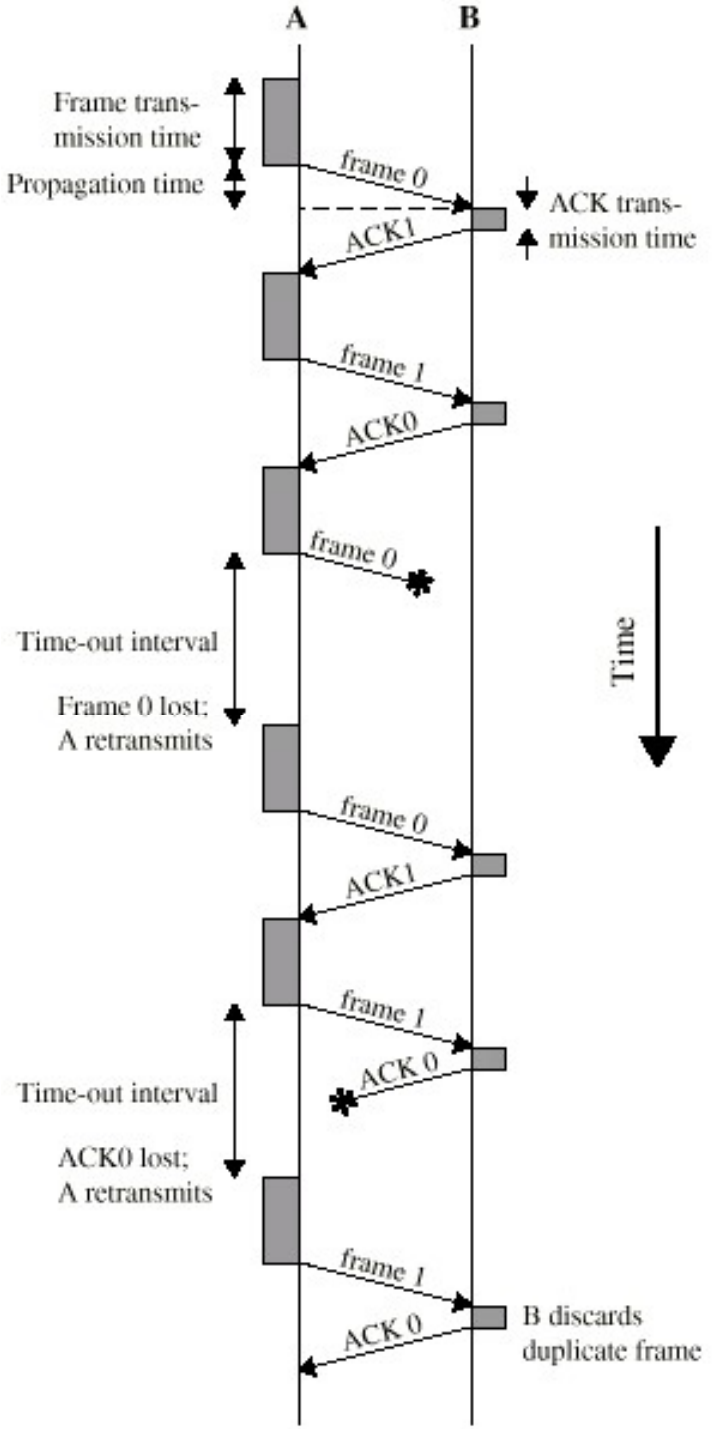
Stop and Wait ARQ

- Stop-and-wait ARQ is based on the stop-and-wait flow control technique outlined previously.
- The source station transmits a single frame and then must await an acknowledgment (ACK).
- No other data frames can be sent until the destination station's reply arrives at the source station.

Stop and Wait ARQ

- Two sorts of errors could occur.
  - **First**, the frame that arrives at the destination could be damaged. The receiver detects this by using the error-detection technique referred to earlier and simply discards the frame. To account for this possibility, the source station is equipped with a timer. After a frame is transmitted, the source station waits for an acknowledgment. If no acknowledgment is received by the time that the timer expires, then the same frame is sent again. Note that this method requires that the transmitter maintain a copy of a transmitted frame until an acknowledgment is received for that frame.
  - The **second** sort of error is a damaged acknowledgment. Consider the following situation. Station A sends a frame. The frame is received correctly by station B, which responds with an acknowledgment (ACK). The ACK is damaged in transit and is not recognizable by A, which will therefore time out and resend the same frame. This duplicate frame arrives and is accepted by B. B has therefore accepted two copies of the same frame as if they were separate. To avoid this problem, frames are alternately labeled with 0 or 1, and positive acknowledgments are of the form ACK0 and ACK1. In keeping with the sliding-window convention, an ACK0 acknowledges receipt of a frame numbered 1 and indicates that the receiver is ready for a frame numbered 0.

Stop and Wait ARQ



Stop and Wait ARQ

- Above figure gives an example of the use of stop-and-wait ARQ, showing the transmission of a sequence of frames from source A to destination B.
- The figure shows the two types of errors just described.
- The third frame transmitted by A is lost or damaged and therefore no ACK is returned by B. A times out and retransmits the frame.
- Later, A transmits a frame labeled 1 but the ACK0 for that frame is lost. A times out and retransmits the same frame. When B receives two frames in a row with the same label, it discards the second frame but sends back an ACK0 to each.
- The principal advantage of **stop-and-wait ARQ** is its **simplicity**. Its principal **disadvantage** is that stop-and-wait is an **inefficient mechanism**. The **sliding-window flow control** technique can be adapted to provide **more efficient line use**; in this context, it is sometimes referred to as **continuous ARQ.**

Go-Back-N ARQ

- The sliding-window flow control technique can be adapted to provide more efficient line use, sometimes referred to as *continuous ARQ*.
- The form of error control based on sliding-window flow control that is most commonly used is called go-back-N ARQ.
- In this method, a station may send a series of frames sequentially numbered modulo some maximum value. The number of unacknowledged frames outstanding is determined by window size, using the sliding-window flow control technique.
- While no errors occur, the destination will acknowledge incoming frames as usual (RR = receive ready, or piggybacked acknowledgment).
- If the destination station detects an error in a frame, it may send a negative acknowledgment (REJ = reject) for that frame, as explained in the following rules.
- The destination station will discard that frame and all future incoming frames until the frame in error is correctly received. Thus, the source station, when it receives a REJ, must retransmit the frame in error plus all succeeding frames that were transmitted in the interim.

Go-Back-N ARQ

- Suppose that station A is sending frames to station B. After each transmission, A sets an acknowledgment timer for the frame just transmitted.
- Suppose that B has previously successfully received frame (i – 1) and A has just transmitted frame i.
- The go-back-N technique takes into account the following contingencies:

1. Damaged frame
2. Damaged RR
3. Damaged REJ

Go-Back-N ARQ – Damaged Frame

- If the received frame is invalid (i.e., B detects an error, or the frame is so damaged that B does not even perceive that it has received a frame), B discards the frame and takes no further action as the result of that frame.
- There are **two** subcases:

a) Within a reasonable period of time, A subsequently sends frame (i+1). B receives frame (i+1) out of order and sends a REJ i. A must retransmit frame i and all subsequent frames.

Go-Back-N ARQ – Damaged Frame

b) A does not soon send additional frames. B receives nothing and returns neither an RR nor a REJ. When A's timer expires, it transmits an RR frame that includes a bit known as the P bit, which is set to 1. B interprets the RR frame with a P bit of 1 as a command that must be acknowledged by sending an RR indicating the next frame that it expects, which is frame i. When A receives the RR, it retransmits frame i. Alternatively, A could just retransmit frame i when its timer expires.

Go-Back-N ARQ – Damaged RR

- There are two subcases:

a) B receives frame i and sends RR (i+1), which suffers an error in transit. Because acknowledgments are cumulative (e.g., RR 6 means that all frames through 5 are acknowledged), it may be that A will receive a subsequent RR to a subsequent frame and that it will arrive before the timer associated with frame i expires.

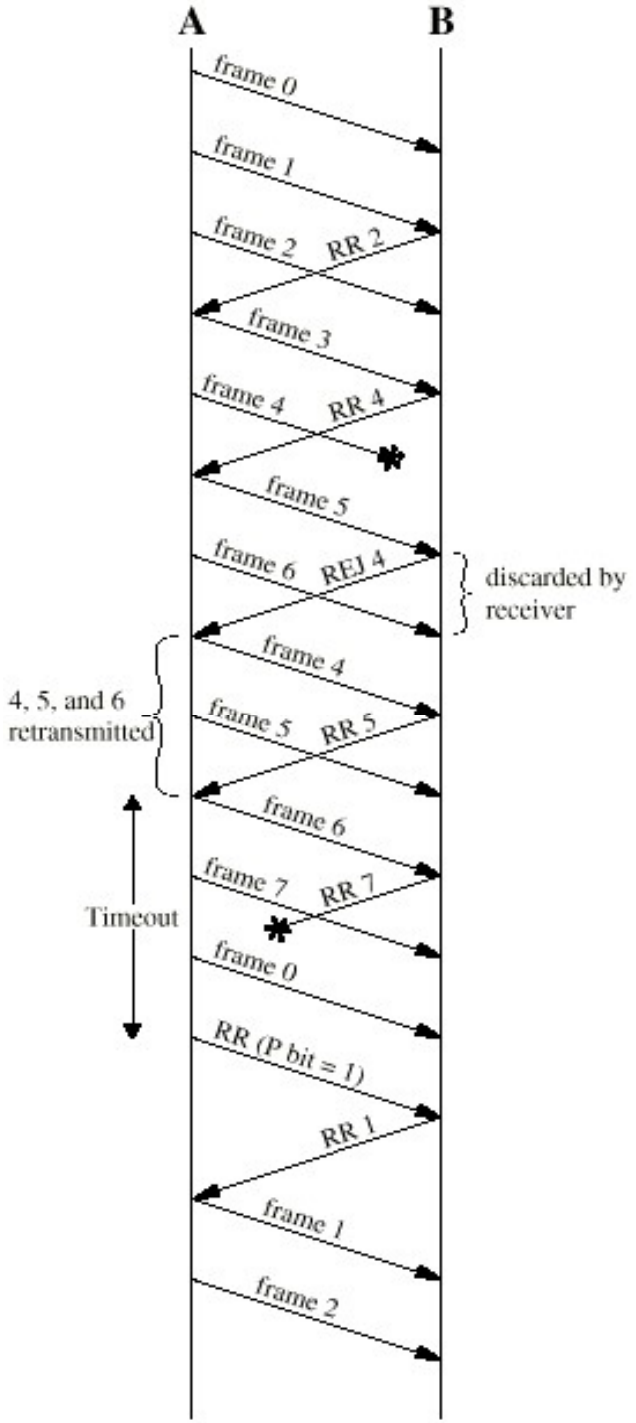
Go-Back-N ARQ – Damaged RR

b. If A's timer expires, it transmits an RR command as in **Damaged frame Case (b)**. It sets another timer, called the P-bit timer. If B fails to respond to the RR command, or if its response suffers an error in transit, then A's P-bit timer will expire. At this point, A will try again by issuing a new RR command and restarting the P-bit timer. This procedure is tried for a number of iterations. If A fails to obtain an acknowledgment after some maximum number of attempts, it initiates a reset procedure.

Go-Back-N REJ

- If a REJ is lost, this is equivalent to Damaged Frame case (b).

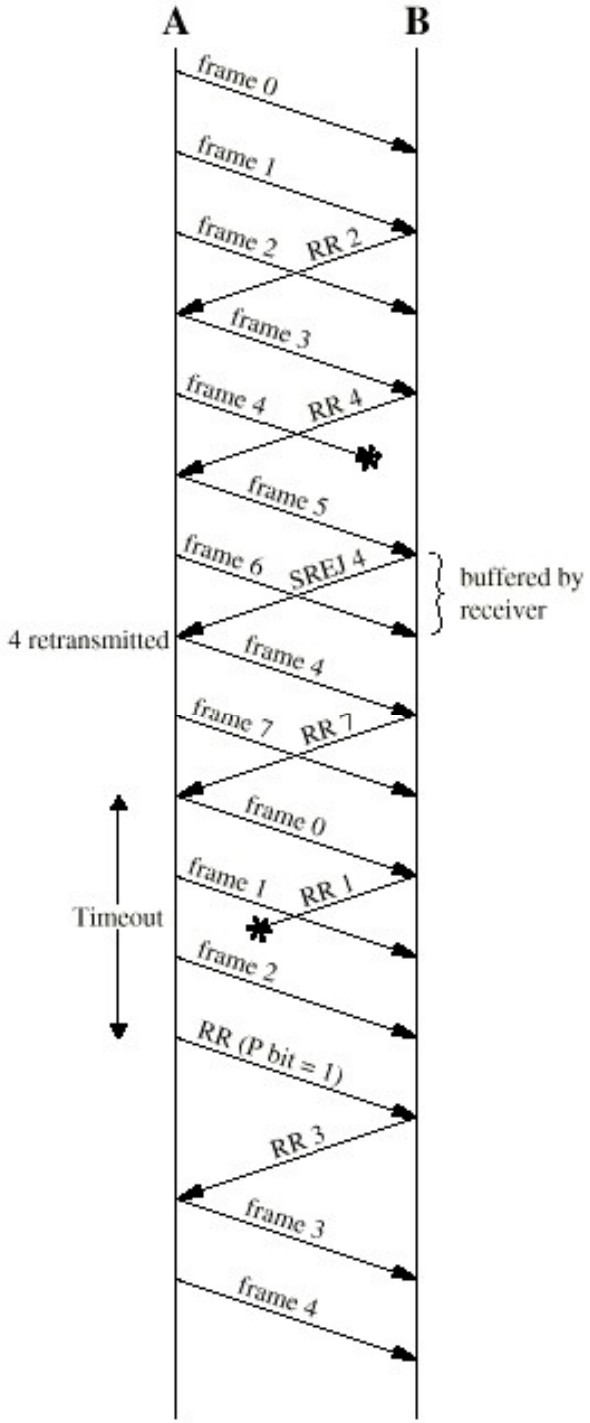
Go-Back-N ARQ


A
B
frame 0
frame 1
frame 2
RR 2
frame 3
frame 4
RR 4
frame 5
frame 6
REJ 4
discarded by receiver
frame 4
4, 5, and 6 retransmitted
frame 5
RR 5
frame 6
frame 7
RR 7
Timeout
frame 0
RR (P bit = 1)
RR 1
frame 1
frame 2

Selective-Reject ARQ

- With selective-reject ARQ, the only frames retransmitted are those that receive a negative acknowledgment, in this case called SREJ, or those that time out.
- Below figure illustrates this scheme. When frame 5 is received out of order, B sends a SREJ 4, indicating that frame 4 has not been received.
- However, B continues to accept incoming frames and buffers them until a valid frame 4 is received. At that point, B can place the frames in the proper order for delivery to higher-layer software.

Selective-Reject ARQ Diagram



Selective-Reject ARQ

- Selective reject would appear to be more efficient than go-back-N, because it minimizes the amount of retransmission.
- On the other hand, the receiver must maintain a buffer large enough to save post-SREJ frames until the frame in error is retransmitted and must contain logic for reinserting that frame in the proper sequence.
- The transmitter, too, requires more complex logic to be able to send a frame out of sequence. Because of such complications, select-reject ARQ is much less widely used than go-back-N ARQ.
- Selective reject is a useful choice for a satellite link because of the long propagation delay involved.