6 Real-Time Communication

Real-time applications are increasingly being implemented on distributed platforms and there are several reasons for their popularity. One important reason is that it is often cost-effective to have a distributed solution using many pieces of cheap hardware, rather than having a centralized, sophisticated, and costly machine. Further, many real-time applications are inherently distributed with different data sources (e.g., sensors) and data receivers (e.g., actuators) of a system placed at geographically separate locations. A distributed implementation in such a situation makes good design sense. Another point going in favour of distributed implementations is fault-tolerance which is very important for safety-critical applications. It is easier to provide fault-tolerance in distributed implementations compared to centralized systems. All distributed real-time systems are based on an underlying communication network. Such networks are expected to deliver messages in a timely fashion, and are often referred to as supporting real-time communication.

We can define real-time communication as one where an application can make specific quality of service demands such as maximum permissible delay, maximum loss rate, etc. on the underlying communication network; and once the network accepts a connection request, it guarantees the requested service quality.

Traditional network protocols such as Ethernet are designed for "best-effort" performance. A best-effort network strives to achieve good average performance, and makes no attempt to meet the individual quality of service requirements for different connections. These networks are intended for use in applications where long delays and high data loss under heavy load conditions are acceptable. Best-effort networks have very little restrictions on the quality of service they deliver, and, therefore, can use protocols that make best utilization of the network resources. It is needless to say that best-effort networks are insufficient for use in real-time applications. In real-time applications, deterministic delays and predictable performance issues take precedence over network utilization considerations.

With increasing automation of day-to-day life with a variety of gadgets and the popularity of embedded systems in industrial applications, the necessity of real-time communications has increased tremendously. This has been manifested in the proliferation of research efforts to support real-time communications. Many of the available literature address hard real-time communication. In hard real-time communication unless an absolute delay bound is met, the communication is assumed to have failed. Many of the industrial and embedded real-time applications require hard real-time communication. On the otherhand, in a firm real-time communication, if occasionally a message delayed, then the delayed message is merely discarded, without affecting the system to any great extent. Such types of communications are

usually required in multimedia applications. Soft real-time communication, on the other hand, is best-effort communication. This chapter focusses on the latest techniques available in the different types of real-time communication.

In this chapter, we first discuss a few examples of applications requiring real-time communications. Next, we discuss some basic concepts concerning computer networks and real-time communication techniques. Subsequently, we consider how real-time communication can be supported in multiple access networks. Finally, we discuss how real-time communication can be supported in packet-switched networks.

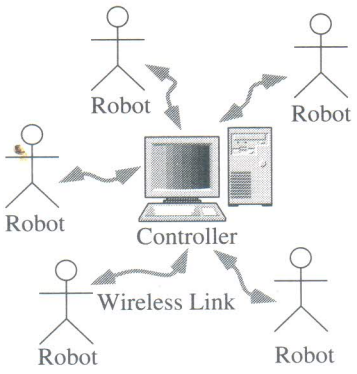
6.1 EXAMPLES OF APPLICATIONS REQUIRING REAL-TIME COMMUNICATION

In this section, we discuss a few applications whose satisfactory functioning is to a great extent dependent on the underlying network's capability to support real-time communication. We should bear these applications in mind while trying to understand various issues in real-time communications discussed later in this chapter. Traditionally, real-time communication technologies were used in distributed process control applications such as computer-controlled power plants, avionics, chemical process control, and automobile. However, current advancements in communication technologies have made it possible for supporting applications requiring real-time communication in many different areas such as manufacturing automation, video broadcast and Internet applications like online banking, stock trading, e-commerce, etc.

Manufacturing Automation: There are several types of manufacturing automation, leading to different scenarios of use of real-time communication in such applications. Here we discuss an example concerning a very simple scenario of manufacturing automation. However, we must remember that very different scenarios than those discussed here are possible.

In an automated factory a set of robots carry out manufacturing activities. Here, the network spans a small geographic area in a LAN environment. The robots communicate among themselves and with the controller (typically, a powerful computer) using a real-time communication protocol implemented on a wireless medium (see Fig. 6.1). The controller coordinates the activities of the robots. The messages that the robots communicate with the controller range from non-critical and non-real-time event logging information to highly critical and hard real-time control information. These communication should be treated differently by the underlying network for satisfactory performance of the application. Naturally, the underlying real-time communication system would be called upon to support these communication requests having widely varying service quality requirements.

Automated Chemical Factory: Consider a chemical factory in which an existing LAN set up is used for controlling and monitoring the parameters of their chemical plant on a real-time basis. The sensors transmit the sampled parameter values at periodic intervals to the computer that acts as the controller. The controller computes the corrections that may be necessary based on the sensed parameter values, and transmits commands to the actuators located inside the plant to carry out certain actions as and when required. These critical activities proceed along with other non-critical activities such as logging, e-mail handling, and surveillance, etc.



▲ **FIGURE 6.1**
Communication Among Robots in an Automated Factory

A real-time communication protocol guarantees the service quality of different categories of messages.

Internet-Based Banking Applications: In an Internet-based banking application, bank customers carry out their banking transactions using web browsers in the comfort of their homes. Though this is often considered to be very desirable, implementation of such an application requires use of sophisticated techniques and designs. For supporting such Internet banking applications, the underlying network needs to be extremely secure and reliable due to the sensitive nature of the financial transactions involved. These applications also have stringent delay requirements. Unless a transaction completes on time, the server may time-out, leading to rejection of user requests and user dissatisfaction.

Multimedia Multicast: In a multimedia multicast application, many receivers placed at various geographical locations receive multimedia information from a set of multimedia servers. The multimedia information is usually delivered in the form of streaming video and audio. Transmission and processing of such information is associated with firm real-time constraints. Another important issue that arises here is the efficient use of network resources—a multicast communication should not be implemented as a set of one-to-one (unicast) communications between the source and the different destinations.

Internet Telephony: VoIP is all set to revolutionize long-distance voice communication. VoIP stands for Voice over IP, where IP refers to the Internet Protocol that underlies all Internet communication. By using VoIP phones and technology one can make phone calls through the Internet. The benefit of this over traditional telephony is that as the actual voice traffic is carried over the Internet, VoIP communication would cost just a small fraction of what an actual telephone call costs. This would be much more expressed, especially over long distances. In a VoIP implementation, the network encrypts voice signals into data packets at the sender end, places it on the network, and decrypts it into voice signals at the receiver end. VoIP applications normally use simple microphones and computer speakers. But, use of IP telephones can provide an experience identical to normal telephoning. VoIP applications require firm real-time data transfer support.

6.2 BASIC CONCEPTS

In this section, we discuss a few basic concepts such as classification of real-time computer communication networks, quality of service parameters and traffic categorization.

6.2.1 Types of Networks

Three types of networks are mainly relevant in real-time communication: Controller Area Networks (CANs), Local Area Networks (LANs), and packet-switched networks. This classification is made on the basis of the size of the network and the communication technology deployed.

Controller Area Network: A Controller Area Network (CAN) is essentially a very small network. CANs are typically used to connect the different components of an embedded controller. The end-to-end length of a CAN is usually less than 50 meters. Since the propagation time of a CAN is very small, functionally a CAN behaves more like a local bus in a computer.

To understand the genesis of CAN and its operation, consider the present day automotive systems. The present day automotive electronics is fairly sophisticated, with automated support for several activities such as engine management, fuel injection, active suspension, braking, lighting, air-conditioning, security, and central locking. A considerable amount of information exchange among various automotive components is required when the engine is operational. The conventional method of networking the components in older models of cars was point-to-point wiring. This interconnection method was a straightforward evolution of the simple electrical systems used in cars where the different electrical components (motor, generator, light, battery, ignition system, etc.) were interconnected using point-to-point wiring. However, as the sophistication of the cars grew, use of such a simple scheme would have required several kilometers of wiring, adding to the cost of manufacturing, weight, complexity, and at the same time contributing to severely reduced reliability.

The limitations of fixed point-to-point wiring techniques in handling the demands of modern automated cars and other embedded applications gave rise to the development of CAN. A special requirement on CAN is effective handling of noise. Automotive components such as electric motors, ignition systems, and RF transmissions are heavy producers of noise. Another distinguishing feature of CAN is the 12 volt power supply that was mandated by the conventional 12 volt automotive power supply. CAN specifies only the physical and data link layers of ISO/OSI model, with higher layers left to the specific implementations.

Because of its robustness, CAN has expanded beyond its automotive origins and can now be found in diverse application areas such as industrial automation systems, trains, ships, agricultural machinery, household appliances, office automation systems, and elevators. Now CAN is an international standard under ISO11898 and ISO11519-2.

Local Area Network: A Local Area Network (LAN) is typically deployed in a building or a campus and is usually privately owned. For example, a LAN can be used to connect a number of computers within an organization to share data and other resources such as a files, printers, FAX services, etc. LANs typically operate at data rates exceeding 10 mbps and many present-day LANs (gigabit Ethernets) operate at 1 gbps. We provide a brief overview of LANs in Section 3.1. The available literature on LANs and their protocols is large, and the reader is referred to [27] for a comprehensive treatment of the subject.

Packet Switched Networks: Packet-switching refers to protocols in which messages are divided into relatively small units of data called packets before they are sent over the network. Each packet is then transmitted individually and a packet can even follow a route different from that taken by other packets to reach its destination. The individual packets are routed through a network based on the destination address contained within each packet. Once all the packets forming a message arrive at the destination, they are recompiled into the original message. Breaking down communication into packets allows the same data path to be shared among several users in the network. This type of communication between sender and receiver is known as connectionless and is in sharp contrast to the dedicated connections realized in connection-oriented networks. Packet-switched networks are usually wide area networks. A prominent example of a packet-switched network is the Internet.

The Internet, sometimes simply called "the Net," is a worldwide system of computer networks—a network of networks in which users at any one computer can, if they have permission, get information from any other computer using protocols such as ftp, http, etc; the user can even talk directly to users at other computers using VoIP. We must, however, be aware of the following two distinctions: the internet is any collection of separate physical networks, interconnected by a common protocol, to form a single logical network while the Internet is the worldwide collection of interconnected networks, which grew out of the ARPANET project [27]. It uses Internet Protocol (IP) to link various physical networks into a single logical network. The Internet began in 1962 as an attempt towards a resilient computer network for the US military and over time has grown into a global communication tool of more than a million computers connected to several tens of thousands of computer networks that share a common addressing scheme.

6.2.2 Quality of Service (QoS)

We had already pointed out that for their satisfactory operation, real-time applications need guarantees regarding the service quality from the underlying network. The service quality expected by an application from the underlying network is often expressed in terms of certain Quality of Service (QoS) parameters. Real-time applications usually have stringent requirements on the following QoS parameters: bandwidth, maximum transmission delay, delay jitter, loss rate, and blocking probability.

Delay: A successful delivery of a packet by a communication network in a real-time application not only depends on the intact receipt of the packet at the receiver end, but what also matters is the time at which it is received. If the time the network takes to deliver a packet exceeds the specified delay bound, then the application times out and can result in a failure in case of a hard real-time application. In a firm real-time application, the data received after the expiry of the specified delay bound would be discarded by the receiver. In case of a soft real-time application, exceeding the delay bound in delivery of messages would result in a degradation of the performance of the application.

Packet delay consists of three parts: propagation, transmission, and queuing delays. Propagation delay depends on the distance the packet travels and the medium in which it travels. Approximately, it is 5μ seconds per kilometer for the commonly used media. The per-hop transmission delay is given by packet size divided by the link bandwidth. The queuing delay is determined by the time that a packet waits in a queue before being transmitted. During network congestion, queuing delay is far greater than the other delay components.

Delay Jitter: Jitter is defined as the maximum variation in delay experienced by messages or packets in a single session. In other words, it is the difference between the maximum and minimum delays that messages might encounter. In a packet-switched network, for example, if the minimum end-to-end delay that a packet may experience during a call is 1 mSec, and the maximum delay is 10 mSec, then the delay jitter of the call is 9 mSec.

In a LAN using a collision-based protocol, jitter may be caused when there are variations in the network load. At high loads, jitter can considerably increase. On the other hand, in a packet-switched network, jitter may arise due to the fact that as packets travel in the network, they may suffer different amounts of queuing delays at different nodes. Also, packets may travel in different paths having different number of hops, resulting in jitter. Jitter is unacceptable in many applications. In many hard real-time applications, presence of jitter exceeding the specified bound can make the system fail. Jitter is very undesirable in many firm real-time applications as well. For example, in a video conferencing application, jitter can show up as a picture frame remaining still for a certain time.

Buffers at the receiver-end can be used to control jitter in many cases. In a packet switched network for example, removing jitter might involve collecting packets in a buffer at the receiver end, and holding them long enough to allow the slowest packets to arrive in time to be processed in correct sequence. The amount of buffer space required at a receiver can be determined from the peak rate of the arriving messages and the delay jitter. The size of the buffer required at the receiver-end would in fact be given by *peak rate* $\times$ *delay jitter*.

Example 6.1

Consider a single video source transmitting 30 frames per second to a certain receiver. Each frame contains 2 mb of data. The jitter in the network is known to be 1 sec. Compute the amount of buffer space required at the receiver to compensate for the jitter.

Solution. The required buffer size at the receiver is given by (peak rate * jitter) = 2 mb * 30 = 60 mb.

From the above example, it can be easily seen that a tight delay jitter bound can help reduce the buffering requirement at the receiver end.

Bandwidth: This parameter indicates the rate at which a connection would be serviced by the network. Bandwidth determines an application's maximum throughput, and in some cases determines the bounds on the end-to-end delay. Sufficient bandwidth is required to sustain the required throughput for an application. In a CAN or LAN environment, the delay experienced by a message is inversely proportional to the bandwidth at which it would be served. That is, the higher the allocated bandwidth, the lower is the delay. However, in a packet switched network such as the Internet, this is only partly true—the transmission delay in Internet due to finite available bandwidth is often insignificant compared to the queuing delays that packets undergo at the switches.

Loss Rate: Loss rate denotes the percentage of all transmitted packets that may be lost during transmission. Packets in a connection may be treated as lost due to a variety of reasons such

as delay-bound violation, delay jitter-bound violation, buffer overflow, and data corruption. Data corruption depends to a large extent on the type of the media being used. Data corruption is insignificant in fibre optic media and can be ignored, whereas data corruption is considerably large in wireless media. Different applications are sensitive to loss rate to different extents. Process control applications typically require zero loss rate, while applications such as multimedia can tolerate a certain amount of data loss.

Blocking Probability: Blocking probability is the probability of a new connection being rejected by the admission control mechanism of a network. A call may be rejected under heavy load situations or want of resources such as bandwidth.

The requirements of different real-time applications with respect to the above QoS parameters may be very different. For example, certain applications such as non-interactive television and audio broadcasting require stringent bounds on jitter but are rather insensitive to delay. On the other hand, sensor data processing in a fly-by-wire aircraft is very sensitive to delay—often only sub-millisecond delays in receipt of sensor signals by the controller is acceptable. In contrast, traditional computer communication applications such as file transfer, electronic mail, and remote login are non-real-time applications. The QoS parameters for such applications are very different from what we discussed for real-time applications. For non-real-time applications, bounds on average packet delay and average throughput are important, and such as worst-case packet delay, jitter, or worst-case throughput are not. Non-real-time applications, however, have strict reliability requirements, since any corruption of parts of a document can make the entire document unusable. Indeed, much of the complexity of the traditional network protocols arises from the need for loss-free communication between non-real-time applications.

Multimedia applications are not affected so much by delay as jitter and loss rate. For example, video transmissions are very sensitive to packet losses since these show up as flickers or glitches on the display screen. In voice applications, the quality of the voice degrades rapidly with loss rate and jitter. Voice data packet sizes are, therefore, deliberately kept small to minimize the packetization delays and to limit the effect of packet losses. The standard 48-byte cell size for ATM network, for example, was chosen primarily for the benefit of the voice applications.

6.2.3 Traffic Categorization

It is important to categorize a traffic source based on its traffic generation characteristics. Such categorization becomes necessary, since a network can guarantee the required quality of service only when these traffic characteristics are specified. The traffic generated by a source can be categorized based on the rate at which data are generated by the source for transmission on a network. The following are three important categories of traffic that are commonly encountered.

CBR Traffic: CBR traffic arises due to Constant Bit Rate data generation by a source. Data generation and transmission involved in hard real-time applications are often CBR traffic. For example, periodic data generated by sensors are CBR type of traffic. In this case, fixed sized messages are transmitted over fixed intervals.

VBR Traffic: VBR traffic, as the name suggests, consists of different rates of data transmission at different times. VBR sources can be of several types. A common type of VBR traffic generated by a data source alternates between a period in which fixed sized packets are generated with deterministic spacing and an idle period. An example of such VBR traffic is

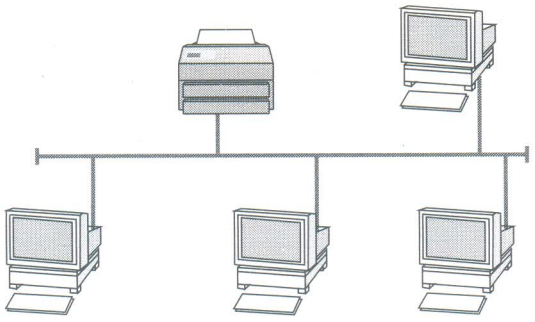
the compressed audio signals generated by speech signals. Typically, in compressed speech signals, to reduce the size of voice traffic, no data is generated during periods of silence. This results in on-off sources where fixed sized data is generated during on periods and no data is generated during idle periods. While silence suppression can produce significant reductions in the bandwidth requirements, this can only be achieved at the cost of having to reconstruct the original traffic at the receiver. Another example of VBR traffic is compressed video signals where variable sized data is generated periodically. Redundancy in digitized video data is usually very high. It, therefore, is often compressed using algorithms such as MPEG, before being transmitted over a network. In such types of VBR traffic, the source periodically submits variable sized packets to the network. It should be clear that variable bit rate transmission is intended to make better use of available bandwidth in applications requiring transmission of video or audio signals.

Sporadic Traffic: Sporadic traffic consists of a special type of variable sized packet transmissions. In sporadic traffic, the packets are generated in bursts followed by long periods of silence. Sporadic traffic is thus a special type of VBR traffic which occurs under very special circumstances. For example, the traffic consisting of certain command, control, and alarm messages generated in response to some exception conditions belong to this category. Consider a fire alarm. When a fire condition is detected, a large number of alarm, command and response messages are generated. The sudden peak load due to alarms is usually called an *alarm avalanche.*

6.3 REAL-TIME COMMUNICATION IN A LAN

Many hard real-time applications span small geographical areas. Examples of such applications include automated manufacturing systems, industrial process control applications, high-speed data acquisition systems. In such applications, Local Area Networks (LANs) usually turn out to be the preferred choice for networking. In these situations, networks such as dedicated point-to-point links or circuit-switching among the different nodes (computers) in the system turn out to be very inefficient and costly. A circuit-switched network simply sets aside a fixed portion of the network bandwidth according to the estimated peak bandwidth requirement of each application. However, such a technique does not work well for those real-time traffic that are inherently bursty in nature. In fact, many categories of real-time traffic are bursty in nature. For such traffic, unless the idle times can be filled by non-real-time traffic, circuit-switching would lead to very low effective bandwidth utilization. As a result, point-to-point or circuit switched networks are not very popular in many real-time applications.

In a LAN, there is a single shared channel and only one node can transmit at any time. The exact time when a node can transmit on the channel is determined by the access arbitration policy of the network. The transmission control policy determines how long a node can transmit. These two policies together are called the access control technique and form the Media Access Control (MAC) layer protocol. In other words, the MAC protocol in a LAN consists of two parts: an access arbitration part that determines when a node can use the channel and a transmission control part that determines for how long a node can continue to use the channel once it starts using it. Before we discuss how real-time communication can be achieved in a LAN, let us first review some basic aspects of LANs that are crucial to understanding our subsequent discussions.



▲ **FIGURE 6.2**
A Bus Interconnection Network

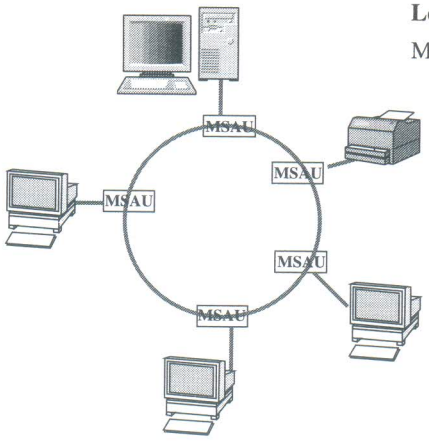
6.3.1 LAN Architectures

Two major LAN architectures are being used: the bus and the ring architectures [13, 27]. These two architectures use different access control techniques. We first briefly discuss these two LAN architectures, and then discuss the associated access control techniques.

Bus Architectures: In bus-based architectures, nodes are connected to the network cable using T-shaped network interface connectors as shown in Fig. 6.2. Terminating points are placed at each end of the network cable. There is a single shared channel (bus) for which the transmitting nodes contend. In a bus architecture, nodes communicate using broadcasting. The most commonly used protocol for access control in traditional bus networks is the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) [27]. CSMA/CD networks are also called multiple access networks. In CSMA/CD networks, when two or more nodes transmit packets simultaneously, the transmissions overlap in time and the resulting signal gets garbled. Such an event is called a collision. A collision entails retransmission of the corrupted data.

In CSMA/CD networks, any node can at any time sense the channel to determine whether the channel is idle. A node transmits a packet only if it senses the channel to be idle. But, this does not guarantee that there will be no collisions. Several nodes might sense the channel to be idle at the same time instant and start transmitting simultaneously, resulting in a collision. The transmitting nodes can detect a collision when it occurs. Therefore, while transmitting a packet, a node would check for a collision and would immediately stop transmitting, if it detects one. It should, therefore, be clear that larger the propagation delay of a network, larger is the probability of collisions of packets.

Ethernet is a LAN standard based on CSMA/CD access control. CSMA/CD protocol does not define a collision resolution protocol on its own. Ethernet uses Binary Exponential Back-Off (BEB) algorithm for collision resolution. Due to its ubiquity, high speed, simplicity and low cost, Ethernet has over the years emerged as one of the most preferred LAN protocols. It is, therefore, not surprising that many attempts have been made in the past to develop protocols based on Ethernet to support real-time communication. However, as far as real-time communications are concerned, the logical ring architecture possesses significant advantages over Ethernet due to its inherent deterministic access arbitration mechanism in contrast to the collision based mechanism



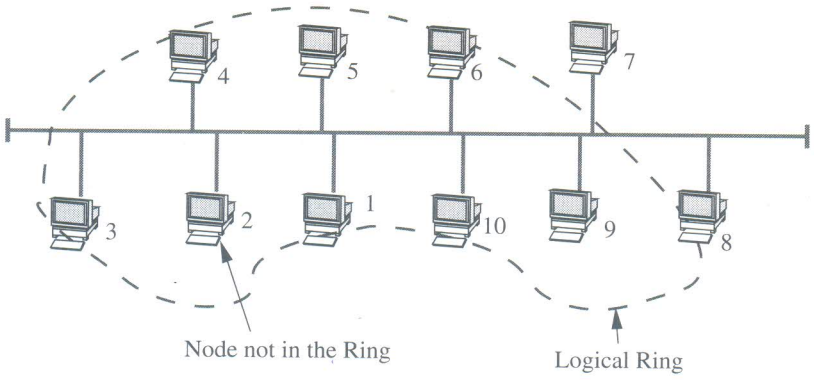
▲ **FIGURE 6.3**
A Ring Network

of Ethernet. In a collision-based network, under high load situations the number of collisions per unit time would increase very rapidly with load, leading to increased retransmissions rapid drop in throughput and rise in delay. As a result, in Ethernet the delay in message transmission increases rapidly as the traffic increases. This puts Ethernet-based networks at a disadvantageous position as far as real-time applications are concerned.

Ring Architectures: A ring architecture has schematically been shown in Fig. 6.3. Nodes in Fig. 6.3 have been shown to be connected to the network using MSAUs. An Multistation Access Unit (MSAU) is a hub or concentrator that connects a group of computers ("nodes" in network terminology) to the ring. The nodes are placed along the ring. The nodes transmit in turn. Each node usually transmits for a certain predetermined period of time. Therefore, packet transmission delays become predictable and can be made sufficiently small as per requirement. As a result, ring-based architectures are often preferred in real-time applications.

However, the ring architecture suffers from a few important problems. First, any break in the ring can bring the whole network down. This makes reliability of ring networks a major concern. Further, ring is a poor fit to the linear topology normally found in most assembly lines and other applications. This made researchers to look for alternative technologies which can have the advantages of both the bus and ring architectures. This led to the development of the token bus architecture. A token bus is a bus-based architecture (see Fig. 6.4), where the stations on the bus are logically arranged in a ring with each station knowing the address of the station to its "left" and "right."

When the logical ring is initialized, the highest numbered station gets a chance to start its transmission. After transmitting for a predetermined duration, the station passes the transmission permission to its immediate neighbour (left or right as per the convention adopted) by sending a special control frame called a *token*. The token propagates around the logical ring. At any time, only the token holder is permitted to transmit packets. Since only one station at a time holds the token in a ring network, collisions can not occur. An important point that must be kept in mind is that the physical order in which the stations are connected to the cable need not be the same as the



▲ **FIGURE 6.4**
A Logical Ring in a Token Bus

order of the stations in the logical ring. This is because a cable is inherently a broadcast medium. Each station would receive every frame transmitted, and discard those that are not addressed to it. After a node exhausts its assigned time slot for transmission, it hands over the token to its logical neighbour. For this, it transmits a special token, specifically addressed to its logical neighbour in the ring, irrespective of whether that station is physically adjacent to the concerned node or not. It is also worth noting that many stations on the network may not be in the ring. For example, the stations 2, 5, 7, and 9 in the Fig. 6.4 are not on the ring at all. The MAC protocol in a token ring network should support adding stations to, and removing stations from the logical ring [27].

6.4 SOFT REAL-TIME COMMUNICATION IN A LAN

Soft real-time communication networks are not expected to provide any absolute QoS guarantees to the applications. They only ensure prioritized treatment for real-time messages, so that the message deadline miss ratio (for real-time messages) can be kept to a minimum and the soft real-time messages can be provided statistical guarantees on delay bounds.

Soft real-time protocols usually assume that both soft real-time and non real-time messages would be transmitted over the network. Soft real-time traffic is assumed to be generated by CBR and VBR sources. Soft real-time message rates are assumed to be very low compared to the channel capacity. Non-real-time messages and certain soft real-time messages may arrive aperiodically in bursts. In presence of bursts, it becomes very difficult to sustain the guarantees to the real-time traffic unless specific arrangements are made. Therefore, the bursts need to be smoothened out for meeting the statistical guarantees on deadline bounds for real-time messages. In the following, we discuss a few important techniques available for this purpose.

A Fixed-Rate Traffic Smoothing Algorithm: Kweon and Shin developed a fixed-rate traffic-smoothing algorithm [15] that is based on the considerations of limits on the transmission capacity of a network. From the limits on the transmission capacity, the input limit for each node in the system is derived. The traffic smoother, which is placed between the MAC layer and TCP/IP layer, smooths a non-real-time stream so that guarantees to real-time messages would not be violated. The traffic-smoothing technique used is a leaky bucket algorithm called Credit

Bucket Depth (CBD). It has two important parameters: CBD and Refresh Period (RP). These two parameters are fixed statically. CBD is the maximum number of credits that are added to the bucket at every refresh. It is also the maximum number of credits that a bucket can hold. RP is the refresh period with which the bucket is replenished with new credits. The ratio CBD/RP is the average guaranteed throughput for non-real-time messages. The number of credits present in the bucket at any time is denoted as the Current Network Share (CNS). A waiting message is taken up for transmission, if the message size is smaller than CNS. When the number of available credits (CNS) is positive, but is smaller than the the size of a message to be transmitted, credits are allowed to be borrowed. So, it is possible that the balance of credits can become negative at times. The CNS may become negative, if credits have been borrowed. At every refresh, the number of credits in the bucket (CNS) is updated as follows: CNS = min(CNS + CBD, CBD). That is, CBD amount of credits are replenished, subject to the limit that CNS does not exceed CBD. When a non-real-time message arrives at a node for transmission, the smoothing mechanism carries out the following steps:

```
if (CNS > 0){
        CNS = CNS - message.number of Bytes;
        /*  message.number of Bytes is the size of the message */
        send message for transmission;}
else    {
        hold message in buffer until CNS > 0;}
```

The disadvantage of the fixed-rate traffic smoothing technique is that it is not very flexible as once the delay requirements of the real-time messages are known, the network-wide input limit is fixed. The transmission rate of the non-real-time sources is, therefore, limited and gets reduced as the number of nodes increases on the LAN. Consequently, an increase in the number of stations, leads to a decrease in traffic generations limits of the stations. This shows up as a corresponding decrease in throughput of non-real-time traffic. This technique is also very pessimistic since it is based on worst-case calculations on the total real-time traffic arrival rates in the system.

Adaptive Traffic Smoothing: Kweon and Shin have proposed an adaptive traffic smoother that addresses some of the shortcomings of the CBD algorithm [14]. The main idea behind adaptive traffic smoothing is that in order to provide a reasonable throughput for non-real-time messages, in the presence of VBR real-time traffic sources, the non-real-time traffic transmission rate can be allowed to adapt itself to the load conditions of the underlying network. That is, the nodes are allowed to increase their transmission limits, if the utilization of the network is low. On the other hand, when the utilization of the network becomes high, the nodes transmitting non-real-time messages are made to decrease their transmission limits. In order to implement such a rate adaptive traffic smoother, which would meet the delay requirements of real-time packets and at the same time provide improved average throughput for non-real-time packets, the following two problems must be resolved:

- How to detect a change in network utilization?
- How to adapt the transmission limits to a detected change in network utilization?

The solution that has been proposed is that the number of collisions per unit time can be used as a measure of network utilization. This proposal is based on the observation that at higher network utilizations, more collisions can be expected. This is a simple and approximate

way to determine network utilization at any time. In the event of a collision, the credit bucket is immediately emptied causing suspension of non-real-time packets, except for the packets that are already under transmission. This increases the chance to deliver the real-time packets generated from other nodes within the specified delay bounds, and prevents delays caused by bursts of non-real-time packets generated from this node. For this reason, packet collision is used as a trigger to decrease the throughput as well as to deplete the current credits.

The station input limit CBD/RP can be adapted by either changing the CBD or the RP parameter. The station input limit is increased periodically by decreasing RP periodically by a fixed number, if no collisions have been observed recently. Of course, the lower bound of RP is fixed by some predefined value RP_{min}. When a collision is detected, all credit buckets are depleted and RP is doubled. The upper bound of RP is fixed by a certain predefined value RP_{max}.

Experimental results have shown that by using the adaptive traffic smoothing scheme, message deadline miss ratios (for real-time messages) can be kept well within an acceptable range, for arbitrary arrival rates for non-real-time messages. It should, however, be remembered that this technique though efficient, fails to provide deterministic real-time communication. It is, therefore, suitable only for soft real-time systems and can not be satisfactorily used to support hard real-time communications.

6.5 HARD REAL-TIME COMMUNICATION IN A LAN

We had discussed several example applications in Section 6.1, for which hard real-time communication in a LAN environment becomes necessary. Hard real-time applications often involve transmission of CBR traffic such as periodic sensor signals. In hard real-time applications, the network utilization is deliberately kept low since predictability of network delay takes precedence over network utilization aspects. However, to improve the network utilization, soft and non-real-time traffic are usually allowed to be transmitted in the intervals during which hard real-time messages are not present.

In a LAN, hard real-time communications are normally supported using any of the following three classes [16, 19] of protocols: global priority-based, calendar-based, and bounded-access protocols. We first discuss some important features of these three classes of hard real-time communication protocols. Subsequently, we discuss a few important protocols belonging to these three classes.

Global Priority Protocols: In a global priority protocol, each message in the network is assigned a priority value. This MAC layer protocol tries to ensure that at any time the channel is serving to the highest priority message in the network. This opens up the possibility of using RMA or EDF for scheduling messages in a network deploying a global priority-based protocol. As discussed in Chapter 2, RMA and EDF are optimal static and dynamic priority task scheduling protocols, respectively, and are very popular with real-time application developers. However, the following two problems arise while trying to implement RMA or EDF-based message scheduling algorithms in a network using a global priority-based scheduling protocol:

- After the transmission of a packet starts, it can not be stopped halfway and the remaining bits of the packets transmitted at a later time. In other words, unlike the way tasks could be preempted from CPU usage, packet transmissions can not be meaningfully preempted from channel usage.

- A global priority protocol can not instantaneously determine the message that has the highest priority. Since messages originate at different nodes, a centralized knowledge of messages that are currently eligible to be transmitted is absent. As we shall see in our subsequent discussions, what can be obtained in practice would have to be based on somewhat outdated information.

Both these problems cause violation of some of the basic premises that were made for RMA (and EDF) to optimally schedule tasks. Therefore, if RMA (or for that matter EDF) is used to schedule messages on a network deploying a global priority protocol—the two above mentioned problems would restrict the achievable schedulable utilization of the channel to very low values.

Bounded Access Scheduling: In bounded access scheduling, messages are provided real-time guarantees by bounding the access times of every node to the channel. This ensures that the time for which a packet may have to wait before being transmitted is bounded. Bounded access scheduling protocols fix the time for which a node is permitted to transmit its messages. The individual nodes use a local scheduling algorithm to determine the order in which packets queued up at the local node are taken up for transmission.

Calendar-based Scheduling: A calendar-based protocol, as the name suggests, maintains a calendar that indicates which node is permitted to transmit during which time period. A copy of the calendar is maintained by every node. Traffic sources can reserve time interval for packet transmission by broadcasting. When a node needs to transmit a message for which no reservation was made, it finds a free slot by consulting its local copy of the calendar and reserves the required time interval by broadcasting the reservation information to all nodes. A calendar-based protocol is simple, efficient, and works very well when all messages in the system are periodic and predictable.

In the following, we discuss some important protocols from each of these three categories of protocols.

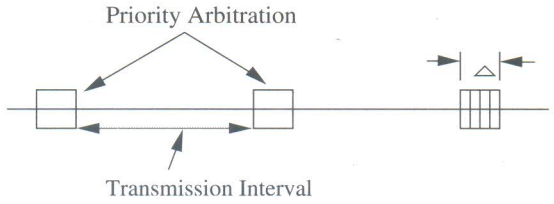
6.5.1 Global Priority-Based Scheduling

A few important global priority arbitration based protocols are discussed in the following.

Countdown Protocol: In this protocol, the time line is divided into fixed sized intervals called slots. At the start of every slot, priority arbitration is carried out to determine the highest priority message in the network. As soon as priority arbitration is complete, the node having the highest priority message is allowed to transmit. In other words, this scheme involves periodic priority arbitration followed by message transmission as shown in Fig. 6.5.

Priority arbitration is achieved as follows. Over each slot, every node that has a pending message, transmits the priority value of its highest priority pending message with msb first as shown in Fig. 6.6. Since simultaneous transmission follows *or* logic, a node that transmits a 0 and receives a 1, knows that there is at least one node that is having a higher priority pending message, and drops out of the contention. The node that transmits last without any collision, can conclude that no nodes have any higher priority messages, and can begin its transmissions.

An important parameter in the efficient working of this protocol is the slot size, Therefore, the slot size needs to be carefully selected. Each slot duration is usually made equal to



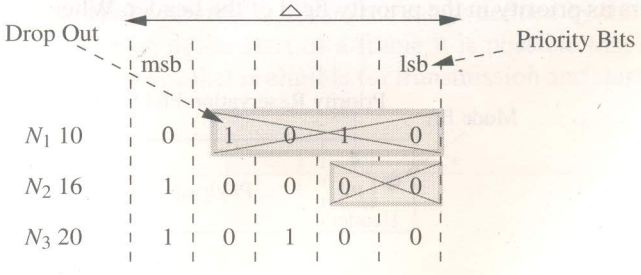
▲ **FIGURE 6.5**
Priority Arbitration and Transmission Intervals

the end-to-end propagation delay of the medium. If the slot size is made any smaller than this, then the priority arbitration scheme would not work since even when a collision occurs, it would not be detected. A slot size larger than this would lead to an increase in channel idle time during every slot.

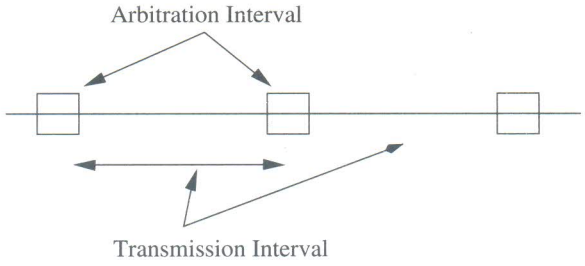
To illustrate the working of this protocol, transmission of bits by different nodes in an arbitration example has schematically been shown in the Fig. 6.6. In Fig. 6.6, three nodes N_1, N_2, and N_3 are participating in the arbitration process. The priorities of messages at the nodes N_1, N_2, and N_3 are 10, 16, and 20, respectively. As shown, node N_1 drops out after transmitting the first bit as it transmits a 0 but listens a 1. Similarly, the node N_2 drops out after transmitting the third bit when it transmits a 0 and listens to a 1. Finally, after transmitting the last bit, node N_3 concludes that it has the highest priority message and begins its transmission.

Virtual Time Protocol: In this protocol, a node uses the state of the channel to reason about the pending packets residing at other nodes. Each node with a packet to send waits for an interval of time that is inversely proportional to the priority of the highest priority message it has. This protocol assigns priority to nodes. The priority of a node is equal to the highest priority message that it has. That is, the lower the priority of the highest priority message that a node has, the longer it waits. At the expiration of the waiting time of a node, it senses the status of the channel. If the channel is busy, then this would imply that a higher priority message is being transmitted and it would need to wait until an idle period. Otherwise, it starts to transmit.

What should be the difference in wait times of two nodes which have messages whose priorities differ by one? Because of the propagation delay in the network, a node can not



▲ **FIGURE 6.6**
Priority Arbitration Example



▲ **FIGURE 6.7**

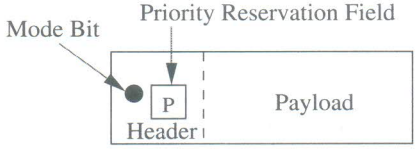
Priority Arbitration in Virtual Time Protocol

instantaneously detect when another node starts to transmit during arbitration. As a result, unless the wait times of two nodes differ by at least as much as the propagation time, the following problem would occur. Assume that two nodes N_1 and N_2 have their priorities differing by one. After N_1 starts transmitting, if N_2 waits for any time shorter than the propagation time, then it can not detect the transmission of N_1 and would start transmitting. This would lead to a collision and result in incorrect priority arbitration. Thus, propagation time effectively bounds the difference between the wait times of two nodes that have consecutive priorities (see Fig. 6.7). However, a wait time much larger than the propagation time would lead to channel idle times and underutilization of the channel.

IEEE 802.5: IEEE 802.5 is a priority-based token ring protocol. In this protocol, the header of the token contains two fields: a reservation field and a mode field (see Fig. 6.8). The token alternates between two modes: a reservation mode and a free mode. A node can at any time determine the mode of the token by examining the mode bit in the header of the token. The messages to be transmitted are split into frames. The token may contain a frame as a payload as shown in Fig. 6.8.

To be able to support different classes of network traffic, IEEE 802.5 supports assigning priorities to messages. The priority of the message that is being transmitted is recorded in the reservation field of the header.

Packet transmissions occur in the reservation mode. The priority of a message that is being transmitted at any time is registered in the reservation field of the header of the token as shown in Fig. 6.8. As the token passes through the ring, every node with pending messages inspects the reservation field in the token header. A node having a higher priority message than that is being transmitted, registers its priority in the priority field of the header. When the token returns to the



▲ **FIGURE 6.8**

Structure of a Token in IEEE 802.5

sending node, it observes the reservation made by another node, puts the token in free mode and releases it. As the free mode token passes through the ring, the node that made the reservation seizes the token, puts it into the reservation mode, and starts transmitting.

Two important results about this protocol are presented in the following two theorems.

THEOREM 6.1 *The minimum time required to complete transmission of a frame using IEEE 802.5 protocol is max(F, θ), where F is the frame transmission time and θ is the propagation time.*

PROOF A node does not start transmitting the next frame until:

- It has completed transmission of the last bit of the frame.
- It has received the header of the transmitted token back.

The first activity takes F time units, which is the frame transmission time. The second activity takes θ time units, which is the propagation time. Therefore, the transmission time is the higher of the two values, which is $max(F, \theta)$.

Theorem 6.1 indicates that when a packet is being transmitted, the minimum time before which transmission of the next packet can not start is bounded by $max(F, \theta)$.

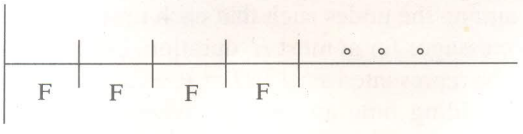
THEOREM 6.2 *A higher priority packet might undergo inversion for at most 2 × max(F, θ) time units.*

PROOF: A higher priority message undergoes inversion, until:

- The reservation mode completes.
- The node receives the token in free mode.

Each of the above step takes $max(F, \theta)$ time units to complete (by Theorem 6.1). Therefore, the total time a higher priority packet may have to wait before being transmitted is given by $2 \times max(F, \theta)$.

Window-Based Protocol: This is also a global priority-based protocol. In this protocol, the time line is divided into frames as shown in Fig. 6.9. Every node maintains the current transmission window defined by priority values (*low*, *high*). A node that has a message in the window, i.e., a message whose priority is within the range defined by the *low* and *high* values can start to transmit. However, at the start of a frame it is possible that more than one node may find that they have a message that is eligible for transmission and start transmitting it. This



▲ **FIGURE 6.9**

Frames in the Window-Based Protocol

would result in a collision. On a collision, every node increments the value of low (i.e., makes $low++$) and on a free frame every node decrements the value of low (i.e., makes $low--$).

6.5.2 Calendar-Based Protocol

An example of calendar-based scheduling protocol is the dynamic reservation technique. In this protocol, each node maintains a calendar data structure where information about the access time reservations of the guaranteed messages of all nodes are maintained. When a message for which no reservation has yet been made arrives at a node, the node first determines a free slot by consulting its local calendar data structure. It then attempts to reserve a suitable free future time interval by broadcasting a *control message* to all nodes. Every node on receipt of a control message, updates its calendar accordingly. This protocol can efficiently handle deterministic periodic messages, and unlike the global priority-based protocols there is no overhead in priority arbitration. However, it becomes very difficult to handle aperiodic and sporadic messages in this protocol. Therefore, this protocol is used only in very simple networks such as CANs.

6.6 BOUNDED ACCESS PROTOCOLS FOR LANs

IEEE 802.4 and RETHER are two popular examples of bounded access protocols that can support real-time communication in a LAN.

6.6.1 IEEE 802.4

IEEE 802.4 protocol can be used in token ring and token bus networks. This protocol is often referred to as the *timed token protocol*. In this protocol, a node can transmit, when it holds the token. Real-time guarantees to messages are provided by bounding the amount of time each node holds the token. This protocol has been incorporated in Fibre Distributed Data Interface (FDDI).

In IEEE 802.4, Target Token Rotation Time (TTRT) is used as a design parameter. TTRT is defined as the expected time between two consecutive visits of the token to a node. At the network initialization time, TTRT is specified by the network administrator depending on the characteristics of the messages that would be transmitted over the network.

The real-time messages of a node are assumed to be periodic in nature and are termed synchronous messages, and the non-real-time messages are termed asynchronous messages. Individual nodes are allocated a portion of TTRT, known as the node's *synchronous bandwidth*. This allotment is made according to the timing characteristics of the synchronous messages originating at all the nodes taken together.

Let the synchronous bandwidth of a node N_i be H_i time units. We have already mentioned that TTRT is distributed among the nodes such that each time the token visits a node N_i, it can transmit its synchronous messages for at most H_i duration. Let SN be the set of all nodes in the network. Then, TTRT can be represented as: $TTRT = \theta + \Sigma_{Ni \epsilon SN} H_i$, where θ is the propagation time and H_i is the token holding time at node N_i. When a node receives the token, it first transmits its synchronous traffic for a time bounded by its synchronous bandwidth. After transmitting all its synchronous traffic, it may transmit its asynchronous traffic, only the token had arrived early at the node. That is, it transmits asynchronous traffic only if the time since the previous departure of the token from the same node is less than TTRT.

A node can transmit non-real-time messages only when the token arrives at the node earlier than expected. The time for which asynchronous frames are transmitted is called *asynchronous overrun*. Asynchronous overrun reduces the effective bandwidth available to transmit synchronous messages and delays the time between consecutive arrival of the token at a node. In the absence of any asynchronous overrun, the expected interval of successive visits of the token to a node is TTRT. Due to asynchronous overrun, the worst case time between two successive visits of the token to a node is 2 × TTRT. Suppose, no node has either any synchronous or asynchronous message to transmit. In this situation, assume that the token arrives at node N_{i+1} TTRT time units early. Now, suppose node N_i transmits asynchronous messages for TTRT time units and the node N_i and all subsequent nodes make full use of their assigned time slots for transmitting synchronous messages. Then, the token would arrive at node i after 2 * TTRT time units since its last visit. Therefore, a network that uses only synchronous mode, time between consecutive token arrivals would be limited by TTRT.

Suppose, among all messages that can originate at the different nodes, let the node N_i have the message having the smallest deadline Δ. That is, Δ is the shortest deadline among all messages in the network. Since, the worst case time between two successive visits of the token to a node is 2 × TTRT, unless TTRT is set to a value smaller than $\Delta/2$ during the network initialization time, the deadline for the shortest deadline message would not be met. On the other hand, a TTRT value much smaller than $\Delta/2$ would lead to increase in larger overhead and would result in low network capacity utilization If TTRT is set to be larger than $\Delta/2$, then real-time messages would miss their deadlines.

Token holding time of an individual node is the synchronous bandwidth alloted to the node. As soon as a node N_i receives the token, it starts a timer set to its synchronous bandwidth (H_i), and releases the token upon expiry of the timer. The synchronous bandwidth allocated to a node N_i is given by the following expression:

$$H_i = (\text{TTRT} - \theta) * \frac{C_i/T_i}{\Sigma C_i/T_i} \tag{6.1}$$

where C_i is the size of the message (in bits) that node N_i requires to transmit over T_i interval, and $\frac{C_i}{T_i}$ is the channel utilization due to the node N_i.

Example 6.1

Suppose a network designed using IEEE 802.4 protocol has three nodes. Node N_1 needs to transmit 1 mb of data every 300 mSec. Node N_2 needs to transmit 1.2 mb of data every 500 mSec. Node N_3 needs to transmit 1.2 mb of data every 500 mSec. Select a suitable TTRT for the network and compute the token holding time for each node. Ignore the propagation time.

Solution. From an examination of the messages, it can be seen that the shortest deadline among all messages is 200 mSec. Therefore, we can select $TTRT = \frac{200}{2} = 100$ mSec. The

channel utilization due to the different nodes would be:

$$\frac{C_1}{T_1} = \frac{1 \times 8}{300} \text{ mb/mSec}$$

$$\frac{C_2}{T_2} = \frac{1.2 \times 8}{500} \text{ mb/mSec}$$

$$\frac{C_3}{T_3} = \frac{2 \times 8}{200} \text{ mb/mSec}$$

$$\frac{C_1}{T_1} + \frac{C_2}{T_2} + \frac{C_3}{T_3} = \frac{1 \times 8}{300} + \frac{1.2 \times 8}{500} + \frac{2 \times 8}{200} = \frac{377.6}{3000} \text{ mb/mSec}$$

Using Expr. 6.1, the token holding times (H_i) of the different nodes can be determined as follows:

$$H_1 = 100 \times \frac{8}{300} \times \frac{3000}{377.6} = 21.18 \text{ mSec}$$

$$H_2 = 100 \times \frac{9.6}{500} \times \frac{3000}{377.6} = 15.25 \text{ mSec}$$

$$H_3 = 100 \times \frac{16}{200} \times \frac{3000}{377.6} = 63.56 \text{ mSec}$$

6.6.2 RETHER

RETHER stands for Real-time ETHERnet. RETHER enhances TCP/IP to provide real-time performance guarantees to real-time applications, without modifying the existing Ethernet hardware [28]. In RETHER, network transmissions can occur in two modes: CSMA/CD mode or RETHER mode. The network switches transparently to RETHER-mode when there are real-time sessions and transits back to CSMA/CD-mode when all real-time sessions terminate. Protocol switching is done to minimize the performance impacts on non-real-time traffic when there are no real-time sessions. In RETHER mode, a token passing scheme is used.

Protocol Description: In the absence of any real-time messages at the nodes, nodes compete for the channel using the usual CSMA/CD protocol. When a node receives a real-time request from a local application, it broadcasts a Switch-to-RETHER message, if the network is not already in RETHER mode. Every node that receives this message responds by setting the mode of its protocol to RETHER mode and acknowledges back to the initiator. The transmitting node waits for the ongoing packet transmission to complete. It then sends an acknowledgment back to the initiator, indicating its willingness to switch to RETHER mode and that there is no data left in the back-off phase of CSMA/CD protocol. After receiving all the acknowledgments, the initiating node creates a token and begins circulating it. This completes a successful switch to RETHER mode.

If more than one initiator tries to initiate RETHER-mode at the same time, each node acknowledges its switch message to the initiator with the smallest ID among the initiating

nodes. An initiator only sends an acknowledgment to another initiator, if its node ID is smaller than its own. In case of loss of the acknowledgment, or when some node does not receive the Switch-to-RETHER broadcast, or when some nodes are dead, then the initiator times out due to non-receipt of all the acknowledgments. After a fixed number of retries, it concludes that the nodes that did not acknowledge are dead, and conveys this to all other live nodes.

The RETHER mode uses a timed token scheme (see Section 6.6.1) to provide bandwidth guarantees. At any time, there can be only one real-time request per node and each real-time request specifies the required transmission bandwidth in terms of the amount of data it needs to send during a fixed interval of time called Targetted Token Rotation Time (TTRT). The maximum token holding time is calculated for each node based on this information (amount of data and TTRT). Based on this information, bandwidth is reserved for each session.

We now discuss the behaviour of the nodes on receipt of the token. The control token circulates among two sets of nodes, the Real-Time Set (RTS) and the Non-Real-Time Set (NRTS). Only nodes that have made a bandwidth reservation belong to RTS. All other nodes belong to NRTS. During each token rotation, the token visits all nodes in the real-time set in order. When a node in real-time set receives the token, it sends its real-time data and passes the token to its next neighbour in the real-time set. The last node in the RTS passes the token to the NRTS, if there are no reservations. Let TTRT be the mean token rotation time and let MTHT_i be the mean token holding time for node N_i. The token is then tagged with a TimeToDeadline field computed as:

$$\text{TimeToDeadline} = \text{TTRT} - \sum_{i \epsilon RTS} \text{MTHT}_i$$

A positive value of TimeToDeadline would indicate that some NRT messages can be transmitted even after transmitting all real-time messages. When an NRT node receives the token, it determines whether there is sufficient time to send a packet without negatively affecting the real-time messages at any of the nodes. If sufficient time exists, it sends a packet, and decrements TimeToDeadline accordingly. It then passes the token to the next node in NRTS. If there is no time to send a packet, it informs the last node in the real-time set that it should be the first node to get the token among NRTS nodes during the next round, and it passes the token to the first node in the RTS.

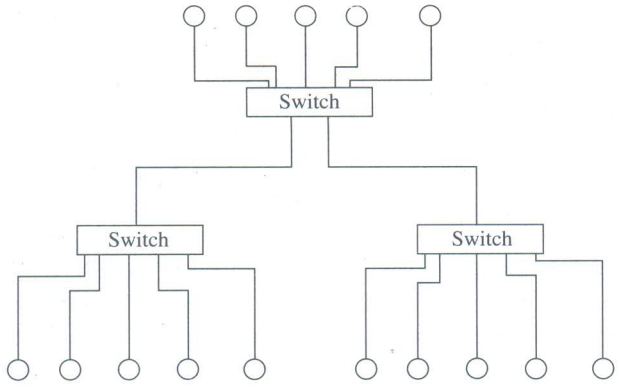
Every new real-time request goes through an admission control procedure that determines if it is possible to accept the request. Admission control is performed locally on each node: A real-time request is admitted, only if

$$\sum_{i \epsilon RTS} \text{MTHT}_i + \text{MTHT}_{\text{new}} + \text{TBNRT} \leq \text{TTRT}$$

where TBNRT is the bandwidth reserved for the NRT set. MTHT_{new} is the bandwidth required by the new node in the RT set. When a real-time node wants to terminate its real-time connection, it merely removes itself from the RTS information on the token.

6.6.3 Switched Real-Time Ethernet

Modern Ethernet networks are more and more being built using switches instead of hubs or coaxial cables. Use of hubs and switches creates a star-like network (see Fig. 6.10). The difference between switches and hubs is intelligence. Hubs simply pass on incoming traffic from any port to all other ports. On the other hand, switches selectively send the traffic through a specific port based on their knowledge of the network topology. In the star-like network topology that is



▲ **FIGURE 6.10**
Switched Real-Time Ethernet

realized, there is a dedicated cable in each direction. Therefore, collisions can not occur on any network cable. The switches do not provide any guarantees as to which one will be sent first. A switch deploying bandwidth reservation can solve this problem.

6.7 PERFORMANCE COMPARISON

In this section, we compare the effectiveness of the different protocols discussed for providing QoS guarantees to real-time applications. We focus our attention only on the bounded access protocol (IEEE 802.4) and a priority-based protocol (IEEE 802.5).

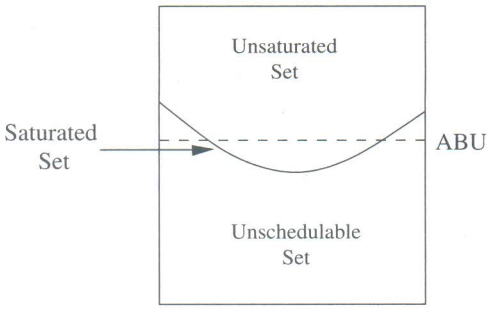
At the design time, a detailed knowledge of the message set that would be transmitted is often unavailable. However, the designers of a real-time network usually have an estimate of the traffic generated by the different real-time sources in terms of the respective channel utilizations. Therefore, a utilization-based metric is appropriate for being used during the network design stage.

We first discuss an intuitive classification of the various types of message sets that may be used in an application. We then discuss two metrics that are based on channel utilization on account of the different message sets. Each network needs to cater to a certain number of real-time messages, called a message set. Of course, there can be a large number of message sets that may arise in practice for different applications. The entire population of real-time message sets can be divided into three classes depending on the schedulability offered by a network as shown in Fig. 6.11. These three classes of messages sets are:

Unsaturated Schedulable: The message sets in this class are schedulable, and remain schedulable even when the size of any message in a message set is slightly increased. These message sets typically result in low channel utilization.

Saturated Schedulable: The message sets in this class are schedulable, but any increase in the size of a message, would make the corresponding message set unschedulable, and messages would miss their respective deadlines.

Unschedulable: The unschedulable set refers to those message sets for which deadlines of at least some messages would be missed.



▲ FIGURE 6.11
Schematic Representation of All Message Sets

We now discuss two utilization-centered metrics based on the above classification of real-time message sets.

- **Absolute Breakdown Utilization (ABU):** This metric indicates the expected value of utilization of a message set S, at which messages start to miss their respective deadlines. It is the average of the utilization of all messages in the saturated set. This metric can more formally be defined as follows. Let $U(S)$ be the utilization of the channel due to a certain message set S. $U(S)$ can be expressed as follows.

$$U(S) = \sum_{i \epsilon S} \frac{C_i}{T_i}$$

where C_i is the size of message $i \epsilon S$, and T_i is its period. ABU can now be defined as:

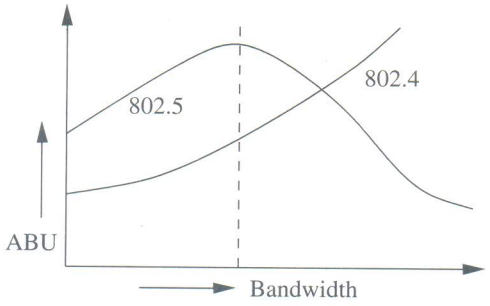
$$\text{ABU} = \frac{\Sigma_{S \epsilon \text{Sat}} U(S)}{|\text{Sat}|}$$

where Sat is the set of all saturated message sets. We can informally view this metric to indicate how much traffic a network can accommodate on the average case, without any message having to miss its deadline.

- **Guarantee Probability (GP(U)):** GP(U) is the guarantee probability at channel utilization U, (GP(U)) indicates the probability that all deadlines of a message set with utilization U would be met.

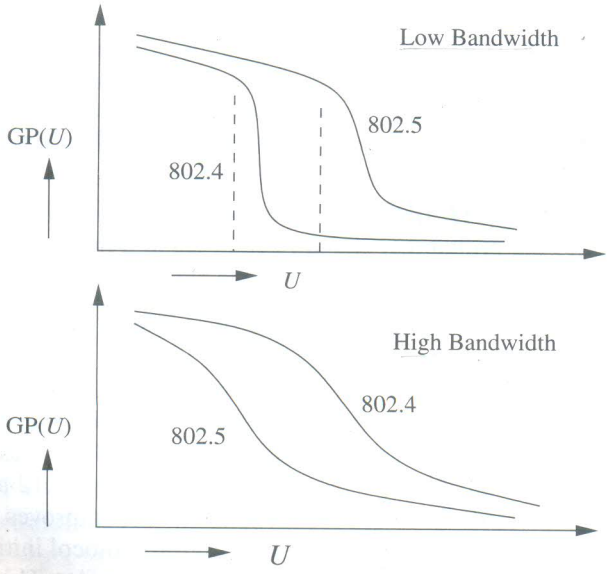
We can expect GP(U) to be close to 1 for utilization lower than ABU, and approach 0 as utilization increases beyond ABU.

In a performance study of the priority-based protocols and bounded access protocols against the above two metrics it was observed that in low bandwidth networks, priority-based protocols work better, whereas at higher bandwidths, the bounded access protocols work better [10]. Results of experimental studies have been shown in Figs. 6.12 and 6.13. The results indicate the the performance of the bounded access protocol improves monotonically with bandwidth. However, the performance of the priority-driven protocol initially improves as the bandwidth is increased, but starts to drop off beyond a certain point. This is against the intuition that the performance of a protocol should improve as the bandwidth is increased. Let us now analyze this anomalous behaviour.



▲ **FIGURE 6.12**
Absolute Breakdown Utilization versus Bandwidth

When the bandwidth is increased beyond a certain value, in case of the priority-based protocol (IEEE 802.5) the decrease in transmission time causes the frame transmission time F to be less than the token rotation time θ. In this case, before releasing a new token, the transmitting node has to wait for the token to return, even after it has completed the transmission of a frame. Thus, the effective frame transmission time in this case is θ and the fraction of the wasted bandwidth is $(\theta - F)/F$. The propagation time (θ) is independent of the bandwidth and hence can be considered as a constant. The token transmission time decreases with increasing bandwidth. Therefore, the percentage of wasted bandwidth increases with increase in bandwidth. This leads to deterioration of performance with increase in bandwidth after a certain bandwidth at which $\theta = F$. The timed token protocol does not exhibit this anomaly, because a node is permitted to transmit continuously during the time it holds the token.



▲ **FIGURE 6.13**
GP(U) versus Utilization at Low and High Bandwidths

From Fig. 6.12 it can be seen that in all cases, the guarantee probability remains close to 1 as long as the utilization is less than ABU. There is a sharp drop in the guarantee probability at utilization values close to ABU. This demonstrates that ABU is a robust measure of average performance of real-time networks. Now, we can summarize our observations as follows. At low network bandwidth, the priority inversions caused by the round-robin scheduling approach of the bounded access protocol tends to adversely impact messages with short deadlines. Thus, at low bandwidths, priority-driven protocol is better suited than the timed token protocol for real-time applications. However, at high bandwidth, priority-driven protocol leads to low channel utilization due to priority arbitration overheads.

6.8 REAL-TIME COMMUNICATION OVER PACKET SWITCHED NETWORKS

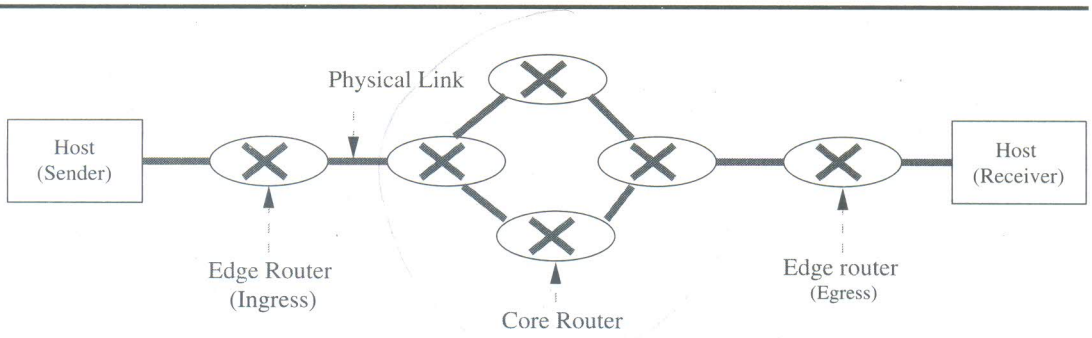
Packet switching is popularly deployed in wide area networks. A prominent example of a packet switched network is the Internet. From a modest beginning made a little over a decade ago, Internet has become a vast repository of information, and enabler of several new applications such as e-commerce. Internet provides *best-effort* service to applications, that is, traffic is processed as quickly as possible. However, there is no guarantee of timeliness or actual delivery. With the increasing commercial usage of Internet, there is now a need to support soft and firm real-time applications. In this section, we first provide some basic concepts about Internet. Subsequently, we discuss how QoS guarantees can be provided on the Internet. In particular, we discuss QoS routing, resource reservation, traffic shaping and policing—some of the key techniques that are being used to provide QoS guarantees to applications.

6.8.1 A Basic Model of Internet

The basic Internet service model has been schematically shown in Fig. 6.14. As shown in the figure, the receiver and the sender are connected to each other through several (possibly thousands of) routers and links. The routers are of two basic types: edge routers and core routers. The core routers are typically located at the ISP[1] and the edge routers are connected to the nodes at the periphery of the network. The edge routers can either be ingress or egress type. The egress routers are attached at the receiver-end, whereas an ingress router is attached at the sender's end. An ingress router is the first interface to the data sent by a sender host. As shown in Fig. 6.14, the different hosts and routers are interconnected using links. The type of physical media used at a link can be as varying as a twisted pair cable, a fiber optic cable, or even a wireless medium. The data generated by various users are routed by the ingress routers to the routers at the ISP. The routers at the ISP are termed core routers. The density of these routers is very high at the ISP end, since huge amount of data and links typically converge at the ISP. From the core routers, data is forwarded through the egress routers (at the receiver end) to the corresponding receivers.

Real-time communication can be supported on the Internet by providing preferential treatment to data of certain senders that have entered into a contract with the ISP. But for that to happen, data packets must have been classified before transmission. At the connection set-up time, the sender provides the ISP with two parameters: the traffic specification and the QoS specification.

[1]ISP stands for Internet Service Provider. It is the institution that provides access to the users to the Internet.



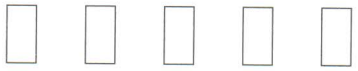
▲ **FIGURE 6.14**
A Simple Model of the Internet

The ISP uses this to determine whether it can accept or has to refuse the client request. It checks for the availability of resources on the possible routes on which it can forward the data packets. Alternatively, we can say that it tries to find a path (unicast or multicast routing) having the necessary resources to satisfy the required QoS for the specified traffic pattern to be generated by the source.

When the network accepts a connection request, it guarantees that the required QoS will be met. However, the given guarantee remains valid only if the client-generated traffic remains within the specified traffic bounds. If the client traffic violates the prespecified traffic bounds, then the network may drop packets and/or reshape the traffic. Policing is the term used to indicate dropping of the packets of misbehaving traffic sources. Note that the network needs to meet the given performance bounds as long as the client traffic remains within the traffic specification, irrespective of the behaviour of other clients. However, this performance can not be guaranteed, if the connections are not protected from the misbehaving (or malicious) real-time and non real-time sources. To avoid violating the guarantees made to real-time connections, the network must either explicitly control the input rates on a per-connection basis (shaping and policing) or adopt scheduling algorithms at the switches. Several popular scheduling algorithms such as Fair Queuing, Weighted Fair Queuing, etc. exist. These techniques are a part of the modern QoS architectures for the Internet such as IntServ and DiffServ. We elaborate these topics, as well as how QoS guarantees can be provided to the real-time traffic in the following sections.

6.8.2 Traffic Characterization

As already mentioned, while requesting for a real-time connection, the source has to specify the traffic characteristics and the required QoS. The network, in turn, uses the two to check whether it has adequate resources to meet the required QoS. This is part of an admission control procedure. Traffic characterization is essentially a model of the characteristics of the data generated by a source to be transmitted over a network. Normally, traffic is characterized by bounding the volume of data it can generate per unit time. An accurate bound on the volume of the traffic can help to bound the amount of network resources that may have to be reserved to provide the required quality of services. During connection establishment, traffic sources are required to specify their traffic characteristics. Network resources are reserved based on the traffic characterization and the service requirements. Many models have been proposed for traffic characterization. In the following, we discuss a few important ones.



▲ **FIGURE 6.15**
Constant Bit-Rate Traffic

(X_{min}, S_{max}) model: The (X_{min}, S_{max}) model bounds a traffic source with a peak rate. A connection satisfies (X_{min}, S_{max}) model, if the inter-arrival times between two packets is always less than X_{min}, and size of the largest packet does not exceed S_{max}. Both the peak and average rates of traffic in this model are identical, and is given by $\frac{S_{max}}{X_{min}}$.

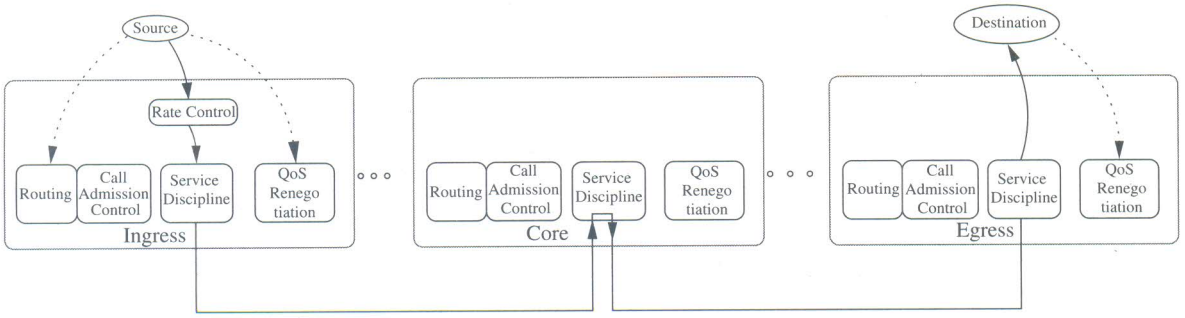
A CBR traffic has been shown in Fig. 6.15. It can be observed from Fig. 6.15 that for CBR traffic both the worst case and the average packet arrival times are indeed the same. Therefore, this model provides a tight bound for CBR traffic. However, this model can not accurately specify bursty traffic and models the worst-case traffic in this case. If bursty traffic is specified using this model, then it leads to very conservative resource reservation, resulting in low utilization of the reserved resources and inefficient functioning of the network.

(r, T) model: In this model, the time axis is divided into intervals of length T each. Each such interval is called a frame. A source satisfies (r, T) model, if it generates no more than $r.T$ bits of traffic in any interval T. In this model, r is the upper bound on the average rate at which traffic is generated over the averaging interval T. This model is in many ways similar to the (X_{min}, S_{max}) model. It also provides a tight bound for CBR traffic. However, for bursty traffic it leads to very conservative resource reservation, leading to low utilization of the reserved resources.

$(X_{min}, X_{avg}, S_{max}, I)$ model: In this model, X_{min} specifies the minimum inter-arrival time between two packets. S_{max} specifies the maximum packet size and I is an interval over which the observations are valid. X_{ave} is the average inter-arrival time of packets over an interval of size I. A connection would satisfy $(X_{min}, X_{avg}, S_{max}, I)$ model, if it satisfies (X_{min}, S_{max}) model, and additionally, during any interval of length I the average inter-arrival time of packets is X_{ave}. In this model, the peak and the average rates of the traffic are given by S_{max}/X_{min} and S_{max}/X_{ave}, respectively. Note that this model bounds both peak and average rates of the traffic. Therefore, this model provides a more accurate characterization of VBR traffic compared to either (X_{min}, S_{max}) model or (r, T) model.

(σ, ρ) model: In this model, σ is the maximum burst size and ρ is the long term average rate of the traffic source. Average traffic ρ is calculated by observing the number of packets generated over a sufficiently large duration and dividing this by the size of the duration. A connection would satisfy (σ, ρ) model, if during any interval of length t, the number of bits generated by the connection in that interval is less than $\sigma + \rho \times t$. This model can satisfactorily be used to model bursty traffic sources.

Multiple rate bounding: Bursty traffic sources can be characterized sufficiently accurately by bounding the traffic over multiple averaging intervals. A traffic would satisfy $\{(r_1, T_1) (r_2, T_2) \ldots\}$, if $T_1 < T_2 < T_3 \ldots$, and over any interval I the number of bits generated is bounded by $r_i \times T_i$, if $T_{i-1} < \mathrm{I} < T_i$. As the averaging T_i interval gets longer, a source is bounded by a rate lower than its peak rate, and closer to its long term average rate.



▲ **FIGURE 6.16**
QoS Framework

6.9 QoS FRAMEWORK

There are several components that work to provide QoS support to a connection request. These components can be viewed as the elements of a QoS framework to be set up by a network. This QoS framework has been shown in Fig. 6.16. The different elements of this framework are briefly discussed in the following. In the subsequent sections we provide a more elaborate treatment of these elements.

Call Admission Control: Call admission control tests whether the required resources are available. Then the required resources are reserved during call establishment, so that traffic can flow according to QoS specification. When the life time of a call is over, the call is torn down; this should involve releasing the resources reserved for the call.

Rate Control: Rate control ensures that the source does not violate the traffic (rate) contract, to which it has agreed. A source might violate its prespecified traffic (rate) due to malicious users or inexact traffic characterization.

Service Discipline: Service discipline allocates resources to connections during data transfer, adhering to the reservations made during channel establishment. Three types of resources allocated by a service discipline, namely, bandwidth, processing time (for deciding the time of service), and buffer space.

QoS Renegotiation: A renegotiation request can come either from the user who wants to change his own QoS requirement, or from the network due to overload and congestion.

6.10 ROUTING

As described in the previous section, packet routing is an important element that can be effectively leveraged to provide QoS guarantees to applications. Once a traffic source has specified its traffic characteristics and its QoS requirements to the network, it waits for the network to accept the request before it can start its transmissions. The network, on its part, checks whether adequate resources are available along any path from source to destination to meet the request.

If no route with required resources are available, then the request is rejected. Route selection for a session takes place during the connection establishment phase in both unicast and multicast connections.

Traditional Internet protocols use routing algorithms such as the shortest path routing in which the routing can be optimized for some metric without taking into account whether the required resource is actually available. Consequently, the flows might finally be routed over paths that are unable to support the necessary resource requirements while other paths might exist which could have easily satisfied the specified requirements. Such a situation might lead to a breach of committed QoS guarantees to connections. Researchers have now proposed *QoS routing* or *constraint-based routing* which can overcome these problems. The primary goals of QoS routing are:

- To select routes that can meet the QoS requirements of a connection.
- To increase the utilization of the network.

While determining a route, QoS routing schemes not only consider the topology of a network, but also the requirements of the flow, the resource availability at the links, etc. Therefore, QoS routing would be able to find a longer and lightly-loaded path rather than the shortest path that may be heavily loaded. QoS routing schemes are, therefore, expected to be more successful than the traditional routing schemes in meeting QoS guarantees requested by the connection.

6.10.1 Routing Algorithms

The complexity of a QoS routing algorithm depends on the chosen QoS constraints such as hop count, bandwidth, delay, delay-jitter etc. Let us now analyze the QoS constraints, as they have a major impact on the complexity of a QoS routing algorithm.

The QoS constraints can be divided into three broad classes [30]. Let $d(i, j)$ denote a chosen constraint for the link (i, j). For any path $P = (i, j, k, \ldots l, m)$, a constraint d would be termed *additive*, if $d(P) = d(i, j) + d(j, k) + \cdots + d(l, m)$. That is, the constraint for a path is the sum of the constraints of the individual links making up the path. For example, end-to-end delay is an additive constraint and is equal to the sum of the delay of the individual links in the path. Jitter, cost, and hop count are a few more important additive constraints.

A constraint is termed *multiplicative*, if $d(P) = d(i, j) \times d(j, k) \times \cdots \times d(l, m)$. For example, reliability constraint is multiplicative. The reliability of a path is given by the product of the reliability of the individual links in the path.

A constraint is termed *concave*, if $d(P) = \min\{d(i, j), d(j, k), \ldots, d(l, m)\}$. That is, for a concave constraint, the constraint of a path is the minimum of all the constraints of the individual links making up the path. An example of a concave constraint is bandwidth. Bandwidth on a path is equal to the minimum bandwidth of a link on that path. Wang and Crowcroft proved that the problem of finding a path subject to two or more additive and/or multiplicative constraints in any possible combination is NP-complete [31]. However, the proof of NP-completeness is based on the assumptions that: 1) all the considered constraints are independent; and 2) the delay and jitter of every link are known a priori. Although the first assumption is normally true for circuit-switched networks; bandwidth, delay, and jitter are not independent parameters in case of packet-switched networks. In spite of this, it is true that determining a route subject to the path constraints is a computationally expensive problem. As a result, polynomial algorithms like Bellman-Ford

and the extended Dijkstra's algorithm [27] exist for computing routes with hop count, delay, and jitter constraints. Interestingly, bandwidth and hop count constraints are more important than the delay or jitter constraints, since these two are to a large extent determined by the bandwidth and hop count constraints. Most real-time services require bandwidth as an essential constraint, while the hop count constraint determines effective resource utilization.

In the last two decades, several algorithms have been proposed to address the various problems associated with unicast and multicast routing. Comprehensive reviews of unicast and multicast algorithms for QoS routing are available in [3, 21]. Multimedia applications, which require multicast QoS routing, are becoming very important and are being widely used. In the following, we discuss some issues relevant to multicast QoS routing.

6.10.2 Multicast Routing

Internet is increasingly being used for many firm real-time applications. Video-on-demand and VoIP have already become common place. Besides, many distributed applications require multicast communication. Giving each source an independent point-to-point connection to each destination is an unacceptable approach because of the potential wastage of resources. An approach that is commonly being used is to construct a multicast tree connection which connects each source to all the receivers. Each vertex on a multicast tree would represent either a router or a host.

When there are many sources transmitting to several receivers—during multicast routing, considerable savings of resources can be achieved by having the sources share resources whenever possible. This is because all the sources are not active all the time. For example, in an audio conference, only one (or at best a few) people speak at any time. Therefore, several audio conference connections can share resources to optimally use the resources. In this case, the total resources for all the connections taken together can be significantly smaller than the sum total of the resources required for individual point-to-point connections, and yet meeting the necessary QoS requirements. In such a scenario, we can define a *multicast group* as one in which traffic from a set of sources traverse some common routers and transmission links.

The QoS routing function in a multicast application involves finding a tree rooted at a source and covering all the receivers, with every path from the sender to the receivers satisfying the QoS requirement. There are essentially two basic types of tree construction algorithms: source-based and core-based. In a source-based algorithm, the construction of the multicast tree is achieved by initiating the tree construction from each of the sources, whereas in a core-based algorithm a core is formed first, each receiver then joins the core.

6.11 RESOURCE RESERVATION

A network can provide QoS guarantees to connections only if it can successfully reserve appropriate resources along the identified routes. Merely finding a route that has sufficient resources may not automatically solve the problem of providing QoS guarantees. If no reservation scheme is undertaken, then the traffic being dynamic, can increase and congest the system that, in turn, will hurt the QoS guarantees badly. So, definite allocation of resources helps in safeguarding the QoS guarantees to the individual connections. In this section we discuss a popular and widely used protocol named Resource reSerVation Protocol (RSVP) [32, 35]. RSVP attempts to reduce the bandwidth and buffer space reserved for a multicast group at the routers

and links traversed by more than one multicast tree by considering the aggregate requirement of the multicast group as a whole. Further, the reservation is dynamic in nature, that is, a receiver needs to periodically send messages to the routers to maintain its resources. This results in efficient resource usage in the Internet environment, since in the Internet environment receivers may not formally tear down connections when they do not require a connection any more.

6.11.1 Resource reSerVation Protocol (RSVP)

RSVP was designed in 1993 at Xerox Palo Alto Research Center. The idea was to design a resource reservation protocol that can scale to multicast communication and can incorporate heterogeneous receivers. It is important to remember that RSVP is solely a resource reservation protocol, it does not construct any routes. It assumes that the multicast tree has been set up by the routers of the network. Admission of a new multicast connection is handled by the admission control module in a lower layer.

RSVP establishes and maintains a *sink tree* for each destination and uses it to send control messages from each destination. A sink tree is a spanning tree that is rooted at the destination and connects all the sources in a multicast group. It is obtained by tracing in the reverse direction from the destination to every source along the paths in a multicast tree.

Each router used by a multicast group maintains two types of information: *path state* and *reservation state*. The path state of each router consists of the path used by the multicast group. RSVP uses this information to maintain the sink trees of the destination. The reservation state of each router is concerned with resources set aside for different destinations to support the required QoS. The name of the destination (reserver) and the amount reserved are among the information provided by the reservation table.

Protocol Operation: RSVP is a receiver-oriented protocol. That is, the receiver of a data flow initiates and maintains the resource reservation used for that flow. Each router may receive reservation messages from each of the downstream links in the multicast tree, but it sends only one reservation message to its upstream link.

A path message is a control message sent by a source and forwarded by the routers in a multicast tree. Each source sends its first path message before it commences transmission and sends subsequent path messages periodically. Among the information carried in each path message of a source is the flow specification of its message stream. A router updates the *path state* upon receiving the path message from a source.

Each destination sends reservation request messages (resv messages, for short) which are forwarded along the sink tree of the destination. Among the information contained in each reservation message is the flow specification giving the QoS requirement of the receiver.

It is clear that the amount of bandwidth that a router reserves should not exceed the link's capacity. Towards this, whenever the router receives a new reservation message, it determines if its downstream links on the multicast tree can provide the required reservation. If the admission test fails, the router rejects the reservation and returns an error message to the appropriate receiver(s). RSVP does not define any specific admission test, but it assumes that the router performs such a test and that the RSVP can request for such a test to be performed.

The destination sends its first reservation message when it receives a path message from a source whose message stream it wishes to receive. Afterwards it sends reservation messages periodically to maintain reservation. The maintenance of reservation through periodic messages is what sets RSVP apart from most of the other resource reservation protocols. This is called

soft state and is the preferred design choice for applications that are dynamic in nature. However, the need for each host to periodically transmit messages for the purpose of refreshing the states maintained by routers leads to a higher protocol overhead. This overhead is somewhat lessened by the fact that control messages require lower overhead than data messages. Merging of control messages from all hosts and forwarding whenever there is no new state information, the protocol overhead can further be reduced.

Another concept associated with RSVP is that of a filter. A filter is a list of the names of sources whose message streams can use the resources reserved for the destination. A destination that wants a filter, includes the filter in its reservation message. When no filter is applied, all sources in the multicast group can use the resources reserved for it.

6.12 RATE CONTROL

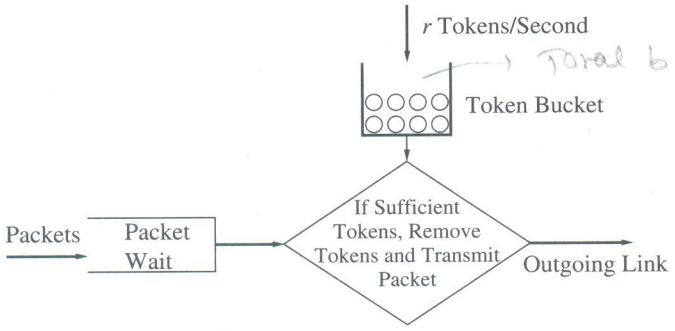
In a resource sharing packet-switched network, the admission control and service disciplines by themselves are not sufficient to provide performance guarantees. This is so because users may, inadvertently or otherwise, attempt to exceed the rates specified at the time of channel establishment. Rate control is the key mechanism used to prevent the greedy or malicious real-time and non real-time sources from negatively affecting the QoS guarantees given to other connections. Rate control can be achieved using either traffic shaping or policing. Table 6.1 summarizes the main differences between shaping and policing. The choice between which one to use depends on the type of application and its specific QoS requirements.

Several traffic shaping and policing techniques are available. In the following, we discuss an important traffic shaping and policing technique called *token bucket with leaky rate control.*

Token Bucket with Leaky Rate Control: A popular technique used in shaping and policing is token bucket with leaky rate control [27]. The leaky bucket algorithm is a linear bounded arrival process. This technique has schematically been shown in Fig. 6.17. As shown in Fig. 6.17, a bucket can hold up to certain prespecified number of tokens. Assume that a certain token bucket can hold up to b tokens. Tokens are added to this bucket as follows: new tokens that might be added to the bucket are generated at a rate of r tokens per second. If the bucket contains less then b tokens when a token is generated, the newly generated token is added to the bucket, otherwise the newly generated token is ignored, and the token bucket remains full with b tokens.

TABLE. 6.1 Traffic Shaping versus Policing

	Shaping	Policing
Objective	Buffers the packets that are above the committed rates.	Drops the excess packets over the committed rates. Does not buffer.
Handling Bursts	Uses a leaky bucket to delay traffic, achieving a smoothing effect.	Propagates bursts. Does no smoothing.
Advantage	Avoids retransmission due to dropped packets.	Avoids delay due to queuing.
Disadvantages	Can introduce delays due to queuing.	Can reduce throughput of affected streams.



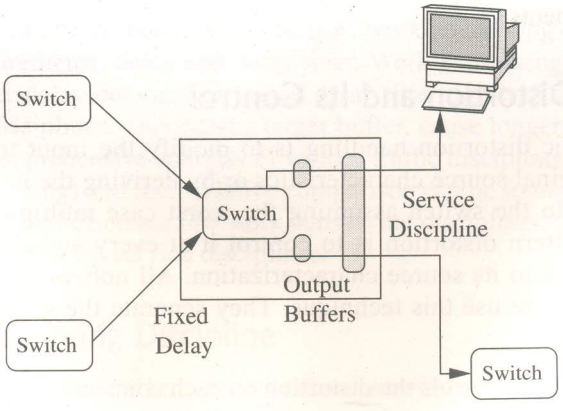
▲ **FIGURE 6.17**

Token Bucket with Leaky Rate Control

In leaky rate control, a waiting packet is taken for transmission only if there is at least one token in the bucket. If enough tokens are not available in the bucket, then the packet waits until the bucket has enough tokens in case of a shaper. However, in case of a policer, the packet is discarded. If transmission of the packet takes place, then the token is removed. Because there can be at most b tokens in the bucket, the maximum burst size of a leaky-bucket-policed flow is b packets. Furthermore, because the token generation rate is r, the maximum number of packets that can enter the network of any interval of time of length t is $rt + b$. Thus, the token generation rate, r, serves to limit the long-term average rate at which the packet can enter the network. This technique can also be used to police the peak rate along with long-term average rate.

6.12.1 Service Discipline

A simple model of an Internet is a collection of sending and receiving nodes and network switches. At every switch, there may be many incoming and many outgoing communication links as shown in Fig. 6.18. Depending upon the destination of a packet, it is queued in the buffer



▲ **FIGURE 6.18**

A Model of a Packet Switched Network

of the particular outgoing link. A scheduler selects packets to be transmitted from the buffer at the output link based on some scheduling policy. A service discipline is the terminology used to denote the mechanism used to schedule incoming packets for transmission. For real-time connections, rate-based service disciplines are used to protect the guarantees given to connections from network load fluctuations. This guarantees a minimum service rate to individual connections regardless of the traffic generated by other connections.

A service discipline helps isolate well-behaved guaranteed traffic sources from ill-behaved sources, network load fluctuation and best-effort traffic. An ill-behaving user or a malfunctioning equipment may send packets to the switch at a higher rate than declared. Even when the source is well-behaved, network load fluctuations may cause higher instantaneous arrival rate even from the source, though the traffic satisfied the specified rate at the entrance of the network. That is, a traffic may lose its original rate characteristics as it progresses through the network.

Unless special care is taken, traffic gets burstier and burstier after it passes through each switch, and it may no longer satisfy its source characterization. This is because the queuing at each switch has a bunching effect on the packets of a session.

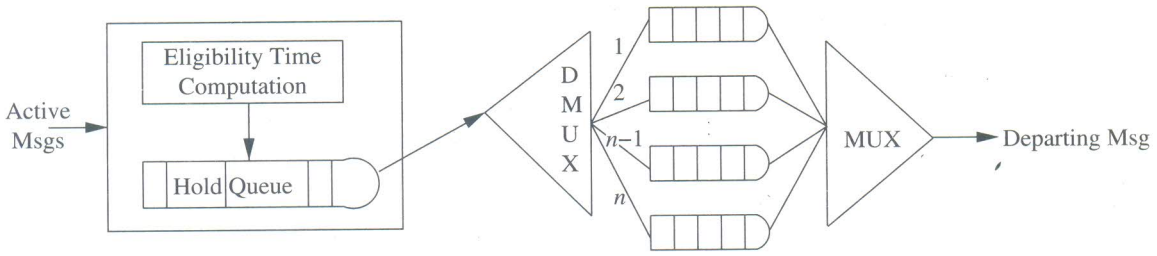
For example, if a connection satisfies (σ, ρ) model at the source, then its worst case traffic characterization just before the ith switch becomes $(\sigma + \Delta\sigma, \rho)$, where $\Delta\sigma = \sum_{j=1}^{i-1} \rho \cdot d_j^{\max}$ and $d_j^{\max}$ is the local delay bound for the connection at the jth switch.

In the case of best-effort traffic, since the traffic is not constrained they can very well cause congestion and disrupt the service to the well-behaved guaranteed traffic sources. The service discipline at any time decides output buffers from which packets would be taken up for transmission next. The switch/router does the switching of the data packets from its input port to the desired output port, so that it reaches the destination address. Every switch/router can be modelled as a queue, where packets from different flows compete for the switching processing time and output link. There may be many incoming links to a switch, and a number of virtual connections for each link, and there may be a number of packets in each connection to be served. These packets may compete for the same output link. The manner in which the queued packets are selected for transmission on the link is known as the scheduling discipline. The scheduling discipline at each switch selects which packet to transmit next by discriminating packets based on their QoS requirements.

6.12.2 Traffic Distortion and Its Control

The role of the traffic distortion handling is to modify the input traffic either by reconstructing it to its original source characteristics or by deriving the new traffic characterization at the entrance to the switch assuming the worst case multiplexing. Another way to handle the traffic pattern distortion is to control it at every switch by reconstructing the traffic of a connection to its source characterization. All non-work-conserving disciplines proposed in the literature use this technique. They separate the service discipline into two components:

1. **A rate controller:** It controls the distortion on each connection using regulators and allocates bandwidth to them.
2. **A scheduler:** It orders transmission of packets from different connections, and provides per-connection delay bounds.



▲ **FIGURE 6.19**
Processing in a Non-Work Conserving Service Discipline

6.12.3 Types of Service Disciplines

There are essentially two types of service disciplines:

- **Work Conserving Discipline:** In work conserving scheduling schemes, the server is never idle if there is a packet to be sent. Thus, a work conserving service discipline forwards an incoming packet on a required outgoing link, as long as the outgoing link is idle. It queues the packet, if the required outgoing link is busy. Thus, work conserving service discipline does not help in achieving traffic shaping. A few of the well-known work-conserving service disciplines are: WFQ, multi-level FCFS queue, and Delay-EDD.
- **Non-Work Conserving Discipline:** In a non-work conserving discipline, each packet is assigned, either explicitly, or implicitly, an eligibility time, a time after which a packet becomes eligible to be serviced (see Fig. 6.19). An outgoing link idles when no packets are eligible for transmission. For supporting best-effort service, conventional service disciplines such as FCFS or round-robin services are sufficient. However, per connection performance guarantees need more elaborate service disciplines. A server is allowed to send only the packets which are eligible and it is never idle as long as there are eligible packets to send. A few of the well-known non-work-conserving disciplines are Jitter-EDD, Stop-and-Go, HRR, and RCSP.

The type of service discipline being deployed (i.e., work conserving or non-work conserving) affects buffer requirements, delay and delay jitter. Work conserving disciplines need less buffer and cause shorter delay but can not bound delay jitter tightly. The opposite is true for non-work conserving disciplines: they need a larger buffer, cause longer delay, but can bound delay jitter tightly. A comprehensive analysis of various traffic disciplines (both work conserving and non-work conserving) and their relative comparison can be found in [33].

In the following, we first discuss a few work conserving disciplines, and subsequently, discuss a few non-work-conserving service disciplines.

6.12.4 Work Conserving Discipline

In this subsection, we discuss a few important work-conserving service disciplines.

Multi-level FCFS Queue: It is used to implement static priority schedulers. Each level of a FCFS queue corresponds to a different priority level. Each connection is assigned a priority and

all packets from that connection are inserted into the FCFS queue of that priority level. Multiple connections can be assigned to the same priority level. Packets are scheduled in FCFS order from the highest-priority non-empty queue for transmission. The only restriction in using multi-level FCFS queue for static priority schedulers is that the number of different delay bound values that can be provided to the connections are bounded by the number of priority levels of the system.

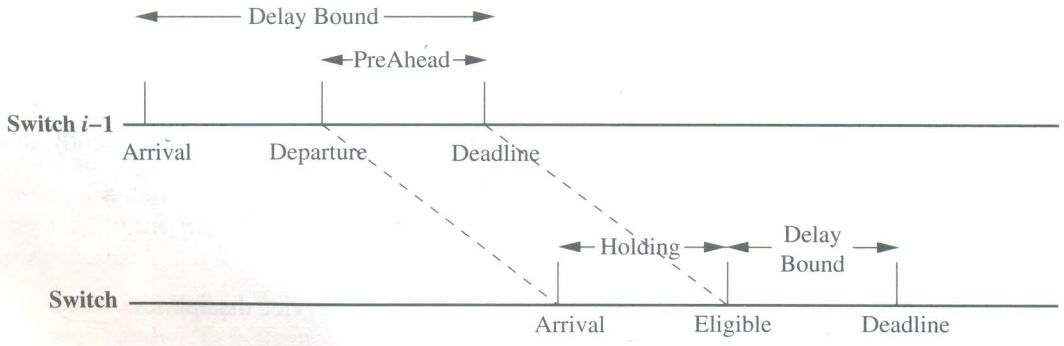
Weighted Fair Queuing (WFQ): WFQ [6] has evolved from Fluid Fair Queuing (FFQ), also known as Packet Generalized Processor Sharing (PGPS). In FFQ there is a separate FIFO queue for each connection sharing the same link. During any time interval when there are exactly N non-empty queues, the server serves the N packets at the head of the queues simultaneously, each at the rate of $1/N$th of the link speed. FFQ allows different connections to have different service shares. An FFQ is characterized by N positive real numbers $\phi_1, \phi_2 \cdots \phi_N$, each corresponding to one queue. At any time, the service rate for a non-empty queue i is exactly $(\phi_i / \Sigma_{j \epsilon B(\tau)} \phi_j) * C$ where $B(\tau)$ is the set of non-empty queues and C is the link speed. Therefore, FFQ serves packets in non-empty queues according to their service shares. FFQ is impractical as it assumes that the server can serve all connections with non-empty queues simultaneously and that the traffic is infinitely divisible. In a more realistic packet system, only one connection can receive service at a time and an entire packet must be served before another packet can be served.

6.12.5 Non-Work Conserving Disciplines

In the following, we discuss a few important non-work-conserving service disciplines.

Jitter-Earliest Due Date: Jitter-Earliest Due Date (Jitter-EDD) [29] provides delay-jitter bounds. After a packet has been served at a server, a field in its header is stamped with the difference between its deadline and the actual finishing time. A regulator at the entrance of the next server holds the packet for this period before it is made eligible to be scheduled.

Jitter-EDD is illustrated in Fig. 6.20, which shows the progress of a packet through two adjacent switches i and $i + 1$. As shown in Fig. 6.20, in the first server at switch $i - 1$, the packet got served PreAhead seconds before its deadline. So, in the next server (switch i), it is made eligible to be sent only after PreAhead seconds. Since there is a constant delay between the eligibility times of the packet at two adjacent servers, the packet stream can be provided a delay



▲ **FIGURE 6.20**

Packet Service in Jitter-EDD

jitter bound. Assuming that there is no regulator at the destination host, the end-to-end delay jitter bound is the same as the local delay bound at the last server.

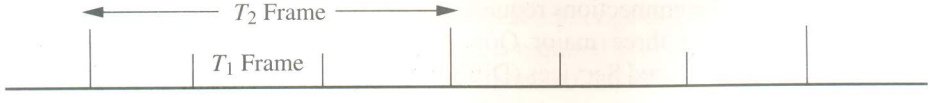
Stop-and-Go: Stop-and-Go discipline [7] is based on multi-level framing strategy. First, we describe this discipline using one-level framing, and then we extend it to multi-level framing. Let us consider one-level framing first. In one-level framing, at each node, the local time axis is divided into intervals of fixed size T, called a frame. Although convenient, frames of different nodes need not be synchronized. Bandwidth is allocated to each connection as a certain fraction of the frame time, which bounds the average rate of the traffic of a connection, where the averaging interval is T. Stop-and-Go defines departing and arriving frames for each link. At each switch, the arriving frame of each incoming link is mapped to the departing frame of the output link by introducing a constant delay θ, where $0 \leq \theta < T$. According to the Stop-and-Go discipline, the transmission of a packet that has arrived on any link l during a frame f should always be postponed until the beginning of the next frame. Thus, the server remains idle until then, even if there are packets queued for transmission.

The framing strategy introduces the problem of coupling between delay bound and bandwidth allocation granularity. The delay of any packet at a single switch is bounded by two frame times. To reduce the delay, a smaller T is desired. However, since T is also used to specify traffic it is tied to bandwidth allocation granularity. Assuming a fixed packet size P, the minimum granularity of bandwidth allocation is P/T. To have more flexibility in allocating bandwidth, or smaller bandwidth allocation granularity, a larger T is preferred. It is clear that low delay bound and fine granularity of bandwidth allocation can not be achieved simultaneously in a framing strategy like Stop-and-Go.

Multi-level framing is used to get around this coupling problem. In this, the time axis is divided into a hierarchical framing structure shown in Fig. 6.21. For a n level framing with frame sizes $T_1, \ldots, T_n$, and $T_m + 1 = K_m.T_m$ for $m = 1, \ldots, n - 1$, packets on a level p connection need to observe the Stop-and-Go rule with frame size T_p. That is, level p packets which arrived at an output link during a T_p frame will not become eligible for transmission until the start of next T_p frame.

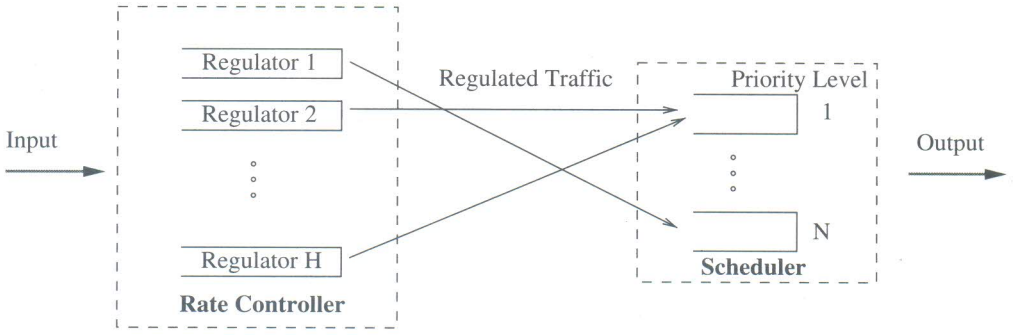
Rate-Controlled Static-Priority: The previously discussed service-disciplines were either based on sorted priority queue mechanism (e.g., WFQ) or framing strategy (e.g., Stop-and-Go). Sorted priority service-disciplines are complex and difficult to implement and framing strategy suffer from the dependencies between queuing delay and granularity of bandwidth. To get around these problems, Zhang and Ferrari [34], proposed Rate-Controlled Static-Priority wherein the function of rate control has been decoupled from delay control.

As shown in Fig. 6.22, an RCSP server has two components: a rate-controller and a static priority scheduler. Conceptually, a rate controller consists of a set of regulators corresponding to each connection; each regulator is responsible for shaping the input traffic of the corresponding connection into the desired traffic pattern. Upon arrival of each packet, its eligibility time is calculated by the regulator and this time is assigned to the packet. The packet is held at the



▲ **FIGURE 6.21**

Two-Level Framing with $T_2 = 3 \times T_1$



▲ **FIGURE 6.22**
Rate Controlled Static Priority

regulator till its assigned eligibility time expires, after which it is handed to the scheduler for scheduling and transmission. The scheduler is implemented by a multi-level FCFS queue, which serves the packets in order of their priorities, i.e., the first packet is chosen from the highest priority non-empty queue. There are two types of regulators:

- Rate-jitter (RJ) regulator, which controls rate-jitter by partially reconstructing the traffic.
- Delay-jitter (DJ) regulator, which controls delay-jitter by fully reconstructing the traffic. We assume that the RCSP satisfy the (X_{min}, X_{ave}, I) characterization. The (X_{min}, X_{ave}, I) RJ regulator ensures that the output of the regulator satisfy the (X_{min}, X_{ave}, I) traffic model, while the DJ regulator ensures that the output traffic of the regulator is exactly the same as the output traffic of the regulator at the previous server. Thus, if the traffic satisfies the (X_{min}, X_{ave}, I) characterization at network entrance, both types of regulators will ensure that the output of the regulator, which is the input to the scheduler, will satisfy the same traffic characterization.

In an RCSP server the scheduler uses a non-preemptive static priority policy; it always selects the packet at the head of highest priority queue that is not empty. The SP scheduler has a number of priority levels with each priority level corresponding to a delay bound. Each connection is assigned to a priority level during connection establishment time. Multiple connections can be assigned to the same priority level, and all packets on the connections associated with a priority level are appended to the end of the queue for that priority level.

6.13 QoS MODELS

Internet Engineering Task Force (IETF)[2] proposed several models for providing QoS assurances to Internet-based applications. These models are called *QoS models* or *QoS architectures*. The models require that the connections requesting QoS must follow certain procedures to avail QoS guarantees. There are three major QoS models that have been proposed: Integrated Services (IntServ), Differentiated Services (DiffServ) and lastly Multi-Protocol Label Switching

[2]The Internet Engineering Task Force (IETF) is a large open international community of network designers, operators, vendors, and researchers concerned with the evolution of the Internet architecture and the smooth operation of the Internet.

(MPLS). DiffServ is an improvement on IntServ, while MPLS is a packet-forwarding scheme having the advantages of both IntServ and DiffServ. In the following, we discuss IntServ and DiffServ service models.

6.13.1 Integrated Services

The Internet has emerged as a common platform for both real-time and non-real-time traffic and thus precludes the need to design a parallel infrastructure for real-time traffic. Use of existing Internet-layer protocol (e.g., IP) for real-time data is proposed so that economy of communication and interoperability between IntServ and non-IntServ networks can be realized. Integrated Services (IntServ) Model [2] includes two types of services targeted towards real-time traffic: guaranteed and predictive services. It integrates these two types of services, and it is designed to work well with multicast as well as unicast traffic. IntServ requires explicit resource reservation to be done, which, in turn, requires flow-specific states to be maintained in the routers.

IntServ is a per-flow based QoS framework with dynamic resource reservation. Routers reserve resources for specific traffic flows using RSVP [32, 35].

As stated earlier, IntServ has been designed to provide two types of services:

- **Guaranteed Service:**Guaranteed service provides an upper bound on end-to-end queuing delay. This service model is aimed to support applications with hard real-time requirements.
- **Predictive Service:**Predictive service can provide only statistical guarantees.

By using per-flow resource reservation, IntServ can provide per flow QoS guarantees. However, introduction of flow-specific states in routers represents a fundamental change to the Internet architecture. Another fact that makes an IntServ difficult to implement is that in the Internet backbone, where thousands of flows may be present, the router may find it impractical to maintain soft states for each flow. The important limitations of IntServ that deter its widespread use can be summarized as follows:

- The amount of state information increases proportionally with the number of flows. This places a huge storage and processing overhead on the routers. Therefore, this architecture does not scale well in the Internet core;
- The routers incur considerable processing overhead. All routers must implement RSVP, admission control, packet classification and packet scheduling techniques;
- Ubiquitous deployment of IntServ is required for guaranteed service. Incremental deployment of IntServ is difficult.

6.13.2 Differentiated Services

To overcome the limitations of IntServ, IETF proposed Differentiated Services (DiffServ) [1], which is simpler and more scalable. DiffServ redefines the Type of Service (TOS) field in the header of IPv4 or IPv6 traffic class byte as Differentiated Service (DS) field. The first six bits of the field are called DS Code Point (DSCP). The different markings of DSCP indicates the behaviour (known as per-hop behaviour or PHB) each router is required to apply to individual packets.

The information required by the buffer management and scheduling mechanisms is carried within the packet. Therefore, differentiated services do not require special signalling protocols to control the mechanisms that are used to select different treatments for the individual packets. Consequently, the amount of state information, which is required to be maintained per node, is

proportional to the number of service classes and not proportional to the number of application flows. This is a major improvement over IntServ.

Before a packet enters a DiffServ domain, its DSCP field is marked by the sending-host or the ingress router according to the QoS the packet is entitled to have. Within a DiffServ domain, each router only needs to examine the DSCP field to decide the proper treatment for the packet.

DiffServ is based on an important design principle. It tries to push complex processings associated with QoS provisioning to the network boundary. The network boundary refers to the application hosts, and the edge routers. Since a network boundary handles only a small number of flows, it can efficiently perform tasks such as packet classification and traffic conditioning. In contrast, core routers usually deal with very large number of flows, and should, therefore, only perform simple operations. This differentiation of the network boundary and the core routers is a vital part of the scalability of DiffServ.

Currently, DiffServ provides two service models besides the best-effort model: premium service model and assured service model. Premium service is a guaranteed peak rate service, which is optimized for regular traffic patterns and incurs almost no queuing delays. This model can provide absolute QoS guarantees. An example application of this service class, is a virtual leased line. A virtual leased line can eliminate the cost of having to maintain a separate network for an organization. Assured service class provides statistical guarantees to applications.

SUMMARY

- Real-time communication is different from traditional computer communication because the latter provides best-effort service, where as real-time communication networks can support the Quality of Service (QoS) requirements of applications.
- The QoS parameters that are usually considered by real-time communication networks include delay, jitter, loss rate, and bandwidth (throughput) requirements.
- Real-time communication protocols allow both non-real-time and real-time messages to be handled in the same framework.
- Controller Area Networks (CANs) are small networks spanning a few meters and use simple protocols such as a calendar-based protocol to guarantee the QoS requirements of applications.
- Real-time LANs are useful in industrial and automated home and office environments. The protocols used for supporting QoS guarantees are usually based on Ethernet or token ring protocols. For hard real-time communications, token ring protocols have natural advantages over Ethernet-based protocols.
- In the Internet environment, QoS requirements can be provided by using the techniques of QoS routing, resource reservation, traffic shaping and policing.
- IntServ and DiffServ are two important QoS models for use in the Internet domain.

EXERCISES

1. State whether you consider the following statements to be TRUE or FALSE. Justify your answer in each case.
   (a) Streaming compressed video transmission is an example of real-time VBR (variable bit rate) traffic.

(b) In a logical ring, the order of nodes in the ring must be the same as their order in the physical network.

(c) The virtual time protocol is an example of a bounded access type of scheduling in a multiple access network.

(d) In a bounded access token ring protocol using either the proportional or the local transmission time allocation schemes, the sums of the synchronous bandwidths ΣHi equals 1.

(e) Under normal real-time traffic conditions, the maximum delay suffered by the highest priority packets at a switch when a multi-level FIFO queue-based service discipline is deployed would be lower compared to the case when a framing-based service discipline is deployed.

(f) Real-time computer communication is essentially communication at high data rates.

(g) Under normal traffic conditions, a priority queue-based service discipline would incur less processing overheads at a switch compared to a multi-level FIFO queue-based one.

(h) In the virtual-time protocol, suppose the highest priority packet that a node needs to transmit at some instant of time is m. Then, for efficient working of the protocol, during priority arbitration interval the node should wait for $2 * \delta * m$ time units before starting transmission. (δ is the propagation delay in the network.)

(i) The countdown protocol should work successfully irrespective of whether the priority arbitration transmissions start with either the msb or lsb end of the priority value, as long as all nodes agree on the convention.

(j) Even when a global priority-based message transmission is supported by a real-time communication network, Rate Monotonic Algorithm (RMA) scheduling of real-time messages is ineffective.

(k) For transmitting VBR real-time traffic among tasks over a LAN, a calendar-based reservation protocol would yield higher GP(U) than either a global priority-based, or a bounded access type of protocol. (GP(U) is the guarantee probability at utilization U)

(l) In multiple bounding average rate characterization of bursty traffic, the larger the averaging interval the lower is the rate with which the source is bound.

(m) The (X_{min}, S_{max}) model for real-time traffic can be satisfactorily used for resource reservation to provide QoS guarantee for compressed audio and video signals.

(In the (X_{min}, S_{max}) model the minimum interarrival time between packets is bounded below by X_{min} and the largest packet size is bounded above by S_{max})

(n) Propagation delay primarily determines the end-to-end delay that a message might suffer while being transferred over a network.

(o) The maximum delay suffered by packets under a multi-level FCFS queue-based service discipline would be lower than that in a framing-based service discipline under similar traffic conditions.

(p) In any practical real-time communication protocol, we can expect that the guarantee probability (GP) to be close to 0 for utilization lower than the Average Breakdown Utilization (ABU) and GP would approach 1 as the utilization increases beyond ABU.

(q) If a certain token-passing network has a channel capacity of 100 mbps, then the IEEE 802.5 (priority-based protocol) would provide higher guarantee probability compared to IEEE 802.4 (timed token protocol) at very low channel utilizations.

(r) The countdown real-time global priority communication protocol should work successfully irrespective of whether the priority arbitration transmissions start with either the msb or lsb end of the priority value, as long as all nodes agree on the convention.

(s) Even when a global priority-based message transmission is supported by a real-time communication network, Rate Monotonic Algorithm (RMA) scheduling of real-time messages is not very effective.

2. Dynamic changes to the reserved resources is an important distinguishing characteristic of the RSVP protocol. Using four or five sentences explain how RSVP protocol is capable of dynamically changing the reservation status of a connection during the life time of the connection.
3. What do you understand by the term "hard real-time communication support by a network?" Give two example applications where hard real-time communication support from the underlying communication network is required. Give an overview of how hard real-time communication can be supported by a network.
4. What do you understand by the term "firm real-time communication support by a network?" Give two example applications where firm real-time communication support from the underlying communication network is required. Give an overview of how firm real-time communication can be supported by a network.
5. In an IEEE 802.4 network, if δ is the shortest deadline of all messages required to be transmitted over the network, identify the pros and cons of making TTRT equal to

   (a) δ (b) $\frac{\delta}{10}$ (c) $\frac{\delta}{2}$ (d) $2 \times \delta$
6. Answer the following in the context of a chemical manufacturing company that wishes to automate its process control application:
   (a) What problems might arise if an attempt is made to implement the chemical plant control software using the Ethernet LAN available in the factory?
   (b) How can a global priority protocol be supported in a LAN with collision-based access?
   (c) If RMA scheduling of packets is to be supported, what is the maximum channel utilization that can be achieved?
   (d) What are the main obstacles to efficiently implement RMA in this set up?
7. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit up to b_i kilobytes over each period of duration P_i milliseconds, where b_i and P_i are given in the table below. Assume that the propagation time is negligible compared to TTRT and that the network bandwidth is 1 mbps.

Node	b_i	P_i
$n1$	100 kb	10 mSec
$n2$	200 kb	15 mSec
$n3$	500 kb	100 mSec

   (a) Choose suitable Target Token Rotation Time (TTRT).
   (b) What is the maximum time for which a real-time message may suffer inversion?
   (c) Obtain suitable values of f_i (total number of bits that can be transmitted by the nodes n_1, n_2, and n_3 over every cycle) under proportional and local allocation schemes.
   (d) Determine the worst case times by which n_1 completes transmitting 100 kb of real-time message under each of the two bandwidth allocation schemes.
   (e) Determine the worst case time by which n_1 completes transmitting 100 kb of non-real-time message under each of the two bandwidth allocation schemes.
8. Consider a calendar-based reservation protocol to transmit real-time messages over a collision-based network such as a Controller Area Network (CAN):
   (a) Explain how transmission of asynchronous messages by nodes can be handled. (Asynchronous messages have probabilistic arrival times and do not have any specified time bounds).
   (b) Explain with proper reasoning the types of traffic for which a calendar-based protocol would perform satisfactorily and the types for which it will not.

9. The bandwidth of a priority-based token ring network (IEEE 802.5) is 100 mbps. Assume that the propagation time of the network is 10 mSec. Determine the fraction of bandwidth wasted during every frame transmission, when the frame size is 2 kbps (kilobytes per second) Under what conditions would the wasted bandwidth be zero?
10. Explain two traffic specification models that can be satisfactorily used to specify bursty traffic.
11. Identify the factors which contribute to delay jitter in real-time communications in packet-switched networks. Assume that a certain real-time application receives data at the rate of 10 mbps. The QoS guarantee to the application permits a delay jitter of 20 mSec. Compute the buffer requirement at the receiver.
12. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit up to b_i bits over each period of duration P_i milliseconds, where b_i and P_i are given in the table below.

Node	b_i	P_i
N_1	1 K	10,000
N_2	4 K	50,000
N_3	16 K	90,000
N_4	16 K	90,000

Choose suitable Target Token Rotation Time (TTRT) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycle). Assume that the propagation time is negligible compared to TTRT and that the system bandwidth is 1 mbps.

13. If you are constrained to use an existing Ethernet LAN for a factory automation application, what are the different protocols that you can use to support communicating real-time tasks? Which of the alternatives that you have identified would work best? Give your reason.
14. Consider a 10 mbps token-ring network operating under the priority-based protocol (IEEE 802.5). For this network, the walk time is 2 mSec and the frame length is 1024 bytes. Determine the fraction of wasted bandwidth due to the wait time required for the token to return to the transmitting node after transmission of a frame is complete. At the given bandwidth would a bounded access protocol perform better? Explain your answer.
15. Consider a 10 mbps token ring network. The walk time is 1 mSec and the frame size is 512 bytes. Determine the maximum time for which a message may undergo priority inversion under IEEE 802.4 and IEEE 802.5 protocols.
16. (a) Identify at least two factors which contribute to delay jitter in real-time communications and explain how they cause jitter.

    (b) What is the difference between execution time and response time of a task? In what circumstances can they be different?
17. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit up to b_i bits over each period of duration P_i milliseconds, where b_i and P_i are given in the table below.

Node	b_i in kilobytes	P_i in milliseconds
N_1	4	10
N_2	10	50
N_3	10	90
N_4	20	100

Choose suitable Target Token Rotation Time (TTRT) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycle). Assume that the propagation time is 1 mSec and that the bandwidth of the network is 10 mbps.

18. A (σ, ρ) traffic characterization of a certain packet switched real-time traffic is characterized by a peak traffic (σ) of 100 packets and average traffic rate (ρ) of 10 packets per second. The packet size is 512 bytes. Assuming that the packets undergo a maximum queuing delay of 50 mSec at each of the switches, what would be the traffic characterization after the 10th switch on the route?
19. Consider a 10 mbps token-ring network operating under the priority-based protocol (IEEE 802.5). For this network, the walk time is 2 mSec and the frame length is 1024 bytes. Determine the fraction of wasted bandwidth due to the wait time required for the token to return to the transmitting node after transmission of a frame is complete. At the given bandwidth would a bounded access protocol perform better? Explain.
20. Consider a 10 mbps token ring network. The walk time is 1 mSec. The frame size is 512 bytes. Determine the maximum time for which a message may undergo priority inversion under IEEE 802.4 and IEEE 802.5 protocols.
21. Consider the use of timed token protocol (IEEE 802.4) in the following situation. We have four nodes in the system. The real-time requirement is that node N_i be able to transmit up to b_i bits over each period of duration P_i milliseconds, where b_i and P_i are given in the table below.

Node	b_i in kilobytes	P_i in milliseconds
N_1	4	10
N_2	10	50
N_3	10	90
N_4	20	100

Choose suitable Target Token Rotation Time (TTRT) and obtain suitable values of f_i (total number of bits that can be transmitted by node N_i over every cycler). Assume that the propagation time is 1 mSec and that the bandwidth of the network is 10 mbps.

22. Would it be advisable to use an Ethernet LAN in a hard real-time application such as factory automation? Justify your answer. Evaluate the pros and cons of using an Ethernet-based protocol in such an application.
23. Suggest a scheme that can help handle aperiodic and sporadic messages in a reservation (calendar-based) protocol without making major changes to the protocol.
24. Identify the factors which contribute to delay jitter in real-time communications. Assume that a certain real-time application receives data at the rate of 10 mbps. The QoS guarantee to the application permits a delay jitter of 20 mSec. Compute the buffer requirement at the receiver.
25. In a real-time packet-switched network, explain the roles of a traffic policer and a traffic shaper. Explain one popular traffic shaping and one traffic policing technique.
26. Explain why traffic gets distorted in a packet-switched network and how traffic reshaping is achieved for providing QoS guarantee.
27. Compare the performance of IEEE 802.4 protocol with IEEE 802.5 protocol for real-time applications at high, medium, and low bandwidths.
28. Explain using four or five sentences, how global priority arbitration is achieved using virtual time protocol. At which ISO layer would such a protocol operate?
29. Suppose we have three periodic messages m_1, m_2, and m_3 to be transmitted from three stations S_1, S_2, and S_3, respectively, on an FDDI network. Let the Token Rotation Time (TTRT) be 10 mSec and the walk time be 1 mSec . The transmission time (C_i mSec) and the period of the messages (T_i) are as fol-

lows: (C_1 = 9 mSec, T_1 = 110 mSec); (C_2 = 12 mSec, T_2 = 150 mSec); (C_3 = 15 mSec, T_3 = 160 mSec). Determine the synchronous bandwidth that needs to be allocated to each station for successful transmission of the messages.

30. Explain the important shortcomings of the IntServ architecture in supporting real-time communication. How does DiffServ overcome it?
31. Compare the advantages and disadvantages of using a ring network versus a collision-based network for real-time communication.
32. What is the difference between a work conserving and a non-work-conserving service discipline? Explain how a non-work-conserving service discipline can help in controlling traffic distortion.
33. What do you understand by QoS routing. Explain the different types of QoS routing algorithms.
34. Explain what are the different QoS constraints that are considered during QoS routing. Explain the features of these constraints based on which the constraints can be classified. What are the implications of these features on the routing algorithms?
35. What do you understand by QoS routing? Give some examples of additive, multiplicative, and concave constraints that are normally considered in QoS routing schemes. Explain how these features are considered in QoS routing protocols.
36. Explain the difference between traffic shaping and policing. Name a traffic shaping and policing protocol and briefly describe its operation.
37. What is a Controller Area Network (CAN)? Name a few applications that are based on CAN. Explain a real-time communication protocol that can be used in a CAN.
38. Explain how soft real-time communication with statistical delay guarantees can be provided in an Ethernet LAN environment using rate-adaptive traffic smoothing technique.
39. Explain the different elements that play a role in provisioning QoS guarantees to applications in the Internet.