CHAPTER 2

Networking: Functional Elements and Current Practice

Corporations, businesses, banks, government agencies, medical establishments, and educational and research institutions have begun to rely heavily on distributed information applications for storing, transporting, and accessing data, for distributed computing, for telemetry and remote control, and for communicating by audio and visual media. Communication networks provide the transport infrastructure that carries the information flows between such distributed information applications.

In this chapter we begin by developing a three-layered view of communication networks. We delineate the subject matter of this book—networking—as dealing with the problem of sharing the resources of a bit-carrier infrastructure. We then identify the basic functional elements of networking: multiplexing, switching, routing, and network management. Then, in the first part of the chapter, we discuss the functions that are carried out by each of these elements. This is followed, in the second part of the chapter, by a discussion of current practice in communication networks. In this part, we first describe the dominant bit-carrier technologies: optical fiber backbones and cellular wireless access networks. Next, we describe the seven-layer Open Systems Interconnection (OSI) architecture, which has traditionally been used to conceptualize communication networks, although it is not always strictly followed. We then provide an overview of telephone networks (the most widely deployed communication networks), followed by the two dominant packet networking technologies: the Internet and Asynchronous Transfer Mode (ATM) networking. For completeness, a brief discussion of X.25 networks, Frame Relay networks, and Integrated Services Digital Network (ISDN) is also provided. These are precursors to today's networks, and many of the concepts that have now matured had their inception in these technologies.

The remainder of the book is organized into three parts: Part I on multiplexing, Part II on switching, and Part III on routing. This chapter deals

with functional and architectural issues, and Chapters 3, 9, and 13 discuss the performance and engineering issues associated with multiplexing, switching, and routing Chapters 3, 9, and 13 are the first chapters in their respective parts.

2.1 Networking as Resource Sharing

The points at which distributed information applications connect to the generators and absorbers of information flows can be viewed as *sources* and *sinks* of traffic (see Figure 2.1). Examples of traffic sources are telephone instruments, video cameras, or a file on a computer disk that is being transmitted to another location. Examples of traffic sinks are telephone receivers, television monitors, or computer storage devices.

To focus on the emphasis of this book, we explode Figure 2.1 into the layered view shown in Figure 2.2. In this view, the sources and sinks of information and the distributed applications connect to the communication network via common *information services*. The information services layer comprises all the hardware and software required to facilitate the necessary transport services and to attach the sources or sinks to the communication network—for example, voice coding, packet buffering and playout, and voice decoding (for packet telephony), or mail preparation and forwarding software (for electronic mail), or a browser (for the World Wide Web).

The communication network is built from the raw material of *communication links*. Information covers geographical distance by being carried over such links. The design and fabrication of such links involve the consideration of electromagnetic propagation in various media, transducers, modulation schemes, error-control coding, and physical interfaces. Modern communication

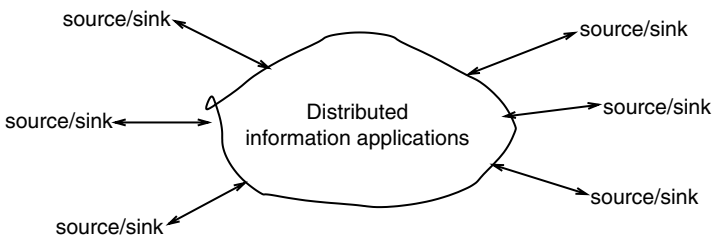

Figure 2.1 Sources and sinks of information attached to distributed information applications; *source/sink* is a device that can be a source or a sink or both.

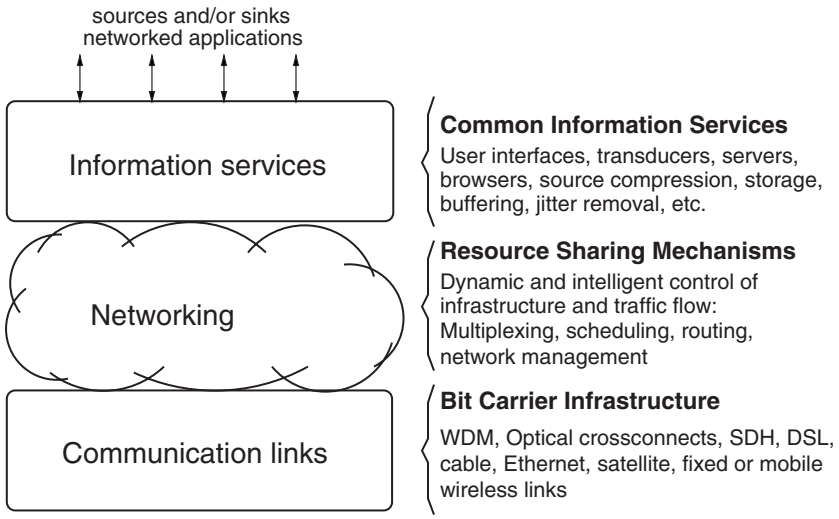

Figure 2.2 A three-layered view of a communication network. Networking is concerned with resource-sharing mechanisms that efficiently share the bit-carrier infrastructure and control the quality of service provided to the various applications using the network.

networks are largely digital, and the information they carry is transported as digital data; hence the term *data communication networks*. Thus, from the viewpoint of communication network engineering, or *networking*, we will view the communication links simply as *imperfect bit-pipes*, the imperfection being that the bit-pipes can delay, lose, or modify the information they carry. (Although this abstraction suffices for most of the book, when we turn to wireless networks in Chapter 8 we provide an understanding of digital communication over mobile wireless channels.) In the right side of Figure 2.2 is a list of several bit-carrier technologies over copper, fiber, and wireless media. Given the basic raw material of communication links, the next task in developing a communication network is to interconnect several links to form a bit-carrier network. In this book we do not consider the problem of the design of physical network topologies (i.e., the placement of network nodes and the choice of which nodes to interconnect and with what link speeds); we implicitly assume that a network topology is given.

Residing between the information services and the bit-carrier network is the set of functions that constitute the subject of *networking*, as depicted in Figure 2.2.

Ideally, each instance of an information flow from the information services layer can expect from the network a certain *quality of service* (QoS) (e.g., in terms of guaranteed or statistical bounds on service denial, information loss, throughput, and delay). The bit-carrier network is an expensive resource and must be efficiently shared among the arriving information transport demands so that each demand obtains its required QoS. Thus, as shown in Figure 2.2, the broad functionality that networking is concerned with is *resource sharing*. The resources of the bit-carrier infrastructure must be shared between the contending information flows so that the flows obtain their required QoS and the resources are utilized efficiently. Even if a networking technology does not offer per-flow QoS, there would still be the requirement of efficient sharing of the bit carrier resources so as to maximize the traffic carried, subject to some aggregate performance objectives.

We note that when a communications engineer designs a communication link, the primary quality objective is to obtain the highest possible bit transmission rate with an acceptable bit error rate, given the various physical and resource constraints. In designing a network out of the raw material of communication links, however, we need to become aware of the QoS requirements of the flows that will be carried in the network. Some flows are sensitive to delay but can tolerate loss, whereas others may require reliable transfer and high throughput. Thus, in this sense, networking is concerned with information flows rather than only bit flows. High-quality bit carriers, if integrated into a poorly designed communication network (as seen by an information flow), can lead to poor overall performance. On the other hand, awareness of the limitations of link quality in the process of network design can lead to better utilization of the bit carrier infrastructure while satisfying the QoS requirements of the various flows to be carried in the network. The latter is especially true when wireless links are involved.

Analogy with the Operating System of a Computer

To readers familiar with computer systems, the analogy depicted in Figure 2.3 provides another perspective on the point of view presented here. A computer system has several hardware resources (such as the processor(s), the memory, the secondary storage, and the various peripherals such as displays and printers). In a multitasking computer, several applications could be running simultaneously, thus needing to share these resources. The operating system contains the various resource-sharing algorithms that permit the applications to efficiently share the system resources. Networking occupies much the same position between the bit carriers and the networked applications. This is shown on the left side of Figure 2.3. Distributed network algorithms (often called *protocols* in the networking jargon) implement the resource-sharing mechanisms. Just as operating systems have

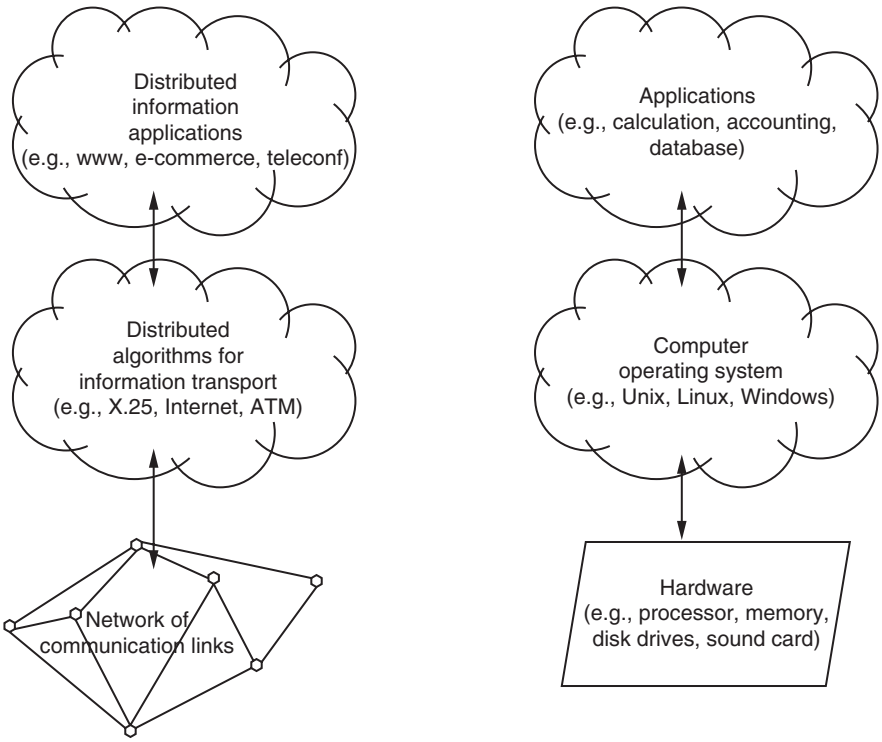

Figure 2.3 Networking is concerned with distributed algorithms for efficient sharing of the resources of a bit-carrier network, in much the same way as a computer's operating system helps computer applications to use and share the hardware resources of a computer.

evolved over time, and just as there could be several competing computer operating systems, so are there various networking technologies.

2.2 The Functional Elements

When a demand arrives for the transport of a bit stream between points in a communication network, we must decide on which set of links of the physical bit-carrier network this demand should be *routed*, and on those links how the bit stream of this demand should be *multiplexed* with the existing bit streams. At the nodes where links meet, the bit stream must be *switched* from one link to the other.

All this must be done so that the QoS of the existing demands is not disturbed and the QoS of the new demand is met. In principle, it is possible for the system to determine the best way to accept the demand by evaluating all possible routes in the network and a variety of possible service strategies at each hop on each route. For example, this could be done so as to maximize the amount of traffic carried in the network in the long run. In practice, however, not all these decisions are made for each demand arrival; this would be impracticable. Instead, a *divide-and-conquer* approach is followed. For example, routing can be taken to be fixed between each pair of points. Even the multiplexing strategy can be specified at each link; for example, the strategy could be to give strict nonpreemptive priority to interactive voice calls and to put all other packets into a single queue. When a new demand arrives along a route, the decision can simply be whether to accept or reject the demand; in some strategies it may be necessary to reparametrize the multiplexers along the path of the demand. The routing can then be optimized so as to maximize the load-carrying capacity of the network. One can also study the effect of different multiplexing strategies on the network capacity.

Apart from the three activities of multiplexing, switching, and routing, networks invariably have *network management*, which can be viewed as procedures for monitoring the network and reporting actual performance (in comparison with performance objectives), and for taking care of situations "not engineered for."

In the remainder of this section we elaborate on the issues involved in these activities, and in the next section we provide an overview of how current networks address these issues. We conclude this chapter with some perspectives. For completeness, we discuss network management briefly in this chapter, and this topic is not covered further in the book.

2.2.1 Multiplexing

At any instant in time, several information flows must be carried by a network. Given the routing, the task of systematically merging several flows into a network is called multiplexing. We obtain the multiplexing effect at the network level by using multiplexing techniques at the individual links in the network. After the multiplexing at an individual link has been defined and analyzed, we can put together a network of links and study the overall effect of the multiplexing techniques implemented at each of the links. Multiplexing is the subject of Part I of this book.

There are broadly two classes of multiplexing techniques: circuit multiplexing and packet multiplexing.

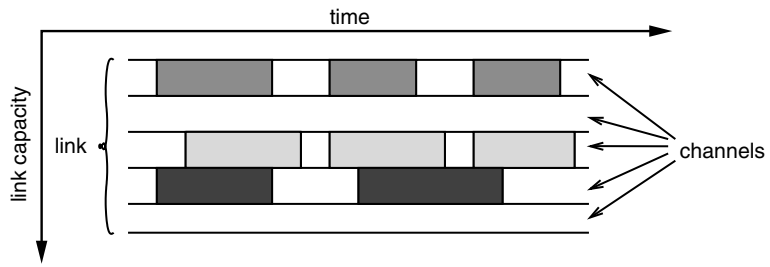

Figure 2.4 Circuit multiplexing: static partitioning of a bit-pipe.

In *circuit multiplexing*, there is a statically partitioned allocation of the capacity (i.e., bit transmission rate) of the bit-pipe to the various flows. This is depicted in Figure 2.4, where a link's transmission rate (depicted as the space between the two thick lines) is partitioned into several *channels* (five in the figure). In practice, the link rates and channel rates are not arbitrary but rather conform to standards that specify the possible set of link rates and the channel rates into which a link can be partitioned. For example, the International Telecommunications Union (ITU) specifies the E-1 digital link, which has a raw bit rate of 2.048 Mbps, of which 1.920 Mbps can be used to carry user data; this rate can be divided into 30 channels, each one 64 Kbps. If a link is circuit multiplexed, each *conversation* (or *flow*) is allocated to a channel and holds the channel for its entire duration.

Most traffic flows do not generate data at a sustained constant rate but instead typically send data in bursts. As shown in Figure 2.5, all the common sources of traffic have such behavior. This point is also reflected in Figure 2.4, where it is shown that the channels are occupied by bursts of activity, with idle periods in between. A circuit-multiplexed link transmits the data from a source at a fixed rate allocated to the source at *connection setup* time. If all the channels in the link are busy and another request arrives, such a request is *blocked*. Thus the main performance objective when engineering a circuit multiplexed link is the *probability of blocking* requests for channels. If the link handles different *classes* of flows, each with its own blocking objective, then the design of a channel allocation strategy becomes an important problem.

In contrast, in a packet-multiplexed link the entire bit rate of the link is applied to the service of any data that arrive from the sources. We see from Figure 2.6 that this results in the bursts of data from the various sources being interleaved into the link. Comparing this figure with Figure 2.4, notice that the

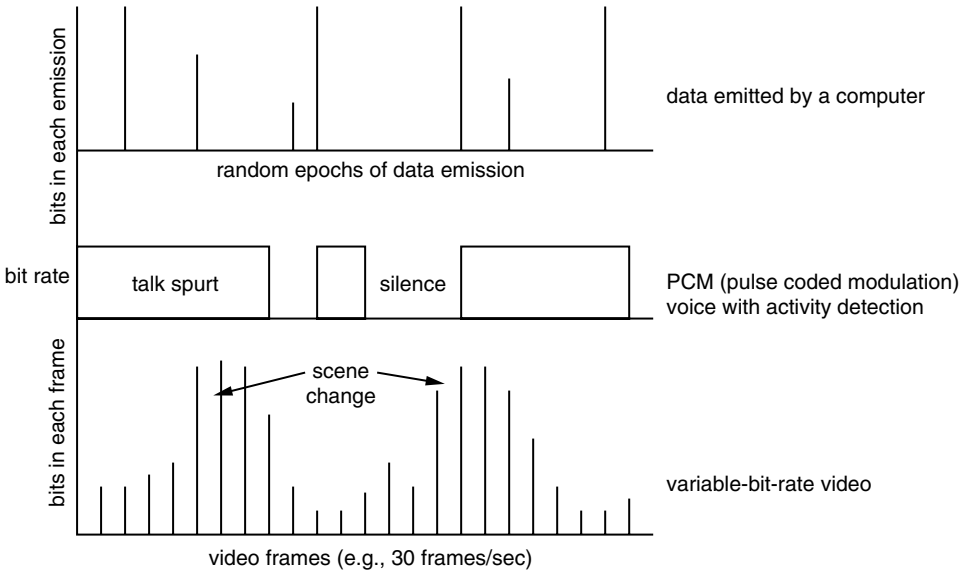

Figure 2.5 Traffic flow from sources is typically not smooth, but *bursty*.

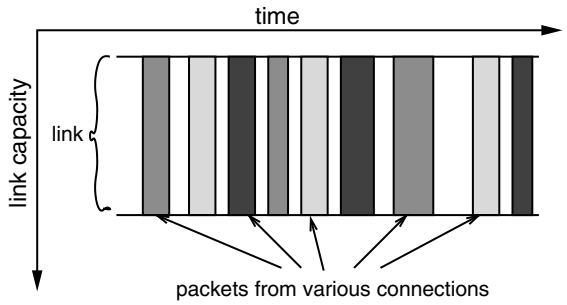

Figure 2.6 Packet multiplexing: no partitioning of the bit-pipe.

bursts occupying the link are shown to be shorter, because they are transmitted at the full link rate. The sources emit information in chunks of bits (*packets*). Intuitively, it is clear that the average rate at which the sources, being multiplexed into the link, emit data must be less than the bit rate of the link. On the other hand,

the total *peak* rate at which the sources can emit data typically exceeds the link's bit rate. At such times, the excess data arriving to the link must be *queued*, and such queueing causes data to be *delayed*. If the space available to buffer the arriving data is inadequate, then there could also be data *loss*. From a modeling perspective it is common to view the link as a bit "server" that serves the "customers" waiting at the packet queue. It follows that, when we design a packet multiplexer, the performance objective is usually a delay or loss objective; for example, the mean delay should be less than some given value, or the probability of the delay exceeding a required bound should be small, or the probability of packet loss should be no more than a small value.

In a circuit-multiplexed network, a flow in the network is identified at setup time by the sequence of channels it is assigned to. With packet multiplexing, however, there is additional *overhead*, because each packet must carry a *header* (and perhaps a *trailer*) that identifies the flow to which the packet belongs.

Whereas circuit multiplexing is essentially only the one concept, packet multiplexing has many variations that arise depending on the networking context. Figure 2.7 shows several computers and terminals interlinked by a

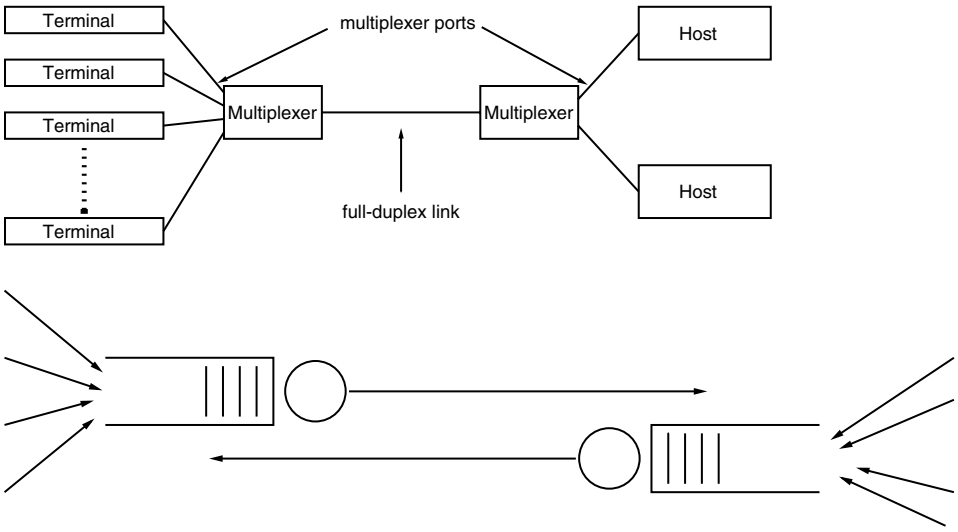

Figure 2.7 In centralized packet multiplexing, the multiplexers have full control over the link's transmission rate. The bottom part of the figure shows a queueing schematic of the traffic being multiplexed into each end of the link.

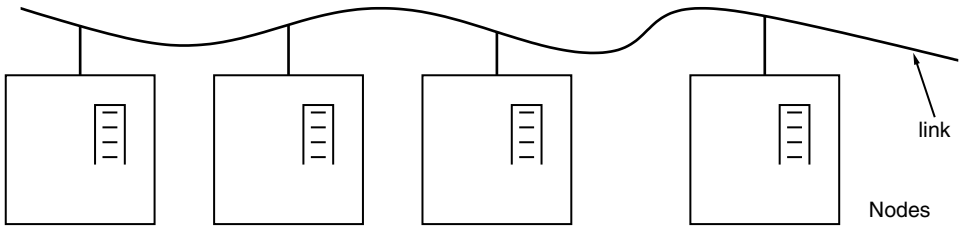

Figure 2.8 In distributed packet multiplexing, the various sources share the link in a distributed fashion with no central control.

point-to-point bit-pipe. A multiplexer at each end "stuffs" packets (destined for the other end) into the link. Notice that each multiplexer has full control of the link in the direction in which it sends packets. The packet *scheduler* at the multiplexer can make decisions about which packet to send next into the link, depending on the various QoS objectives. We can call such an arrangement *centralized* packet multiplexing.

In contrast, consider the situation depicted in Figure 2.8. Several computers, or other computerized devices (which we will call *nodes*), are attached to a *multipoint* link at various points along its length. This is physically possible for a wired link by means of a device called a *tap*, at which a transmitter of an attached computer can inject electrical energy that propagates over the wire. The same figure can also be a depiction of a wireless communication setup. The common feature is that if a station transmits, all other stations can potentially receive variously attenuated versions of the transmitted signal. Obviously, if the sharing of the link is completely uncoordinated, then, owing to *collisions*, even under moderate load most of the transmissions will be wasted, leading to very poor effective transmission rates, as seen by the sources and sinks of traffic.

The simplest "coordination" mechanism is called *random access*, in which nodes attempt to transmit and try to determine whether their transmissions succeeded; if not, they wait for random amounts of time before reattempting. Such random *back-off* simply reduces the probability of a collision, in case the failure of transmission was the result of a collision. In a wired network, a collision can be sensed because the transmitter can measure the power on the medium as it transmits and thus can determine whether the power level exceeds a certain threshold (indicating that someone else is transmitting as well). In a wireless network, the idea is to very carefully avoid collisions by sensing the channel for

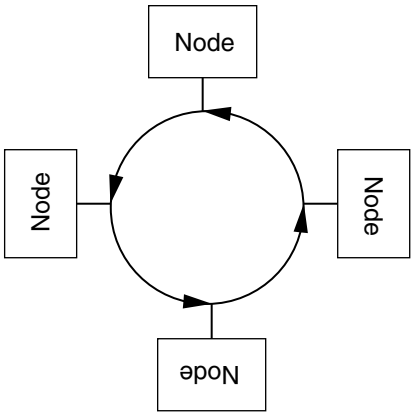

Figure 2.9 Polling-based distributed packet multiplexing; the token ring architecture.

activity before transmitting, and attempting to reserve the channel for the duration of a transmission.

A more coordinated distributed packet multiplexing approach derives from the basic concept of *polling*. Figure 2.9 depicts a *token ring network*, in which a token is passed around and possession of the token permits a node to transmit. Yet another approach is for a master station to poll each of the other stations and thus give turns for data transmission between each station and the master or between the stations themselves.

Figure 2.10 shows a satellite network in which there are several (geographically widespread) small terminals (attached to individual computers or the local networks of small organizations), and these share a satellite link to a large hub. The channel *inbound* from the terminals to the hub needs to be shared. An important difference between this distributed multiplexing situation and the ones discussed earlier is that there is a large *propagation delay* across the link, owing simply to the fact that electromagnetic waves travel at a finite speed (a rough value is 0.25 second each way, for a link using a single geostationary satellite hop). Contention- or polling-based access in such a situation can waste a lot of link transmission rate, because the overheads of these strategies grow with the propagation delay. This is because in a contention mechanism, when a collision occurs it takes time of the order of the propagation delay to detect and resolve the collision, and in polling a whole propagation delay's worth of time is wasted

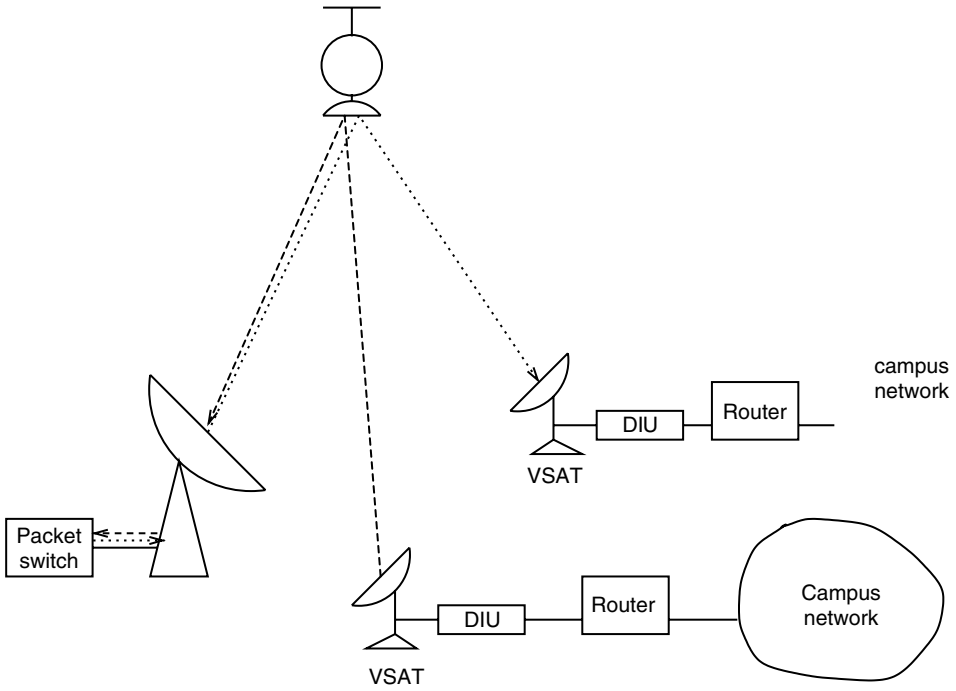

Figure 2.10 Reservation-based distributed packet multiplexing; a very small aperture terminal (VSAT) satellite network. A DIU (digital interface unit) connects the radio frequency equipment on the satellite side to digital equipment on the customer's network.

each time an idle terminal is polled. Instead, the terminals request reservations of time on the inbound channel. The reservations are granted on the outbound channel (which, as we can see, is centrally packet multiplexed at the hub). when the reservations are granted, the terminals take turns accessing the inbound direction of the link. Reservations are periodically renegotiated over durations referred to as *frames*.

The situations depicted in Figures 2.7–2.10 are the simplest possible; in general, there could be a network with multiple links, each of which could have its own multiplexing strategy. Devices interconnecting the links would be capable of performing the appropriate multiplexing on their respective interfaces.

Figure 2.11 provides a summary depiction of the taxonomy of link multiplexing that we have presented. It is also possible to create hybrid multiplexing schemes that combine circuit multiplexing and packet multiplexing. The approach is depicted in Figure 2.12. The link's transmission rate is partitioned

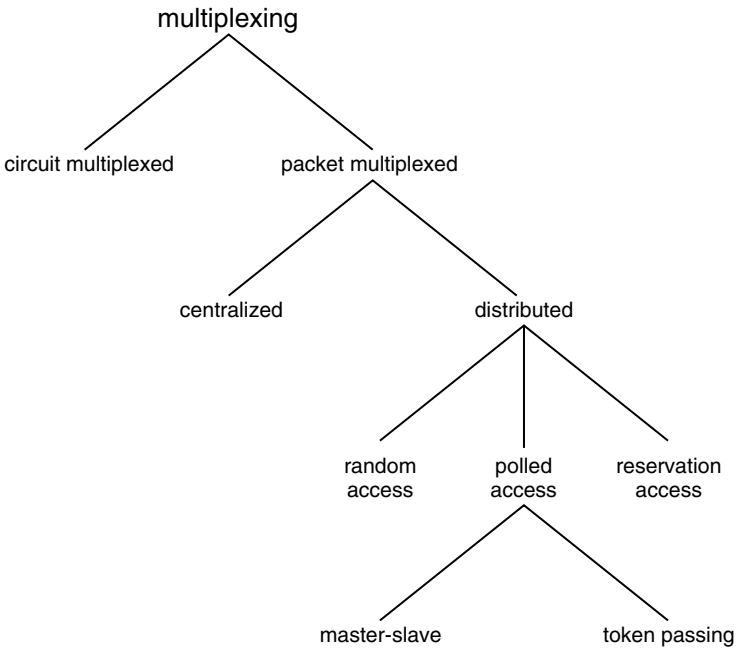

Figure 2.11 A summary view of a link multiplexing taxonomy.

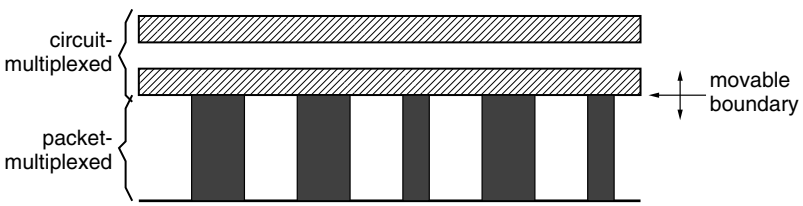

Figure 2.12 Hybrid link multiplexing; combining circuit multiplexing and packet multiplexing on a link.

into several digital channels; some of them can be used in the circuit-multiplexed mode, and the others in the packet-multiplexed mode. As Figure 2.12 shows, in general, the boundary between the circuit-multiplexed part and the packet-multiplexed part can be dynamically varied. An important example is the Integrated Services Digital Network (ISDN) service, in which a telephone network subscriber can use the twisted pair telephone wire to access circuit-switched telephony as well as packet-multiplexed network services. For example, let us consider the 2B+D ISDN interface between a user and an ISDN access node; here, on a telephone wire, one carries two 64 Kbps B (or bearer) channels and one 16 Kbps packet D (or "data") channel. The D channel is used for packet signaling between the user and the network. The B channels can be used in various ways. For example, the user can set up a circuit-multiplexed phone call on one channel and a fax call on the other channel. Alternatively, the user can use one channel for a phone call while simultaneously using the other channel for accessing a packet-multiplexed network. For a more detailed discussion of ISDN technology, see Section 2.3.3.

Many packet-multiplexed networks also use *virtual circuit multiplexing*, in which the path for the packet flow between the source and destination is decided by a call setup procedure, very much like that in circuit-multiplexed networks.

2.2.2 Switching

Typically, a data flow will need to traverse more than one link. To see why, consider a network of N sources or sinks. Assume only *unicast* flows, that is, each flow has only one source and one destination. A maximum of $N(N-1)$ flows are possible, and in the general case, the network must be capable of supporting each of these flows by providing a possible path for each of them. To support all possible flows, a brute force approach would be to construct a fully connected network with $N(N-1)/2$ duplex transmission links, as shown in Figure 2.13a. In such a network, the number of links increases in proportion to N^2 and is obviously economically infeasible. Furthermore, a large number of these links would be very long, and many of them would also probably never be used. A more economical arrangement would be to have a centralized network in which a central node collects the data flows from the sources and then distributes them to the appropriate sinks. Such a network uses only N transmission links and is shown in Figure 2.13b. This can also become infeasible for large operational areas because of the long transmission links that will be required. However, this procedure can be repeated recursively using subsets of N to construct a network in which the path from a source to a sink involves multiple links or hops. Such a network is shown in Figure 2.13c.

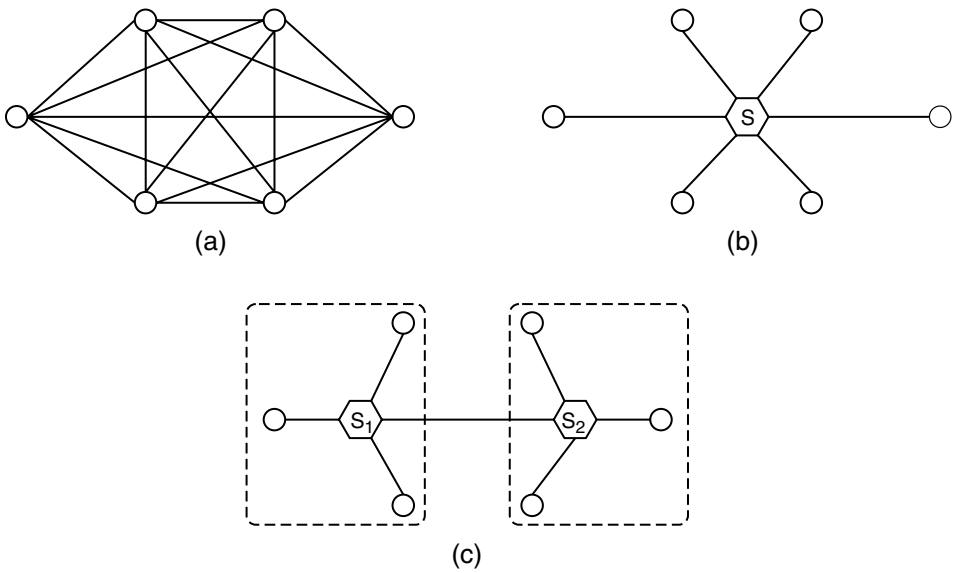

Figure 2.13 A six-node network constructed in three ways: (a) a fully connected network; (b) a centralized network with every node connected to a central node that selectively establishes paths between the communicating nodes; (c) a hierarchical network with two networks connecting nodes in a smaller area.

Observe that the link between S_1 and S_2 in Figure 2.13c is now a shared resource among the nodes of the network and will be made available to the nodes by the switches on demand. The capacity of this link can be designed so as to satisfy a large fraction of the demands rather than every demand. With such a design criterion, the capacity of the links can be significantly lower than that required to satisfy every demand. Also, because the links interconnecting the subnetworks will be shared among a large number of users, the sizing of the capacity of these links is a multiplexing problem.

For simplicity, the preceding discussion makes the assumption that the link cost is proportional to the distance. In practice, the link costs depend on many factors, such as the terrain, the medium, the bandwidth, and so on, and is not strictly proportional to the length of the link nor the transmission rate on it. A generally accepted notion is that longer links are more cost-efficient when the transmission rates are high.

The hierarchical network of Figure 2.13c achieves many objectives. Building smaller networks around a central node makes them more manageable. The links between different subnetworks will be shared by a larger number of users, thereby increasing its usage efficiency. At the higher levels of the hierarchy, the links will require higher bandwidths because they interconnect larger groups of users and may also traverse longer distances. This also works out well because, as we have said before, the cost of a transmission link is a sublinear function of its capacity and distance, and high-capacity links are more economical over longer distances. By appropriately designing the interconnections, we can improve network reliability and avoid single points of failure, as in the networks of Figure 2.13b and 2.13c.

From this discussion, we can define a switch as a device that sits at the junction of two or more links and moves the flow units between them to allow the sharing of these links among a large number of users. Figure 2.14 shows such a block diagram view of a switch. A switch allows us to replace transmission links with a device that can switch flows between the links. The construction of the switch can be quite complex. However, this is a fair trade (between transmission and switching) because, traditionally, transmission capacity is significantly more expensive than switching capability.

The switch functionality is achieved by first *demultiplexing* the flows on each incoming link, *identifying* the output link for the flow, *forwarding* or *switching* the element of each flow to its destination output link, and then *multiplexing* the outgoing flows onto each output link. These are very fast timescale functions and must be performed on every packet or slot of a *time-division-multiplexed* TDM frame. In addition, in circuit-multiplexed networks, the switches must process call or flow requests. These are slower timescale functions and include performing the necessary signaling and, if available, reserving resources to maintain the call. Although it is rarely done, resource reservation is also possible in packet-multiplexed networks, with a resource reservation phase preceding the actual flow of data. In both circuit- and packet-multiplexed networks, the switches

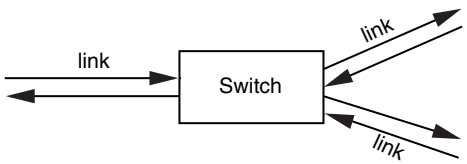

Figure 2.14 A switch moves data between links.

also must exchange information about the network and switch conditions and must calculate routes to different destinations in the network. These functions are typically performed on even slower timescales of a few minutes to hours, the timescales at which the traffic characteristics are expected to change.

We detail these functions with reference to packet and circuit switches separately. First, consider a packet switch, such as a router in the Internet or a cell switch in an ATM (Asynchronous Transfer Mode; see section 2.3.6) network. The packet lengths could be fixed as in the ATM network (where they are 53 bytes long), in which case the input links can be assumed to be slotted in time, with the length of the time slot equal to the packet transmission time. The packet lengths could also be variable, as in IP networks, where we must consider the links to be unslotted. Typically, in this case there will be upper and lower bounds on the packet lengths. A *store-and-forward* packet switch waits until the entire packet is received before processing it, whereas a *cut-through* packet switch starts processing the packet header as soon as the header is received. In the latter case, the transmission on the output port could start before the entire packet is received on the input port. The advantages of cut-through switching are limited, and it is rarely implemented in practice.

The components of a packet switch are shown in Figure 2.15. A line interface extracts the packet from the input link by appropriately identifying the

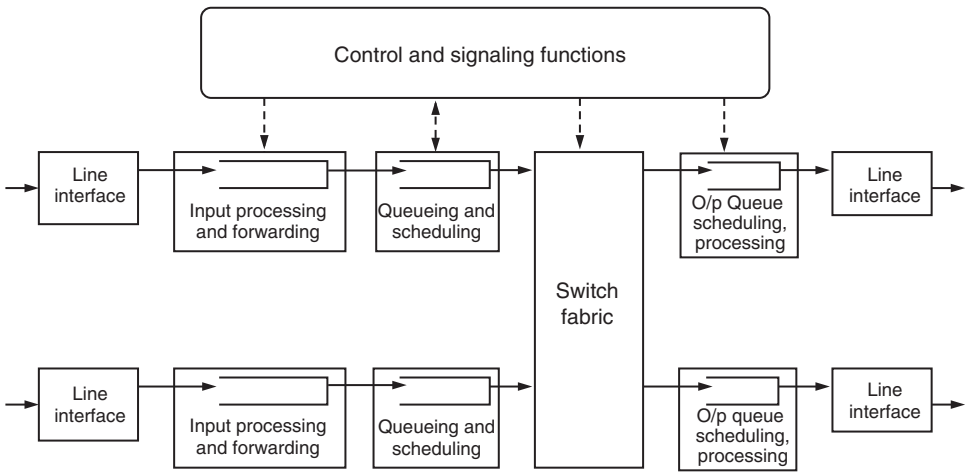

Figure 2.15 The components of a packet switch.

boundaries of the bits and of the packet. An input processor then extracts the header, and an associated forwarding engine performs a *route lookup* using this header information by consulting a *routing table* to determine the output link. This is essentially the demultiplexing function mentioned earlier. In a multiservice network, the service type of the packet is also determined at this stage to identify the kind of service that the packet is to be provided. The type of service determines the scheduling of the packet's transmission on the output link and drop priorities during periods of congestion. If it is not possible to send the packet to the output port immediately, it is queued and scheduled to be moved to the output port according to its service type. The switch fabric next moves the packet to the output queue. An output processor determines the queue position and *schedules* its transmission on the output link. Finally the packet is transmitted on the physical interface by the output line interface.

In addition to processing packets, other functions are performed by the switch. In virtual circuit-based packet multiplexed networks, such as ATM, Frame Relay, and X.25 networks, the actual information flow is preceded by a call setup phase. In this phase, the path for the packets corresponding to the information flow is set up much as in the case of circuit multiplexing. During this phase other parameters of the call, such as the bandwidth to be reserved for the call and the permissible delays and loss rate on the packets of the flow, can also be determined. This is very similar to call setup in a telephone network except that many more parameters for the call are negotiated, and information flow is in the form of packets and perhaps of variable rate. In datagram packet switching, the path for the packets from the flow is not explicitly set up. The destination port is decided separately for each packet by consulting a routing table, and, in principle, it is possible that different packets from the same flow follow different paths through the network. This kind of stateless, next-hop packet switching is used in the Internet. In packet-multiplexed networks, routing and other signaling protocols and the routing algorithms must be executed so that the network can construct the routing table that determines the output port for a packet. All these functions are shown as control and signaling functions in Figure 2.15. Note that the output of the control and signaling block interacts with all the other packet processing blocks in the switch.

In Chapters 9–12 we discuss the architectures and design choices for packet switches and also the details of some of the functional blocks in the switch.

Next, let us consider a switch in a circuit-multiplexed network—specifically, that in a telephone network. The telephone network has been using TDM-based digital communication channels beyond the local exchange for quite some time now, and we will assume that is the case. To understand what a switch in this

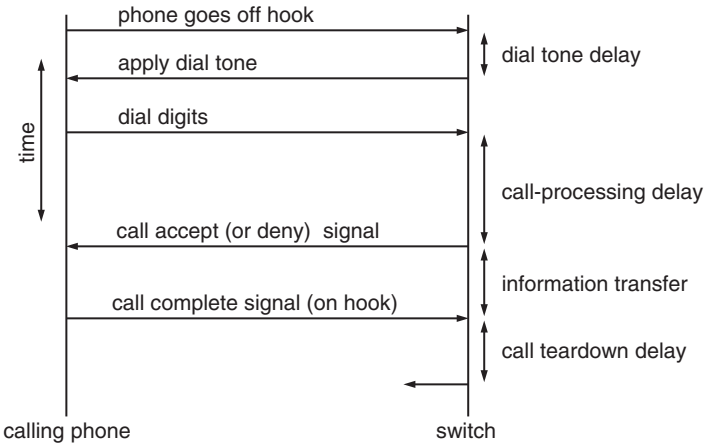

Figure 2.16 Call setup procedure and the delays in a switch in a circuit-multiplexed network. Call-processing delay is also called postdial delay or ringing delay.

network does, consider the call setup with the sequence of events shown in Figure 2.16. When the calling phone goes off hook, the condition is recognized by the switch in the local exchange and a dial tone is presented to the calling phone, indicating readiness of the switch to accept digits. The dialed digits are read by the exchange and analyzed to determine the destination of the call. The routing algorithm is used to determine the path to be assigned to this call in the network. This could involve an exchange of signaling messages with other nodes in the network. If a path is available, it is reserved for this call; if it is not, the call is dropped and assumed lost forever. From the point of view of the switch, a path is determined by the two pairs "input port:TDM slot" and "output port:TDM slot." After the path is established in this manner, the switching fabric is programmed to move the contents of the "input port:TDM slot" to "output port:TDM slot" in every frame. This is shown in Figure 2.17, where the contents of slot x of input i are copied into slot y of output link o. On completion of the call, the reserved path is released for use by other calls, and the billing functions are executed. In addition to these functions associated with a call, the switch must also perform background functions for executing the routing protocols and algorithms as well as management and maintenance of the switch.

It is possible that in a circuit switch the input and output links are free but the switch cannot set up a path between them *through* the switch. This can happen

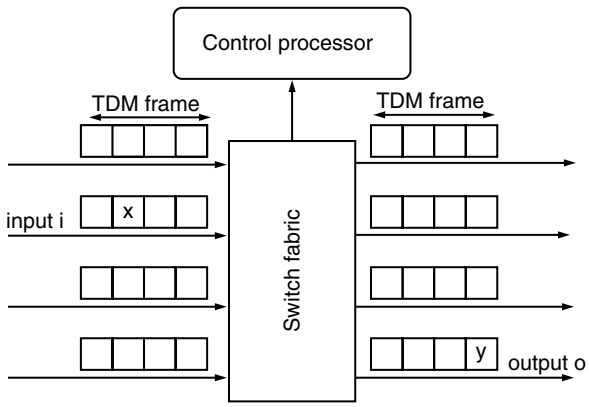

Figure 2.17 The operation of a circuit switch.

because of the paths through the switch that the other active calls may be taking, a condition called *switch blocking*. Switches in circuit-multiplexed networks should be designed to minimize switch-blocking probabilities and to handle a specified call attempt rate. In Chapter 9 we discuss the issues in such designs in more detail.

2.2.3 Routing

Routing is concerned with establishing end-to-end paths between sources and sinks of information. A route can be thought of as a concatenation of communication links and switches that connects a source and a sink. Both routing and multiplexing are activities in which a network switch, also called a node, is involved. The difference is that multiplexing several flows onto a link is the concern of a single network node, whereas the function of routing necessarily involves multiple network nodes.

It is useful to distinguish between the terms *routing* and *forwarding*. Forwarding refers to looking up the outgoing link on which a data packet is to be transmitted (for packet-multiplexed networks) or the time slot in the outgoing TDM frame to which a time slot in the incoming TDM frame must be mapped (for circuit-multiplexed networks). As discussed in Section 2.2.2, this is done by consulting a routing table, which maintains the necessary information. Forwarding is a *data* or *user plane* activity because it deals with data. In contrast, routing can be viewed as the collection of actions, involving multiple nodes, that has the goal

of building the routing table. In this sense, routing is a *control plane* activity. The data and control planes are discussed further in Section 2.3.2; see also Figure 2.22. Several issues that are intimately connected with the routing problem are discussed next.

Information related to network topology and destination reachability is of crucial importance. Depending on how routing is planned in a network, two cases arise. In *centralized routing*, all the topology information is stored at a central point, where routes are computed. Topology information is collected by a distributed protocol, but the route computation is done at the central point. These routes are then conveyed to the nodes in the network where they are needed. In *distributed routing*, the nodes in a network participate in a distributed protocol that not only collects topology-related information but also computes routes in a distributed manner. The distributed protocol may involve exchange of summarized information, as in, for example, the distance to a set of destinations. It may also involve exchange of local topology information, as in, for example, a node advertising information on the status of its attached links.

Related to the topology database is the issue of *aggregation* of topology information. This is important for the scalability of a routing protocol. When a network becomes large, it is possible for the regular topological updates to become so frequent that significant processing and bandwidth resources are consumed. The solution is to design hierarchies of topology-related information, with only summarized or aggregated information about one level in the hierarchy being available to others.

Although establishing end-to-end paths is a basic requirement, a routing algorithm is expected to do more: The routes assigned to flows must be such that their quality of service (QoS) demands are satisfied and, at the same time, network resources are efficiently utilized. Achieving both objectives simultaneously is a challenging task because they are conflicting: It is easy to satisfy QoS objectives by keeping the network lightly loaded, whereas we can easily achieve high resource utilization by not caring about QoS objectives.

There are two ways routes are found across a network. When *hop-by-hop* routing is chosen, each node is interested in computing only the next hop toward a destination. The full path to the destination is not sought. Thus, the topology database is used merely to decide the best next hop toward the destination. The other alternative is to find the complete path to the destination. Evidently, the topology database must contain sufficiently detailed information for this to be possible. The second alternative is referred to as *source routing* because the first node on the path of the traffic (the source node) can compute the complete route to the destination.

The control plane activities related to routing have several timescales associated with them, as elaborated on in section 2.2.5. To begin with, we have the regular exchange of topology information that is necessary to build the topology database. However, route computation need not necessarily occur at this timescale. If a network is static in the sense that no node or link is going down or coming up, then the topology is static. Furthermore, if the traffic pattern is stationary, then routes need not be recomputed often. But the distributed protocol that disseminates topology information must continue its exchanges at regular and frequent intervals. Route recomputation occurs at slower timescales, driven by significant changes in the network topology, such as link or node failure, or by major changes in the traffic pattern. Route recomputation may (or may not) also be driven by performance objectives. For example, when the average transfer delay on a path begins to exceed some acceptable value, indicating congestion somewhere on the path, a route recomputation may be triggered. We note, however, that even though route recomputation can, in theory, be triggered by the onset of congestion, many practical networks may not do this at all. For example, in the Internet, route computation does not consider congestion levels and traffic patterns directly but instead utilizes *weights* defined by network administrators. It is expected that the weight definitions will take into account projected traffic demands and link capacities.

In general, we would need performance measures to gauge the quality of routing. Good routing should mean that under low offered load, information transfer delays are relatively small. As the load offered to a network increases, eventually a part of the offered load must be rejected if QoS commitments are to be met. This is the point at which *connection blocking* begins. A consequence of good routing is that the onset of blocking is delayed. That is, the network can accommodate substantially more traffic before blocking begins.

Again, we hasten to add that, in practice, some networks may not use the notion of connection blocking at all. Indeed, in the Internet, the IP layer is connectionless, and therefore *at the IP layer*, there is no concept of a connection and hence no question of connection blocking. With increasing load, performance measures (for example, average packet delay) indicate higher levels of congestion, and consequently performance degrades, but the IP layer does nothing about it.

In a circuit-multiplexed network, the route is determined when the call is set up. Subsequently, forwarding simply reduces to switching of the appropriate TDM time slot on the input side to the corresponding slot on the output side. Section 2.2.2 discusses this in greater detail. Similarly, virtual circuit-based packet-multiplexed networks have a connection establishment phase (or call setup phase) prior to data transfer, and, again, routes are determined during this phase.

Every packet associated with the call is tagged with a label at every switch; the label identifies the outgoing port to which the packet should be switched. The labels have local significance only, because they merely indicate an input-port-to-output-port map at a particular switch that was established when the call was set up. In contrast, in a datagram-based packet-multiplexed network, there is no connection establishment phase before data transfer begins. Each packet is routed as a separate entity, without reference to the connection or session to which it belongs. Indeed, in a datagram-based network, the very notion of a connection (at the network layer) does not exist. Therefore, each packet carries the complete source and destination addresses that may possibly be used to find a route for that packet through the network. This mode of operation can be contrasted with that in virtual circuit-based packet-multiplexed networks, where each packet need not carry the complete source and destination addresses; it is sufficient for it to carry the label assigned at the time the connection was established, because this label is used to obtain the route.

The recent development of *reactive* routing has been driven by the advent of wireless ad hoc networking. In the preceding discussion, we have tacitly assumed that routing-related control-path activity is carried on irrespective of the presence of data traffic. For example, topology updates are periodically sent and received by a network node even when it has no data packets to forward. The idea is to have routes available for use whenever needed. This approach is referred to as *proactive* routing. However, in wireless ad hoc networks, such continued background control traffic processing can be expensive in terms of excessive power consumption, leading to reduced battery life. This has motivated the approach of acquiring topology information only when it is needed, that is, only when a node needs to forward a data packet and finds that it has no appropriate route. This approach to routing is called reactive to contrast it with the traditional proactive approach.

Next, we take a brief look at routing in packet-multiplexed and circuit-multiplexed networks. We will see that even though the core issues are common to both types of networks, the mechanisms that have evolved to address the issues are sometimes different. We first consider packet-multiplexed networks, followed by circuit-multiplexed networks.

Both virtual circuit and datagram-based networks usually run specific protocols aimed at acquiring information related to neighbor reachability. For example, the Hello protocol is used in both OSPF (Open Shortest Path First)—a routing protocol in the Internet, a datagram-based packet network—and PNNI (Private Network-to-Network Interface), a routing protocol used in ATM networks, which are virtual circuit-based packet networks.

In "small" virtual circuit networks, source routing is normally used. Thus, the first ATM switch on a connection's path would provide a full path, listing all intermediate switch identifiers, to the connection setup packet. As the network grows larger, this approach may not scale because too many nodes may have to be specified. As mentioned before, hierarchical routing schemes would be required in such cases. In datagram networks, hop-by-hop routing is commonly implemented. However, some routing protocols, such as OSPF, do have enough information to obtain complete paths to a destination rather than only the next hop.

Route selection in virtual circuit networks is often accompanied by resource reservation at the same time. Thus, as the connection setup packet travels down the chosen path, the resources necessary to provide the requested QoS are also reserved. This leads naturally to the *crankback* feature found in virtual circuit networks. Here, a source node has a set of alternative routes available to it, and, to begin with, it tries the first of these. If call setup is not successful because some switch on the path has inadequate resources, then a failure message is sent back to the source node. This causes the source to crank back and try other routes in its set, avoiding those routes in the set that pass through the switch where resources are currently limited. In datagram networks, the crankback feature is not usually found. Traditionally, datagram networks carried best-effort traffic, and there was no particular concern about finding adequate resources on a path. In case of drastic events such as link failure, the routing protocol would detect the change in topology, and new routes, bypassing the failed link, would be computed.

Routing in packet-multiplexed networks has the dual objectives of meeting QoS requirements at both call and packet levels and ensuring efficient resource utilization. This is discussed in greater detail in Section 3.1. Thus, minimizing call-blocking probability is an important goal, but so is ensuring that packet-level QoS measures—for example, packet transfer delay and packet loss probability—are also adequate. It is worth noting that the multiplexing policies implemented at switches have a significant impact on packet-level QoS measures. This indicates again that routing and multiplexing need to be considered jointly if we are to achieve overall QoS objectives. However, as mentioned in Section 2.2, we follow a divide-and-conquer approach and consider multiplexing and routing separately.

In circuit-multiplexed networks, a notable point is that even though the network that carries user-generated information—for example, voice—is circuit-multiplexed, the network used for exchange of control information is packet-multiplexed. This is the case, for example, in the telephone network (see Section 2.3.3), where the common channel signaling (CCS) network carries the control information that needs to be exchanged by the switches.

In packet-multiplexed networks, however, both user-generated information and control information are carried as packets and coexist on the same network.

The network topology in circuit-multiplexed networks is considered to be relatively much more static. Thus, there is little need for an automated process such as the Hello protocol, which keeps checking at frequent intervals whether neighboring nodes are "alive." Normally, network topology is administratively configured, and manual reconfiguration is done as necessary. Circuit-multiplexed networks follow the source-routing philosophy.

The absence of a protocol that generates regular topology update messages indicates that the control traffic in a circuit-multiplexed network flows on a slower timescale. The intervals at which routing tables are updated depend on the routing method chosen. For example, many operational telephone networks use fixed traffic routing, in which the switches are equipped with a set of routes corresponding to each destination; these routes are computed offline during the network design phase and are programmed into the switches. This is discussed in greater detail in Section 3.3. When a call is to be set up, a switch tries the stored routes one by one, and if adequate resources are available on a route, the call succeeds. Dynamic traffic routing allows routing tables to be changed dynamically, either in a planned manner or in response to changes in network state. For example, telephone networks may update routing tables at the beginning of the "busy hour" to cope with the increase in traffic; this is an example of planned dynamic routing.

In circuit-multiplexed networks, the objective of routing is to minimize call-blocking probability and utilize links efficiently. After a call is set up, the in-call QoS is automatically ensured, because an end-to-end path is reserved specifically for this call. Thus, in contrast to packet-multiplexed networks, only call-level QoS objectives are of concern here.

Topics related to routing are discussed in Chapters 13, 14, 15, and 16.

2.2.4 Network Management

When a network is established, the topology, the bandwidths, and the multiplexing, switching, routing, and traffic controls are chosen so as to provide a certain grade of service, however that is defined, to handle a specified engineered load and certain kinds of faults. An operational network is a complex system and must be continuously monitored and measured to ensure that the achieved grade of service is as per the specification and design. Often the load may exceed the engineered load because of excessive failures, delay in capacity upgrading, or any other event that was not accounted for during planning. These situations need to be managed by external means, which depend on aggregate measurements and

slower timescale controls. This is in the domain of *network management*. Note that accounting for every eventuality during the network design stage would lead to a very expensive network design; hence, in practice, only a certain range of operational conditions are designed for, and excursions outside this range are carefully monitored and managed when they occur.

All operational networks define a network management architecture that provides them with a mechanism to collect performance data from the various network elements and to remotely configure and hence control these elements in a centralized manner. Examples of performance data include bit and packet error rates on links, the traffic volume delivered by a link, and the volume dropped by it. Mechanisms are also included to identify the specifications of the network elements and their configuration details. Network management includes the ability to isolate faults and reconfigure the network during failures so as to restore service. Service may be restored by using spare resources and does not imply repair of the fault.

Securing the network against threats is also becoming an important aspect of network management. Protection against unauthorized access is provided by firewalls that allow very specific traffic into or out of a network. Managing the access through these devices is a network management function. *Denial-of-service* attacks make a target network handle large volumes of "irrelevant traffic" and thereby choke it. This in turn prevents legitimate users from gaining access to the network. Security management functions should be able to recognize that such attacks are developing and minimize their effect. Many times it is not possible to recognize a security breach when it is in progress. An important aspect of network security management is to examine network logs and do a post facto analysis to recognize security breaches.

Network management protocols decide the kind of data to be collected by the different *managed* nodes and specify how these data are made available to the manager. These protocols also decide which configuration parameters of the managed devices can be read by the network manager. TCP/IP networks (see Section 2.3.5) use the Simple Network Management Protocol (SNMP) and the Remote Monitoring (RMON) protocol to collect device and network statistics. These protocols also permit alarms to be triggered based on preset thresholds, and elementary control actions to be exercised by the network manager. The ITU has developed a network management architecture called Telecommunication Management Network (TMN) that is useful in managing large, complex, multivendor networks. Most telecommunications vendors conform to the TMN specifications, making it possible for an operator to easily integrate equipment purchased from various vendors into a network with unified network management.

Although network management protocols may be standardized, network management practice is usually a collection of heuristics that are developed from operational experience. Little systematic engineering appears to have been brought to bear on this area, and hence we will not discuss network management further in this book.

2.2.5 Traffic Controls and Timescales

Network traffic can be described across many timescales, from microseconds to days or even years. Thus the control mechanisms for the resource-sharing activities also need to be viewed as operating over various timescales. In fact, these resource-sharing and control mechanisms can also be understood in terms of the timescales over which they act. Rather than consider absolute time we identify the following four relative timescales.

- Packet timescale (packet transmission time; microseconds or milliseconds)
- Session, call, or flow timescale (typically minutes)
- Busy hour or traffic variation timescale (typically hours)
- Provisioning timescale (usually hours to days or weeks)

Let us now discuss the relation between these timescales and the four basic activities that we have outlined.

Of course, the packet timescale applies only to packet-multiplexed networks. Packet timescale controls discriminate between the treatment that the network provides to individual packets. These controls include mechanisms such as priority queueing, adaptive window-based congestion control (as in TCP in the Internet), and random packet dropping (for TCP performance optimization). For example, packets of a real-time service, such as interactive telephony, need to be expedited, and hence these can be given priority when competing for transmission over a congested link. When a link is shared between two sets of flows, then a scheduling policy can be set up that allocates half the link's capacity to each set of flows; this requires packet-by-packet decision making at the point where the two flows are being multiplexed into the link. File transfers in an internet (see Section 2.3.5) are controlled by a set of packet-level mechanisms in the end-to-end TCP protocol. A TCP transmitter works with individual packets, deciding when to transmit packets based on acknowledgments or lack of them. We can control the performance of these mechanisms by deliberately and randomly dropping individual packets from TCP flows.

When a session or call (e.g., a phone call in the phone network, or a video streaming session in a packet network) arrives at a network, several decisions can be made. Should this call be accepted or not? A circuit-multiplexed network would block a call if there are no channels remaining to carry a new call of the arriving class (e.g., a nonemergency call). In a packet network, a call may not be accepted if the network cannot offer it the required QoS. If the call is accepted, how should it be routed, and how much transmission rate will the call use up on that route? The route should be such that it has enough resources to handle the call. Such decisions are made only when flows or calls arrive and hence can be viewed as operating over flow arrival and completion timescales, which are certainly slower than packet timescales.

In telecommunication networks the traffic processes are not stationary but instead vary continually over the hours of the day. Different traffic patterns exist during different times of the day. An obvious example is traffic between organizations, which would be heavy during business hours, whereas the residential networks would carry traffic during other times. With networks becoming international, time zones and time differences also govern the traffic pattern. Most networks are engineered for a *busy hour* during which the traffic is at its peak.

The controls at the packet and flow timescales are typically parameterized to optimize network performance for a specified traffic pattern. If the traffic pattern changes, however, a different set of configurations for these controls may be more efficient. Hence at traffic variation timescales it becomes necessary to adjust the packet- and flow-level controls in order to optimize network performance. For example, the network routing may need to be adjusted to accommodate varying patterns of traffic at different times of the day. Thus it is at these timescales that the network operator would adjust routing policies in order to optimize the network performance. In the context of telephone networks, the routing plan could be different for different times of the day or on different days of the week. This is also called the *traffic engineering* timescale.

Notice that the controls at the packet timescale and session timescale are concerned with the control of the way traffic is multiplexed and switched. Similarly, the flow and traffic engineering timescale controls are applied through routing functions.

Resource provisioning, also known as capacity planning, is an important part of network management. In the hierarchy of control timescales listed earlier, provisioning is the slowest timescale control. It entails increasing the transmission rate on some links or adding new links and switches. If the fastest timescale controls (based on controlled multiplexing and switching, or adjusting the routes)

do not provide the desired service quality to the offered traffic, clearly it is time to add capacity to the network. For example, reprovisioning could be triggered when attempts to reroute traffic or to reconfigure the other traffic controls fail to achieve the desired results, and this happens frequently over several traffic engineering timescales. Actions at the provisioning timescale could be to increase the capacity of some links or to establish new links, which might in turn entail upgrading switching equipment. Modern networks are being built on optical fiber links, with intelligent optical crossconnects. Such an infrastructure can be used to rapidly provision additional bandwidth with little or no manual intervention.

2.3 Current Practice

In Sections 2.1 and 2.2 provide a generic view of the issues involved in developing a communication network. Over the past 100 years, several communication network technologies have been developed and deployed. In this section we provide an overview of the current practice. Our aim here is to provide the "big picture" rather than deal with the myriad details. In keeping with the view presented in Figure 2.2, we present this material in two parts. First we discuss the network infrastructure, where we describe the way communication links are derived from physical communication technologies. Second, we discuss networking architectures, describing the way information transport services are developed on top of the bit-carrier infrastructure. We follow this discussion with examples of the major networking technologies: the circuit-switched phone network, IP packet networks (internets) and ATM packet networks. We also provide a brief overview of X.25 networks, Frame Relay networks, and ISDN.

2.3.1 Network Infrastructure

By *network infrastructure* we basically mean the network of communication links (see Figure 2.2). In modern networks, the interswitch links are typically derived from an optical fiber infrastructure. This is becoming so even within large buildings and campuses, where *local area network* (LAN) switches are increasingly being connected by optical fiber links. What remains is the connection of terminals (sources and sinks) to the switches; such access continues to be over copper media, and increasingly over multiple-access wireless links. In the following discussion we provide an overview of optical fiber backbones and wireless access technologies.

Optical Fiber Backbones

Figure 2.18 depicts how an optical fiber infrastructure is used in a packet network; some of the terminology in this diagram is from the Internet, the currently

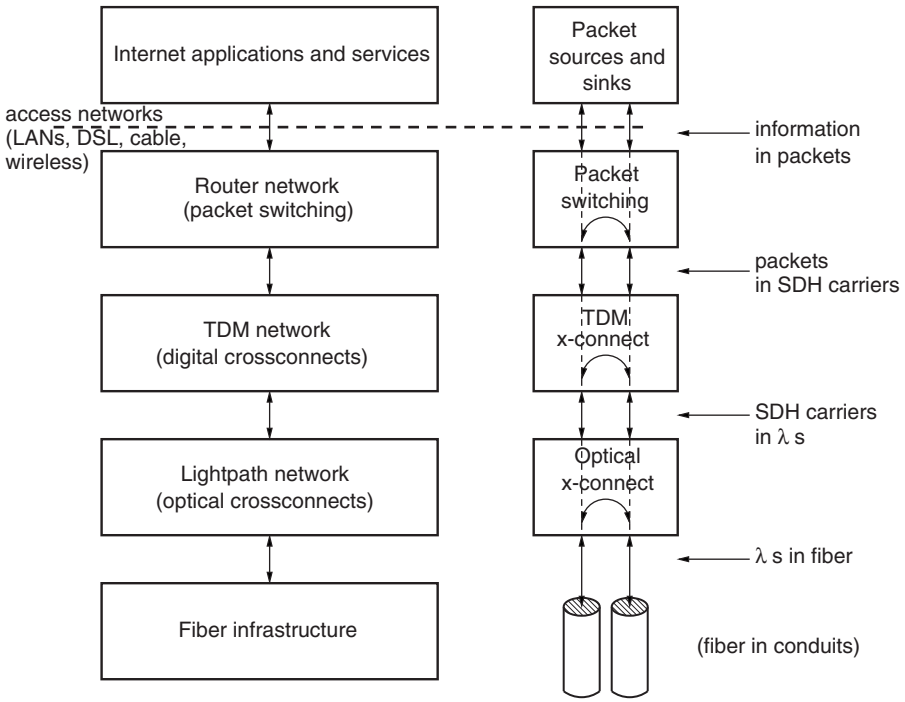

Figure 2.18 From fiber to packets: how links between packet switches are derived from an optical fiber infrastructure. The left side shows the various physical networks that are derived from the fiber infrastructure, and the right side shows the network components that are needed to achieve these layers of networks.

most popular packet network technology, which we describe in more detail in Section 2.3.5.

First, notice the *layering* in the architecture. By layering we mean that one level of the network provides a service to the next upper layer without the upper layer being aware of the functional details of the layer below it. Let us begin with the bottommost layer, which comprises strands of optical fiber sheathed in protective cladding and laid inside conduits. Normally such lengths of fiber terminate in junction boxes in buildings at each end and lie unused until the need arises to provide additional transmission capacity between the end points of the fiber cable. Then the *dark fiber* is *lit up* by terminating each end with connectors,

which are then attached to optical interfaces in transmission equipment. Each fiber can carry several tens of *wavelengths* (or colors), also called λs owing to the symbol often used to denote wavelength. Each wavelength is *modulated* to carry a bit stream; for example, a common modulation scheme is to simply turn the light on and off to signal 1's and 0's. Current technology permits each wavelength to carry a bit rate of about 10 Gbps.

Figure 2.18 shows that the first level of transmission equipment creates a network of *lightpaths* from the *fiber plant*. The simplest case of a lightpath is one in which on every fiber traversed by the path, the same color is used. The electronic device that modulates (or demodulates) the bits into (or from) the lightpath needs to exist only at the ends of the lightpath. In between we require only wavelength switches, or, as they are called in the industry, *optical crossconnects*. An optical crossconnect simply modulates a color on an outgoing fiber according to the modulating light signal it receives on an incoming fiber. If the incoming and outgoing wavelengths are different, then the crossconnect also needs to do *wavelength conversion*. The technology of carrying information over several wavelengths in a fiber, and then optically switching this information between wavelengths in different fibers, is called *wavelength division multiplexing* (WDM).

At the end points of the lightpaths, the bit stream carried by the light must be extracted. This is shown as the next layer in Figure 2.18. The bit stream actually has a periodic, hierarchical format corresponding to a digital transmission standard, such as SDH (Synchronous Digital Hierarchy) or SONET (Synchronous Optical Network). Thus, a bit stream with a 9.95328 Gbps information rate can actually be viewed as comprising 64 separate TDM bit streams, each with an information rate of 155.52 Mbps. A *digital crossconnect* can interconnect these 155.52 Mbps TDM *subcarriers* between the various lightpaths terminating at this crossconnect. Notice that by doing this we have obtained end-to-end digital links at, say, 155.52 Mbps. An *add-drop multiplexer* is used to extract a subcarrier that needs to be terminated at a location, or to insert a subcarrier into a TDM stream.

These digital links, in turn, carry multiplexed information as circuits or packets. In Figure 2.18 we assume the example of a packet network, such as an internet. At places where packet switches are deployed, a digital crossconnect extracts a digital subcarrier, which is then connected to a port of the packet switch. The packet switch then extracts packets from the digital bit stream. Some of these packets are passed on to terminal equipment (shown as packet sources and sinks in Figure 2.18—e.g., computers or packet phones) at the location of the packet switch; as shown in the figure, the terminal equipment is connected to the *backbone* network, by an access network, which could be based on wired technologies

(such as Ethernet, television cable, or digital subscriber loops (DSL)) or any of the several emerging wireless access technologies. At the packet switch, packets not meant for local terminals are switched into output ports of the switch. These are then inserted into a digital subcarrier by the output interface of the packet switch. This subcarrier is then multiplexed into a higher-speed digital carrier, which is then modulated into a lightpath, which is in turn multiplexed into an optical fiber.

In practice, of course, the same optical transmission facilities are used to carry both circuit-multiplexed networks and packet-multiplexed networks. If it is a circuit-switched network that is being built on top of the optical fiber infrastructure, then the digital carriers terminate into circuit switches. Essentially, one can think of a circuit switch as a digital crossconnect that rapidly sets up and tears down end-to-end digital bit-pipes on demand from end users.

With this discussion in mind, it is clear that there are engineering issues at all levels in the architecture shown in Figure 2.18—from the laying of the fiber cables to the design of the lightpath and digital carrier network to the placement of the routers and the choice of link speeds to interconnect them. The physical laying of fiber is, of course, a long-term decision that one cannot revisit or modify frequently. On the other hand, current technology permits the rapid reconfiguration of optical crossconnects and digital crossconnects. Yet current practice does not actually exploit this possibility except over fairly slow timescales: typically days or weeks, at least. The main reason is that standard network management procedures that can effectively exploit this flexibility are not yet available. We can expect that such procedures will be developed as the demand for fast timescale bandwidth reprovisioning grows. Hence, in this book we confine ourselves to the design problem at the highest of the four levels of the infrastructure shown in Figure 2.18. At this level we see only switches (circuit or packet), the links between them, access networks, and terminals. If a circuit-multiplexed network and a packet-multiplexed network are carried on the same transmission infrastructure, then at this level they become logically separate networks. Thus, Figure 2.19 shows a packet network with the details of the transmission facilities abstracted away. This is an example of a network operated by an Internet service provider (ISP). The backbone comprises packet switches interconnected by digital links (e.g., 34 Mbps or 155 Mbps). This network attaches to other networks by *peering links*. Users access the network over high-speed LANs or over lower-speed serial links (e.g., 2 Mbps or 34 Mbps).

As we have stated before, in this book we do not discuss the problems of physical network design—fiber routing, node placement, topology design, and link sizing. We assume that a network infrastructure such as that shown in Figure 2.19 has been realized, and we concern ourselves with the problems of sharing the resources of such a network infrastructure.

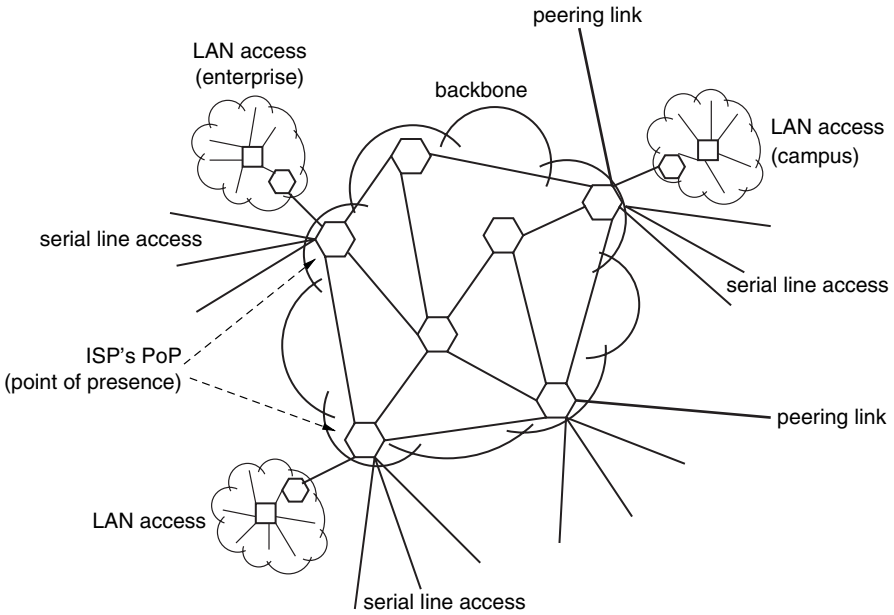

Figure 2.19 A view of a packet network with the optical fiber infrastructure abstracted away. An Internet service provider (ISP) connects to its customers at points of presence (PoPs).

Mobile Wireless Access Networks

So where does wireless transmission fit into the overall picture shown in Figure 2.18? There are two places. Until a few years ago, long distance digital transmission links were carried mainly on microwave wireless facilities. Owing to decreasing costs of optical transmission technology and to the much higher bandwidths available, digital microwave terrestrial networks are rapidly being replaced by optical transmission facilities. On the other hand, over the past 20 years, mobile wireless communication technology has made major advances, spurred by new modulation, coding, and signal processing techniques and the availability of high-performance digital signal processing chips. With hundreds of millions of cellular telephony subscribers, this technology is already playing a major role in the access to circuit-switched networks. In the architecture of new-generation *cellular wireless networks*, the availability of bandwidth for access to packet-switched networks is a major consideration. Furthermore, *wireless local*

area networks are now providing transmission speeds in the tens of megabits per second. Given the major attraction of mobility, wireless networks can be expected to become the access networks of choice in office buildings and homes. Here we provide a brief overview of the communication infrastructure in cellular wireless networks; we discuss wireless local area networks in Chapter 8.

Figure 2.20 shows the components of a typical cellular wireless network infrastructure. The figure is modeled after the popular GSM (Global System for Mobile communications) system for mobile telephony, which is deployed in many regions throughout the world. The wireless links are only between the *mobile stations*, or MSs (shown as cellular phone handsets in the figure), and the *base transceiver stations* (BTSs). An MS can be in the vicinity of several BTSs, but at any point in time, an active MS is associated with one BTS: the one with which it is determined to have the highest probability of reliable communication. The region around a BTS in which MSs would most probably associate themselves with this BTS is called a *cell*. Thus the deployment of several BTSs in a geographical region essentially partitions the region into cells. If during a call, an MS moves between cells, a connection *handover* must be done between two BTSs. To facilitate handovers, the BTSs are laid out so that the cells overlap at their peripheries.

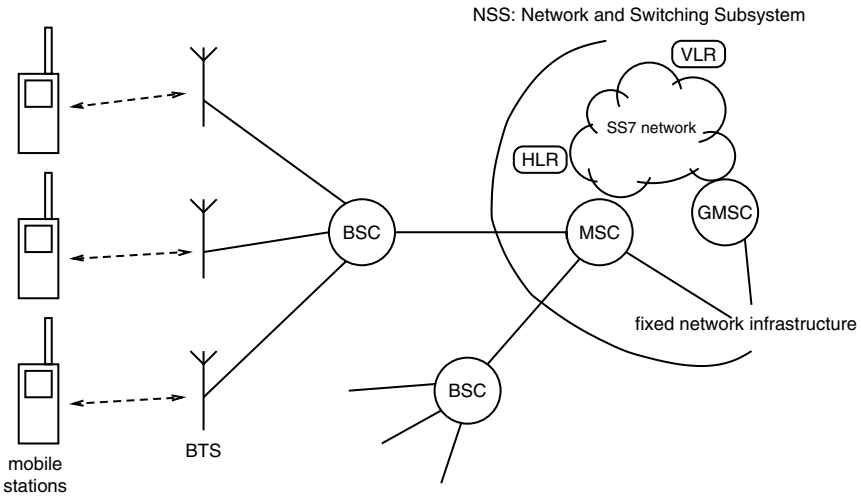

Figure 2.20 A typical cellular wireless network infrastructure.

Several BTSs are linked to *base station controllers* (BSCs) by wired links. Together, the BTSs and the associated BSC are called a BSS (*base station subsystem*). The BTSs provide the fixed ends of the radio links to the MSs; it is the BSC that has the intelligence to participate in the signaling involved in connection handovers. In turn, the BSCs are connected to the *mobile switching center* (MSC), which connects to the fixed network infrastructure.

A cellular operator leases spectrum (say, in the 900 MHz band) from the spectrum management authorities in the region. The spectrum is assigned in two equal-size segments: one to be used for communication from the MSs to the BSSs (this is called the *uplink* direction), and the other to be used from the BSSs to the MSs (the *downlink* direction). This spectrum is then partitioned, and each cell is assigned one of the parts. There is a fixed association between uplink and downlink frequencies, so that if an uplink frequency is assigned to a BTS, then the corresponding downlink frequency is also assigned to the same BTS; hence we can think in terms of the assignment of, say, only the downlink frequencies. The assignment of parts of the spectrum to cells must be done so that the same parts of the spectrum are reused in cells far enough apart so that they do not interfere with each other. Obviously, the less we need to partition the spectrum and the more we can reuse the frequencies over cells, the more communication traffic can be handled by the cellular network.

A digital modulation scheme is used to transmit digital data over the wireless spectrum allocated to a BTS. The approach taken in GSM is to first partition the spectrum into several *carriers*. Each carrier (there is a pairing of uplink and downlink carriers) is digitally modulated. On each of these digitally modulated carriers, several subchannels are defined; these are used for carrying user traffic and for signaling. Note that after an MS is connected to the network, it not only must send and receive data bits but also must exchange connection control messages to, for example, facilitate handovers.

Because the resources (i.e., the spectrum) of a cellular wireless network are limited, an MS cannot have permanent access to the network but instead must make a request for connection. Thus, because an MS is not always connected to the network, there are two problems that must be addressed.

1. Between the time that an MS last accessed the network and the time that it next needs to access, the MS may have moved; hence, it is first necessary to locate the MS and associate it with one of the cells of the network.

2. Because the MS initially does not have any access bandwidth assigned to it, some mechanism is needed for it to initiate a call or to respond to an incoming call.

Call setup and location management are the major activities that need to be overlaid on the basic cellular wireless infrastructure to address the first problem. In Figure 2.20 we show the additional components that are needed. Together, these are called the *network and switching subsystem* (NSS) and comprise the MSC, the GMSC (*gateway MSC*), the HLR (*home location register*), the VLR (*visitor location register*), and the signaling network (standardized as Signaling System 7 by the ITU (see also Section 2.3.3). The SS7 signaling network already exists where there is a modern circuit-multiplexed phone network. As their names suggest, the HLR carries the registration of an MS at its home location, and a VLR in an area enters the picture when the MS is *roaming* in that area. Each operator has a GMSC at which all calls to MSs that are handled by the operator must first arrive. The GMSC, HLR, and VLR exchange signaling messages over the SS7 network, and together they help in setting up a call to a roaming user.

Let us now turn to the second of the two problems just enumerated. In the GSM system there are several permanent channels defined in each cell. Whenever an MS enters a cell it "locks into" these channels. One of these channels is called the *paging and access grant channel* (PAGCH). If a call arrives for an MS and it is determined that the MS may be in a cell or in a group of cells, then the MS is "paged" in all these cells. Another such common channel is basically a slotted Aloha *random access channel* (RACH) (see Chapter 8) and is shared by all the MSs in the cell. When an MS must respond to an incoming call (i.e., it is paged on the PAGCH) or must initiate a call, it contends on the RACH in the cell and conveys a short message to the network. Subsequently, the network allocates a channel to the MS, and call setup signaling starts.

Several other standards exist for using the wireless spectrum to create digital carriers between the MSs and BTSs. Note that GSM divides the spectrum into frequency bands, digitally modulates each band, and then shares these digital bit-pipes in time; thus it creates bit carriers from the spectrum by using an FDM/TDM (frequency-division-multiplexing/time-division-multiplexing) approach. A major contender to GSM is CDMA (*code division multiple access*) technology, which does not partition the spectrum but separates the users by making their signals appear to be orthogonal to each other. We do not provide details of this approach in this book. For the purpose of this chapter, suffice it to observe that the spectrum-sharing technology, along with the call setup signaling and mobility management, provides a basic access infrastructure to mobile terminals. It is important to note that while an MS is connected to the network the mobility management is transparent to the communication, and the MS can be viewed simply as having a link into the network for the duration of the connection (see the dashed line labeled "access networks" in Figure 2.18). The link would

be error-prone and subject to fading, and it could temporarily break during handovers.

Cellular wireless networks facilitate wide area mobile access to circuit-multiplexed phone networks and packet-multiplexed data networks. Because they must support highly mobile users and long distances (hundreds of meters to kilometers) between the mobile and the fixed infrastructure, the bit rates supported are quite small (Kbps in the second-generation cellular networks). On the other hand, wireless local area networks provide mobile access within buildings. These networks provide a communication range of a few tens of meters, tens of Mbps of bit rate, and mobility within the building. Unlike cellular networks, wireless LANs use some form of random access for sharing the wireless spectrum between the users. We discuss these topics at some length in Chapter 8. To provide continuous network access as a person moves from indoors to outdoors and back, access devices are being developed that can use the wireless LAN access in a building and cellular access outdoors, while seamlessly handing over between these as necessary.

2.3.2 Networking Architectures

We have seen in Sections 2.1 and 2.2 that several mechanisms and functions must be built on top of the basic bit-carrier infrastructure (described in Section 2.3.1) to provide information transport services. In Figure 2.2 we further partition these mechanisms and functions into two parts: the networking functions and information services. The former refers to the mechanisms for sharing the resources of the communication infrastructure among the various contending information flows, and the latter refers to certain common services that facilitate the utilization of the network. Many of these mechanisms and functions require the implementation of distributed procedures, which involve two or more remote entities executing their respective parts of the procedure to achieve some overall objective. In networking terminology, such a procedure is called a *protocol*. Because protocols are implemented in a distributed fashion, it is important that they be designed so that correct results are obtained in spite of asynchronous execution and delayed and unreliable information exchange.

Recall that a circuit-multiplexed network can basically be viewed as an extension of the TDM bit-carrier infrastructure (see Section 2.3.1). The circuit switches that interconnect individual user demands are crossconnects that operate rapidly to setup and tear down end-to-end circuits. Thus at the multiplexing and switching level there are no further architectural issues. However, as you have seen, associated with these circuit-multiplexed networks are signaling networks

that are themselves packet-switched. The discussion in this section also applies to such packet-multiplexed signaling networks.

There are many protocols involved in the implementation of a complete communication network. When commercial networks were first developed, there was no common architecture that was followed. Architectures were vendor-specific, and network systems from different vendors would not interoperate. To systematize the implementation of network protocols, a seven-layer architecture, known as the *Open Systems Interconnection* (OSI) model, was developed; it is shown in Figure 2.21. Each function required in a network is mapped into one of the layers. A layer is viewed as providing *services* to the layers above, and as obtaining services from the layers below. Each layer can carry out its function satisfactorily only if it is provided certain requisite services from the layers below. For example, a distributed algorithm may need reliable and sequential delivery of interentity messages in order to operate correctly. Such a distributed algorithm must therefore obtain such a service from the layers below it.

In addition to codifying the places of the various functions in the overall network architecture, an important objective of the OSI model is to facilitate *open systems* as opposed to proprietary systems. The idea is that if the layer interfaces are well defined, then different vendors can supply various layers of the architecture. If each vendor correctly implements the layer it supplies, the whole system should work correctly.

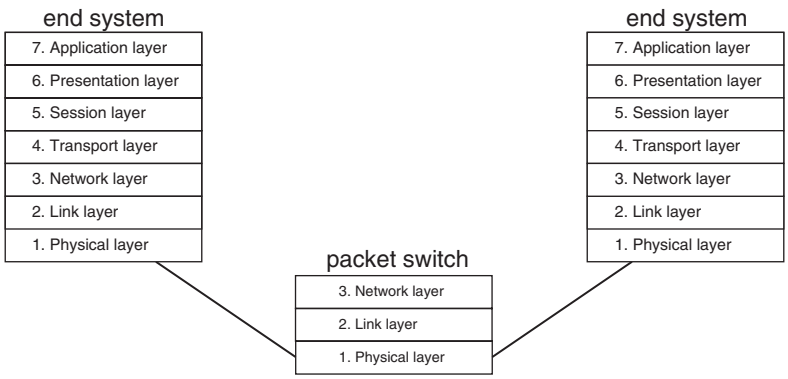

Figure 2.21 The Open Systems Interconnection (OSI) model.

The OSI model is an architecture for packet communications. The peer layers exchange packets in carrying out their distributed procedures. Some of these packets may carry user data, whereas others may carry messages only between the peer entities; these messages (e.g., acknowledgment packets) are used in carrying out the procedures. Peer layers exchange packets by utilizing the services of the layer below. Conceptually, the packets generated by a layer are queued at the interface between it and the layer below. When it is ready, the layer below takes each packet from the layer above, attaches some bits of its own (for carrying out its own functions in conjunction with its peer), and then queues the packet in turn at the next layer boundary. Similarly, when a packet is received it moves up the layers. Each layer removes the bits attached by its peer and carries out its functions, passing the remaining packet to the layer above.

The Layers of the OSI Model

The function of the *physical layer* (layer 1) is to send and receive bits over the communication infrastructure. All the issues involved in digital modulation, demodulation, and channel coding, and even the details of the physical connectors, are subsumed in this layer. Note that the channel coding in the physical layer may not always be able to eliminate bit errors; hence there will be a residual bit error rate at the physical layer. Furthermore, because the bits are transmitted over electromagnetic media, there is a finite speed at which signals travel, and this introduces propagation delay between the bits entering the physical layer at one end of a bit-pipe and emerging at the physical layer at the other end. Delays may also be introduced due to the modulation and demodulation mechanisms. With reference to Figure 2.2, the physical layer provides the communication links. Because every device, whether it is a switch or a terminal, must send and receive bits, the physical layer exists in all devices.

Given the raw bit-pipe provided by the physical layer, the *link layer* (layer 2) makes for reliable communication between the two ends of the pipe. The link layer offers *frames* to the physical layer for transmission. We achieve error control by attaching to the data in a frame certain redundant bits that are derived from the data bits. At the receiving link layer (at the other end of the communication link), a mathematical check is performed over the received bits in each frame, and it can be determined with a very high probability whether or not an error has occurred. In case of an error the frame can be retransmitted by the sending link layer. This can be attempted a few times, after which the link layer gives up and silently discard the packet(s) it could not send. In networks with highly reliable links (e.g., optical fiber links), this retransmission function of the link

layer may not be implemented; in such cases the link layer receiver would discard a received packet upon error detection, leaving the application to handle the loss of the packet.

When the layers above the link layer offer it a packet to be sent, the link layer performs its framing function and then queues the packet at the boundary between itself and the physical layer. It is here that the issues of multiplexing of several flows into the physical link are handled (see Section 2.2.1). If the physical link is shared by many transmitters and receivers, then the sharing of the distributed medium is also viewed as belonging to the link layer. This is called *medium access control* (MAC). In the OSI model, the so-called MAC layer is positioned between the link layer and the physical layer. The link layer also exists in every device in the network, whether it is a switch or a terminal.

The *network layer* (layer 3) ties together the links (point-to-point or multipoint) into a network. At this layer, devices in the network are provided with globally unique *network addresses*. The primary function of the network layer is that of routing packets between these network addresses. Figure 2.21 shows that in a packet switch, only the bottom three layers need to be implemented; these are sufficient for the switch to carry out its function of packet forwarding (but see Section 2.3.5 to see what actually goes into an Internet packet switch). Between the packet switches in a network, a distributed routing protocol is implemented in the network layer to learn good routes. The outcome of this route computation is that a routing table is obtained at each switch; this table determines how packets that need to go out of the switch should be forwarded. When a packet arrives into the network layer from the link layer and needs to be forwarded, the table is looked up and then the packet is again given back to the link layer, where it is queued at the appropriate output port. The network layer also exists in terminals, because they need to know how to forward packets to other terminals on their local network (e.g., a switched Ethernet) or to a router on the network if the packet needs to leave the local network.

The transport layer (layer 4) does not exist in packet switches in the network but instead is an end-to-end layer that resides in the end terminals. Important functions of protocols at this layer are to associate communicating entities in the end terminals and to provide each such instance of communication with additional services. These services bring up the level of services provided by the network layer to a level suitable to be utilized by the communicating entities. If several logical end points in a terminal with *one network address* need to communicate separately with other such end points in other terminals, the transport layer takes care of this *logical* multiplexing and demultiplexing of data from and to the communicating entities. This should be distinguished from the physical multiplexing of several

flows into a bit carrier, a function that in this architecture is carried out by the link layer.

The following are two examples of how a transport layer helps to adapt the service provided by the network layer to the needs of an application. The packets of a packet telephony application are carried asynchronously by the packet network. At the receiver these packets are "played out" according to their original timing relationships, and missing packets are interpolated. Timing and sequence information that can be used by the receiver to carry out such playout can be provided by additional protocol fields added by the transport layer. A file transfer application requires reliable, sequential transfer of its data packets even though the network layer may not provide such service; the transport layer can also provide this service. In the Internet, the procedures of dynamic bandwidth sharing between competing file transfers are also implemented in the transport layer.

The session layer (layer 5) provides services for dialog management between users. For example, if a long file transfer is interrupted, the task of carrying on from that point is a function of this layer.

Functions such as file format conversion, data encryption, and data compression are possible services at the *presentation layer* (layer 6).

The application layer (layer 7) provides commonly used services such as file transfer, e-mail handling interfaces, terminal emulation, and directory services.

Note, that although the bottom four layers are clearly distinguishable as separate entities, common practice typically embeds the functions of the upper three layers into applications themselves. Thus, session recovery and data compression may be embedded within a file transfer application rather than being implemented as separate common services. With reference to Figure 2.2, we can also discern that networking, which is the focus of this book, corresponds to the functions in layers 2, 3, and 4 of the OSI model, whereas the information services layer corresponds to the top three layers.

In addition to transporting user data, communication networks carry *control* and *management* messages. The control functions in networks include connection control, or *signaling*, and adaptive routing. Signaling procedures are used to set up associations between entities that want to use the network to communicate (e.g., packet voice call setup) and to control these associations during communication (e.g., putting an ongoing call on hold). Control also includes routing protocols that communicate between network elements to discover good routes in the network and to relearn new routes in case of network element failures. Network management is concerned with monitoring the flow of traffic and the status of the facilities in the network, and also with actions that may need to be

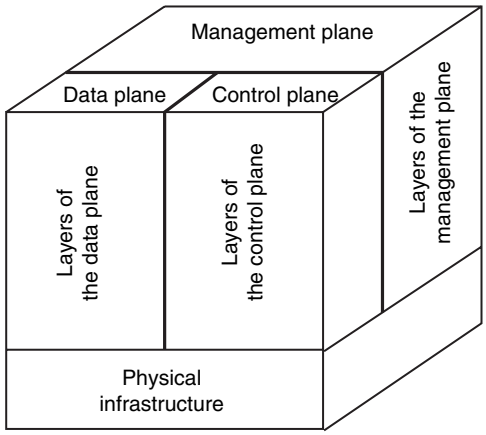

Figure 2.22 Network functions can be viewed in terms of three planes: data, control, and management. They share the same physical infrastructure for carrying out their functions. This model has been adapted from ITU standards.

taken care of—for example, failures and overloads. It is therefore customary to think of networks as having three *planes:* (1) the user or data plane, (2) the control plane, and (3) the management plane (see Figure 2.22). Each of these planes can, in principle, have its own protocols at each of the OSI layers, but they planes would share the physical communication infrastructure. In the Internet, for example, these three planes share the protocols up to the network layer.

Cross-Layer Interactions

As we have mentioned, what we call the networking functions in Section 2.1 correspond to the link/MAC, network, and transport layers of the OSI model. Two important features of this layered model for communication are that each communication function is provided at only one layer, and that layer boundaries are strict, with adjacent layers communicating over well-defined interfaces. Although the seven-layer model has been extremely useful in the conceptualization and design of communication network protocols, the technology has reached a point where the strict layering of functions is routinely violated. A simple example is the use (in the Internet context) of TCP port numbers to identify flows that require a certain QoS, and then the use of this information to manage queues

at lower layers in routers and Ethernet switches (which are, respectively, layer 3 and layer 2 packet-switching devices). Functions are often duplicated: Important examples arise in *IP over ATM*, in which case the ATM layer is often considered to be akin to a link layer. In this case, there is a routing protocol at the ATM level, and another one at the IP level; if the ATM connections use the *Available Bit Rate* (ABR) service (see Section 2.3.6), then there is an adaptive window congestion control at the TCP level and a rate-based congestion control at the ATM level (see Chapter 7). In the latter context, there are research proposals to feed rate information from the ABR control into the adaptive windowing protocol at the TCP level. Of course, all this is happening because of the evolutionary nature of communication network technology and an attempt to create new services from a combination of new technology and an embedded base of old protocols. In view of this we have chosen to deal with the core issues in networking. Wherever it proves useful, we attempt to relate our discussion to the way current technology has addressed the issues.

The interactions between layers take place because of the evolutionary nature of network designs and the need to interwork existing (legacy) systems with new technologies. The traditional layered approach to network design simplifies the conceptualization, design, and implementation of networks. However, recently it has begun to be realized that there are compelling reasons to retain the coupling between the various layers in the process of network design. We can recognize an instance of this in the first example in Chapter 1. We saw that if the voice coder could adapt its coding rate to the occupancy of the link buffer, then there could be a substantial gain in network efficiency. Note that a buffer is a link-level entity, whereas voice coding is an application-level activity. Thus we have an example of beneficial cross-layer interaction.

This issue of possibly beneficial cross-layer interactions is particularly important in mobile communication networks built over wireless communication links. The links vary with time because of multipath fading. These links are shared by a number of traffic flows having various QoS requirements. In addition the end points are mobile and hence resource-limited, a significant limitation being the batteries; mobility also constrains the bit rate that can be supported on the wireless link. In such situations, it becomes very important to make the access and utilization of the network highly adaptive to the channel and traffic conditions, and hence it necessarily leads one to seek approaches that attempt to jointly solve problems at several levels in the layered model. In fact, the channel condition, battery utilization, multiple access, multiplexing, and routing may all need to be interdependent for efficient operation that meets QoS objectives. See Section 8.4 for a more detailed discussion of this topic.

2.3.3 Telephone and ISDN Networks

Until a few years ago, the *public switched telephone network* (PSTN) was used primarily for carrying telephone-quality voice calls but recently has been used for carrying all kinds of other traffic such as data and compressed video. The links in a telephone network are logically partitioned into channels, each of which can carry a voice call; in telephony jargon these channels are called *trunks*, and the group of channels on the link is called a *trunk group*. Each traffic flow on a link is assigned one or more trunks for the entire duration of the flow. Thus the PSTN links are circuit-multiplexed. The switches in the PSTN (also called *telephone exchanges*) establish paths between the channels on the links to which they are attached. Every *call* (an instance of traffic flow) goes through a call setup phase during which a path is chosen for the call. In the telephone network, the predominant calls are requests for "plain old telephone service" (POTS). The exchanges determine a route for each call and the availability of free channels on the links along the route. If possible, channels are allocated on the links along the chosen path, and the intermediate switches are set up to interconnect these channels. If the required number of channels cannot be allocated along a route, alternative routes may be attempted, and if these also fail, the call is blocked. After the circuit for a call is set up, the switches are transparent to the traffic flow of the call. When the end points of the call (the source/sink pair) signal the end of the call, the resources reserved for the call (i.e., channels and switch paths) are torn down.

The telephone network provides exactly one service to any flow that it carries; it can establish bidirectional circuits that can carry high-quality voice signals that have fixed end-to-end data transport delay (the sum of the propagation delay and the switching delay). The sources/sinks need to adapt their needs to the constraints imposed by this service. The QoS offered by a telephone network is specified in terms of the probability that a circuit cannot be allocated to a call (*the call-blocking probability*) and the time it takes to set up the call (*the call setup delay*). The call-blocking probability depends on the network topology and the number of channels on each link. With most of the telephone network being digital and computer-controlled, the call setup delay depends on the computing power of the call-processing computers at the switches, and on the interswitch signaling.

Routing and traffic control in the telephone network facilitate achieving an acceptable call-blocking probability and call setup delay, while using the network resources efficiently. When all the channels on each link on a path are busy, in a richly connected network calls can be *alternate-routed*. If routes or switches are congested, traffic controls are used to reduce new call attempts and to reroute call attempts that have already made progress in call setup.

The telephone network consists of a *local network* serving a *calling area* and the *long distance network*. The phone at the customer premises connects to a *local exchange* (LE), also called *end office* or *central office*, through a local loop. The local loop may also be provided by a cellular telephony infrastructure. In this case the mobile switching center (MSC; see Section 2.3.1) has a function similar to that of the local exchange. In the wireline network, data transmission on the local loop typically is in analog form, and every other part of the network is predominantly digital, with each channel (or trunk) on a link (trunk group) having a fixed capacity of 64 Kbps. The end office location is chosen to minimize the wiring costs to connect the phones distributed in the area served by it. A local network may be served by many end offices. The exchanges in a calling area are interconnected through a *tandem exchange*, and if there is significant traffic between two exchanges, they may be directly connected by *high-usage* (HU) trunks. The use of HU trunks between two exchanges in a calling area depends on the traffic between them and the distances. A call between two LEs is first offered to the HU trunks, and if all the circuits on that are busy, it is offered to the *final route* through the tandem exchange.

An important factor in the dimensioning of the links is its efficiency. The number of circuits on a link depends on the traffic volume expected to flow on that link and the allowable blocking probability. The time average occupancy of a link can be defined to be its efficiency. It can be shown that, for a given call-blocking probability objective, higher-capacity links handling larger volumes are more efficient than smaller-capacity links.

The design of the long distance network is similar to that of the local network except that there are more levels in the hierarchy, and trunk efficiency is a major design issue. The traffic volume between geographically distant LEs is typically small, and direct trunks may be inefficient. Efficiency becomes important with increasing distance because a more expensive resource must be used more efficiently. Thus a switching hierarchy is constructed for the telephone network. However, if there is significant traffic between any two exchanges, they can be connected directly by high-usage trunks, much as in the local network. The telephone network switching hierarchy is shown in Figure 2.23. A number of LEs are served by a primary center, or class 4, exchange, and the hierarchy is built up to quaternary centers, or class 1 exchanges. The exchanges at the top of the hierarchy are connected in a mesh much as in Figure 2.13a. Every exchange, except the class 1 exchanges, *homes* into a higher-class exchange. The first-choice route for a call between phones connected to two different exchanges is through the high-usage trunk toward the destination exchange. If such a path is not available, then the call is attempted through the next level exchange in the hierarchy. Obviously, the

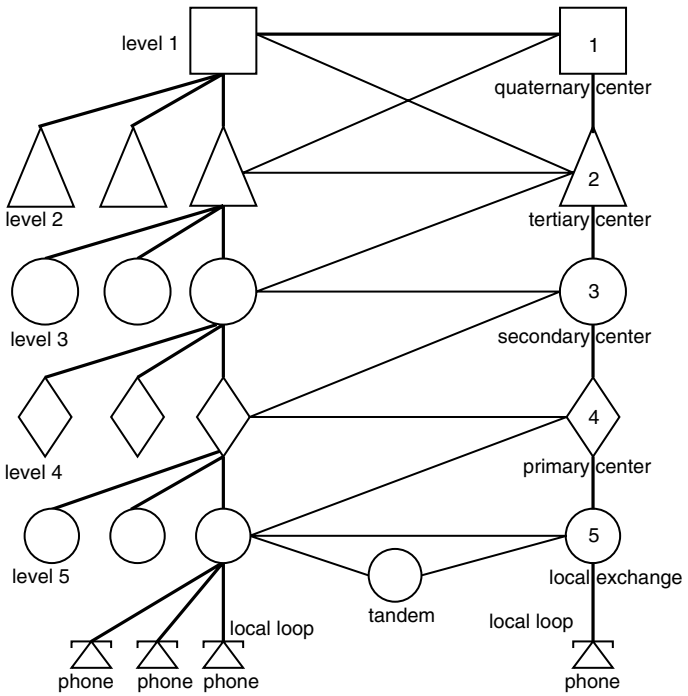

Figure 2.23 Hierarchical arrangement of the telephone network exchanges. The thick lines are links (trunk groups) to be used on the final routes. The notation used in the North American network is shown on the left side, and the ITU-T nomenclature on the right.

highest-level exchange that might be involved is the lowest level that is common to the two phones. Note that a route is always available through the hierarchy, and this route is considered the final route for the call. The final route is shown using thick lines in Figure 2.23. In recent times, nonhierarchical routing schemes have also been devised.

An important aspect of circuit-multiplexed networks such as the telephone networks is call setup. The local exchange communicates status information with the terminal device (such as a phone). All the exchanges communicate control and status information, collectively called *signaling*, among themselves to facilitate path setup and maintenance for the duration of the call. In the early networks, this was accomplished through the exchange of tones of different frequencies on the

same physical channel as that used to carry the voice. Very little information can be reliably exchanged in a reasonable time in this manner to achieve an acceptable call setup delay. This limits the services that can be offered both to the network and to the user. To provide advanced *intelligent network* (IN) features and also to enable the use of more efficient resource (link) utilization capabilities, a fairly sophisticated signaling technology called common channel signaling (CCS) has been developed. CCS uses a logically separate channel for signaling between nodes. The latest version is SS7, and the signaling messages are exchanged over a logically separate network called the signaling network. This logical separation is shown in Figure 2.24. The signaling network corresponds to the control plane of the telephone network and controls the resources in the PSTN.

The network elements of the SS7 network are shown in Figure 2.25. There are three types of nodes in the SS7 network. The *signal switching points* (SSPs) are exchanges that have SS7 capabilities and can hence receive or initiate signaling messages. The *signal transfer points* (STPs) are packet switches in the SS7 network that route the signaling messages, and the *signal control points* (SCPs) are databases that contain the status information that can be used to process a call setup request or provide other services. Normally, each SSP and SCP is connected to more than one STP to achieve signaling reliability. This is not shown in Figure 2.25.

The following is an example call setup procedure between phones connected to exchanges, say *A* and *B*. For the following we assume that the exchanges are directly connected.

1. After the digits are dialed by the calling phone, exchange *A* recognizes that the target exchange is *B*, decides which trunk group to use for the call, and sends an *initial address message* (IAM) to exchange *B* with that information

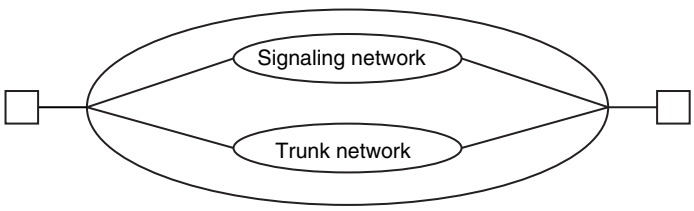

Figure 2.24 The trunk network and the signaling network are logically separate networks. Nodes (exchanges) exchange status and control information over the SS signaling network to allocate and control the resources of the trunk network.

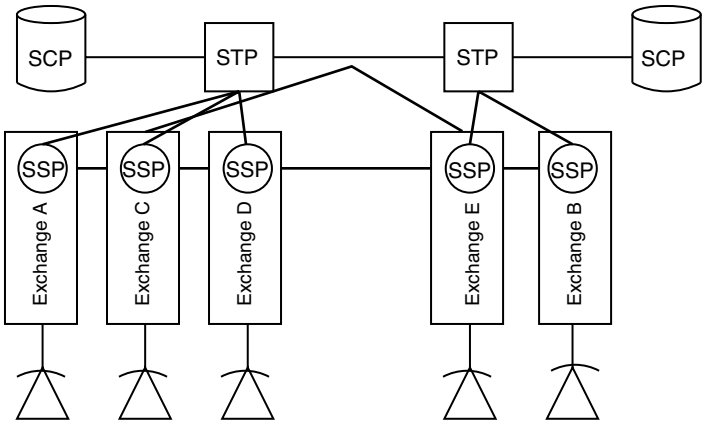

Figure 2.25 The SS7 network. The thick lines are the trunks, and the thin lines are the logical paths for the signaling messages.

over the SS7 network. The IAM passes through one or more STPs to reach the SSP in exchange *B*.

2. On receiving the IAM from *A*, exchange *B* recognizes that the called phone is connected to itself and acknowledges the message with an *address complete message* (ACM) to *A*. Again, STPs are involved in this transfer. Simultaneously, exchange *B* sends a ringing tone over the trunk that *A* has decided to use for the call.
3. On receiving the ACM, exchange *A* connects the local loop of the calling phone to the trunk that it has reserved for the call, and the ringing tone is now available to the calling phone.
4. When the called phone is picked up, *B* sends an *answer message* (ANM) to *A* to indicate that the call is successful.

Call release is achieved in a similar manner.

The SS7 network is built as a packet-switched network. The SSPs are the sources of the signaling packets, and the other SSPs or SCPs can be the destination. The STPs essentially act as routing devices. In fact the protocols used to exchange the signaling messages have a design that follows the layered approach of the seven-layer OSI model discussed in Section 2.3.2.

The SS7 network enables exchange of detailed control and status information. Adding some computation power to this has enabled many new services such as the toll-free service. In this service the toll-free numbers are mapped to different real telephone numbers for different calling areas. The SSP obtains the map by querying the databases at an SCP.

The SS7 network has also been used to provide entirely new access methods. For example, from our discussion of the cellular system in Section 2.3.1, recall the significant amount of signaling that is necessary to provide roaming services to a cellular network user. Also recall that much of this signaling is carried over the SS7 network. Another service that the SS7 has enabled is the Integrated Services Digital Network (ISDN). We provide a brief description of this service.

As mentioned earlier, in the telephone network, digital links are deployed in the backbone of the network at the higher levels of the hierarchy of Figure 2.23, whereas the local loop remains analog. This means that the access interface—the point where the phone instrument connects to the local exchange—is analog, and the benefits of end-to-end digital communication do not reach the end user. In the 1980s ISDN, a new digital access interface to the PSTN, was developed. This interface permits applications to use end-to-end circuits with bit rates in multiples of 64 Kbps. (In ISDN terminology, a 64-Kbps channel is called a bearer (B) channel.) A 16-Kbps data (D) channel is also available, and it can be used for signaling and data transfer. Typically two types of ISDN services are available for an end user. The *basic rate interface* (BRI) is at 144 Kbps and consists of "2B+D" channels. In North America, the *primary rate interface* (PRI) is at 1.536 Mbps and consists of "23B+1D" where the D channel is 64 Kbps, whereas in Europe the PRI is at 1.984 Mbps and consists of "30B+1D" channels, with the D channel being at 64 Kbps. The H channels provide a way to aggregate B channels, and the following aggregations are defined:

- 6 B channels for a 384 Kbps H0 channel
- 23 B channels for a 1472 Kbps H10 channel
- 24 B channels for a 1536 Kbps H11 channel

This ability to let the users demand ($n \times 64$ Kbps) channels for different n makes ISDN a multiservice network. Another important capability of ISDN is the simultaneous establishment of multiple calls, of possibly different types, across an access interface. For example, a 64 Kbps telephone call and a 384 Kbps video connection can be established simultaneously across the same access interface.

An ISDN-compatible end device is called a TE1, and a non-ISDN device is called a TE2 device. TE2 devices connect to the ISDN network through *terminal adapters* (TAs). The two-wire interface from the local exchange is called the *U interface* and is converted to a four-wire interface (called the *T interface*) by network termination equipment called NT1. In North American networks, the NT1 is a customer premises equipment (CPE) device, whereas in many other countries, NT1 termination may be provided by the service provider. This means that the customer premises will have only the T interface. ISDN devices connecting to the ISDN network over the T interface treat the interface as one end of an end-to-end circuit, and they use a communication stack that includes all the layers above the physical layer. A second type of termination, called NT2, is typically used by ISDN-capable *private branch exchanges* (PBXs) to connect to the public ISDN network. An NT2 has the link layer and network layer functions of the OSI model of Figure 2.21 and connects a TE1 over the S interface.

The link layer protocol used in ISDN is called the Link Access Protocol, type D (LAPD) and is specified by the ITU-T Q.920 and Q.921 standards. The network layer protocol is similar to that used by the X.25 network (discussed next) and is specified by the ITU-T Q.930 and Q.931 standards.

The various terminals, interfaces, and protocols are shown in Figure 2.26. Note that Figure 2.26 is an abstract model, and in practice many of the interfaces will be combined into one device.

2.3.4 X.25 and Frame Relay Networks

Early attempts by the telephone companies to introduce public data networks were based on the X.25 protocol suite. This effort gave rise to *packet-switched public data networks* (PSPDNs) in many countries. In this technology, the network links are packet-multiplexed. The links are interconnected by packet switches. A fixed bidirectional path is chosen for all packets of a call; because the path is fixed, even though bandwidth is not reserved along the path, such a call is called a *virtual circuit*. Thus there is a call setup and tear-down phase for X.25 calls. Even though each terminal point of the network has a unique (and long) address, a path for all packets of a call is preestablished, so the switching of packets of a call is done on the basis of short virtual-circuit *labels* carried by the data packets. Traffic control is exercised by means of an end-to-end *window flow control*. Because X.25 was originally designed for poor-quality links, an elaborate error control protocol, LAPB (Link Access Protocol, type B) was implemented at each hop.

X.25-based PSPDNs have been used to carry computer–computer or terminal–computer traffic. The terminal speeds were usually a few kilobits per

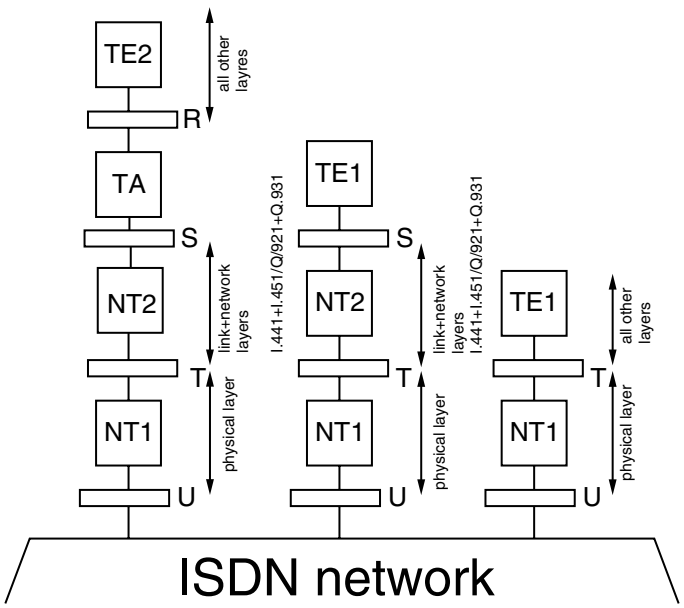

Figure 2.26 Terminals, interfaces, and protocols in an ISDN network.

second (e.g., 19.6 Kbps), and the host attachment speeds were rarely more than 64 Kbps. The links were typically 64 Kbps or 1.5 Mbps. Thus these were low-performance networks, meant primarily for store-and-forward applications that had no QoS requirement from the transport network.

In conjunction with the digitalization of transmission facilities and the development of ISDN, the need was felt for a more efficient packet transport service than X.25. This gave rise to the frame relay protocols and services. Like X.25, Frame Relay is based on virtual-circuit based packet multiplexing. These networks were designed to operate on high-quality digital links with low error rates, and hence hop-by-hop error control was eliminated. The link layer in Frame Relay networks is a derivative of ISDN's layer 2 protocol, namely LAPD. An LAPD receiver checks an incoming frame for errors; if an error is detected the frame is dropped, to be recovered by the end applications. Frames are switched at the link layer rather than at layer 3, as in X.25. Longer frames can be used because the error rates on the links are small. In addition, a Frame Relay network can provide a rate assurance to each connection that is setup through it. Such rate assurances can be

enforced by edge switches, which mark frames that violate their rate assurances, and by network switches, which drop such violating frames during periods of congestion. The most popular application of Frame Relay networks is for the interconnection of corporate local area networks via assured-rate permanent virtual circuits in commercial Frame Relay networks.

2.3.5 The Internet

The Internet refers to the worldwide interconnection of packet networks that all use a suite of protocols that originated in the famous ARPANET project of the 1970s. In this protocol suite, IP (Internet Protocol) is the network layer protocol, and TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are the most commonly used protocols at the transport layer. The common noun "internet" is often used to connote a network that uses the Internet protocol suite. The IP protocol can operate over any link layer (and, by implication, any physical layer) that can transport IP packets. Because it simply requires the implementation of a packet *driver* to carry packets over any bit carrier, an internet can be operated over essentially any bit-carrier infrastructure. The Internet protocol suite also does not define the layers above the transport layer. Thus, for the Internet, Figure 2.21 simplifies to the depiction in Figure 2.27 (we show many more packet switches, and the physical layer is implicit in the links). We will see later that this is a simplified representation; for example, an application may run directly over IP (thus taking care of its transport needs itself).

The most widely deployed version of IP is version 4, which uses 32-bit addresses. The network address of an entity is also called its *IP address*. As in most communication networks, the addresses in the Internet are hierarchically assigned. The address of each device comprises some contiguous high-order bits that identify the subnetwork in which the device resides; this is also called the *network prefix*. The remaining bits identify the device uniquely in the subnetwork. So, for example, all the addresses in a campus may be of the form 10010000.00010000.010xxxxx.xxxxxxxx, where each x can be 0 or 1. In such a case the network prefix is 10010000.00010000.010, and it is of length 19 bits.

Unlike circuit-multiplexed networks or the packet-multiplexed X.25 and ATM networks, internets do not fix a path for the packet flow on a connection. The network simply provides connectivity between end points. Every packet carries the full network address of its destination end point. Each packet switch looks at the arriving packets, consults a routing table (which actually deals with network prefixes), and forwards the packet to an outgoing link that, hopefully, carries it closer to its destination. By rejecting the virtual-circuit approach in favor of

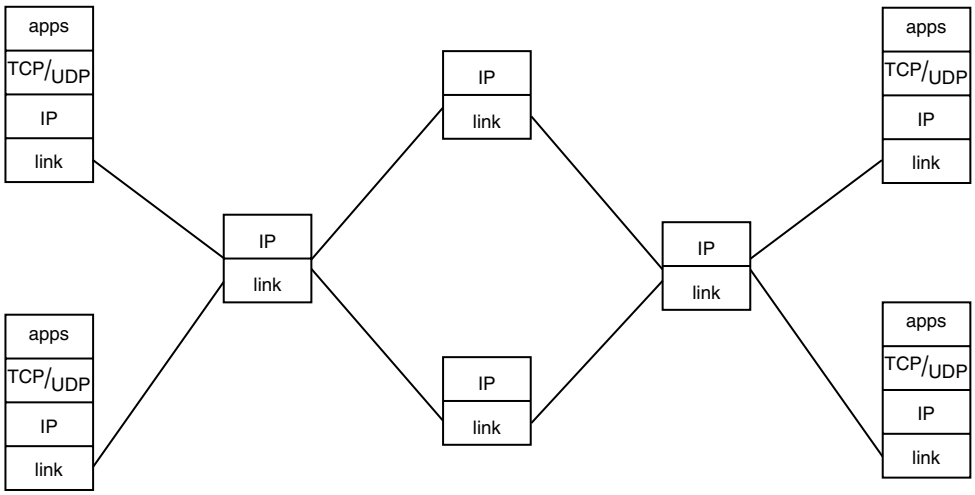

Figure 2.27 The Internet protocol architecture; "apps" means applications. In this simplified depiction, the packet switches are shown to have only IP and the link layer. The end nodes have the transport protocol in addition to IP and the link layer.

per-packet, hop-by-hop, best-effort routing, the Internet gains the advantages of (1) quick delivery of small amounts of data, (2) automatic resilience to link failures, and (3) ease of *multicast* (i.e., the transmission of a packet to multiple destinations by replicating it at appropriate points in the network, rather than by the source sending multiple copies of the packet).

Notice that because IP routes each packet as a separate entity, it is possible for consecutive packets of the same session to follow different routes and then, owing to different delays on the routes, arrive out of order. The IP layer at the destination simply delivers the packets out of order! Measurements showing that there can be significant packet reordering in the Internet have been reported in [28]. In addition, a link layer may discard a packet after unsuccessfully attempting it a few times. A packet may also be discarded at the queue of a physical link because of exhaustion of buffer space or some packet scheduling decision. Hence the packet delivery service that the IP layer provides is *unreliable* and *nonsequential*. This is known as a datagram delivery service.

Figure 2.28 shows a fragment of an internet's topology. Each router is attached to physical links by its interfaces. Note that a multipoint link can have many routers attached to it. All the devices attached to a link are each

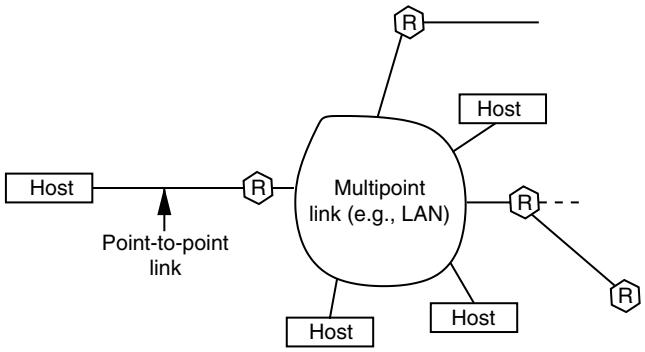

Figure 2.28 A fragment of an internet, showing routers (labeled R), hosts, and links (including multipoint "links," such as LANs).

other's neighbors. The Internet is equipped with *routing protocols* that permit routers to identify good paths on which to forward packets. These protocols work on the basis of metrics assigned to the network links and a distributed algorithm for determining shortest paths under these metrics. Note that a routing protocol is also an application running on the network! Hence it needs to use the packet transport services of the network. Fortunately there are distributed algorithms that can learn shortest paths through the network, and, in the execution of these algorithms, nodes need only exchange packets with their neighbors. Hence routing in an internet can bootstrap itself. A simple protocol (aptly called the Hello protocol) is used by routers to discover neighbors. After neighbors are discovered they begin to exchange routing protocol packets, which are used in computations that gradually lead to all the routers to learn shortest paths to network prefixes. One such distributed algorithm works by each router informing its neighbors about the status of the links to which the router is attached. This information is *flooded* through the network. Eventually every router obtains these *link state advertisements* (LSAs), which can be put together to obtain a full topology view in each router. A shortest path computation can then be locally performed and routing tables built up in each router. This algorithm is implemented by the currently most popular routing protocol, OSPF (Open Shortest Path First). The OSPF protocol is a routing application protocol, but it does not utilize the services of a transport protocol, instead running directly on the IP layer in routers.

The basic Internet packet transport does not promise any QoS to the traffic streams it carries. The network provides connectivity and routing; any

user attached to the Internet can initiate multiple flows. There is no *connection admission control* (CAC); all the flows end up sharing the network resources. Thus the Internet transport provides a highly variable quality of service that can vary widely depending on geographical location and time of day. To bring some sanity to the situation and to enforce some fairness in bandwidth sharing, the end-to-end Transmission Control Protocol (TCP) implements an adaptive window protocol. This protocol reacts to packet losses by reducing its window size and then slowly building it up again. This function of *congestion control* and *bandwidth sharing* is in addition to the two other functions that TCP performs: (1) reliable and sequential packet transport over IP's unreliable and nonsequential packet transport service, and (2) sender–receiver flow control (which prevents, for example, a fast computer from flooding a slow network printer). Chapter 7 discusses TCP's congestion control and bandwidth-sharing function at length.

UDP (User Datagram Protocol) is another popular layer 4 protocol. This protocol simply permits a user above IP to utilize the basic datagram delivery service, thereby multiplexing several flows into one IP address. Note that this is the *logical* multiplexing of several flows originating and terminating at a common IP address. It should be distinguished from the physical multiplexing of flows into a bit carrier, something that is a link layer function. We distinguish the different flows by assigning them different UDP *port numbers*. UDP is used by applications, such as packet voice telephony, that must receive guaranteed service rates in the network and cannot deal with packet loss by retransmission, and hence cannot use the services of TCP. Other mechanisms, above the transport layer, are used to facilitate such applications.

We have stated that the Internet's packet transport is not designed to provide specific QoS to the flows it carries. The service model is quite simple. Nodes that have valid IP addresses can attach themselves to the network and send IP packets back and forth between themselves. The network provides an unreliable, nonsequential packet transport service, with no guarantee of transfer rate or delay, often called best-effort service. The network does not distinguish between the various traffic flows; it does the best it can, treating everyone alike. The idea is that the applications best know what they need and should adapt to the transport that the network provides by using end-to-end mechanisms. TCP's mechanisms for achieving a reliable and sequential packet transport service over the Internet, and some sort of fair sharing of network bandwidth, are a prototypical example of the end-to-end approach. Over the past decade, however, the Internet has become *the* packet transport network of choice for all kinds of store-and-forward data transfer and increasingly is being used by applications that require some minimal quality of packet transport. Broadly there are two approaches that can be followed

to provide some level of QoS over the Internet: new QoS architectures and traffic engineering.

Two QoS architectures have been proposed and extensively studied: the *Integrated Services Architecture*, abbreviated as IntServ, and the *Differentiated Services Architecture*, abbreviated as DiffServ. The proposals in the IntServ architecture essentially allow each session arriving to the network to request QoS guarantees; it must declare its traffic characteristics to the network, and the network has the choice of rejecting the request or accepting it at some lower level of QoS. This architecture requires signaling protocols to be put in place, and packet-scheduling mechanisms are needed at the packet-multiplexed links. Evidently, these protocols and scheduling mechanisms need to be implemented in every router in parts of the network over which such QoS guarantees are needed.

In the high-speed core of the network, the session arrival rates and the packet rates are too high to permit session-by-session analysis and packet-by-packet scheduling. Hence, as in other transport systems (e.g., airlines and railways), it has been argued that only a few levels of differentiation may suffice (e.g., first class, business class, and economy class). The classes are, of course, priced differently. At the simplest level, there may be one priority class of traffic (reserved, for example, for interactive packet telephony), and another class for the remaining traffic. There could be one additional class for the premium store-and-forward traffic, and the remaining traffic could be handled by the default best-effort packet transport. This idea has led to the DiffServ proposals. The scheduling at the links distinguishes a few classes of packets (e.g., 2, 3, or 8, depending on the choice of the network operator). The class of a packet is identified by the contents of its header; such classification can be based on six special bits in the IP header, called the *DS code* (DiffServ code), and could also be based on source and destination addresses and even the source and destination transport protocol port numbers. The schedulers are aware only of such packet *aggregates*; there is no awareness of individual so-called microflows. A DiffServ core network may put a limit on the amount of traffic of each class it is willing to accept from each customer network that transports traffic through it. If a network violates such restrictions, the DiffServ core can reject the excess traffic or handle it at lower levels of service. It is up to the edge nodes of the customer network to police the traffic that it offers to the DiffServ core.

It has been argued that in conjunction with an IntServ architecture in the lower-speed edges of the network, a DiffServ architecture would suffice in the core to provide an overall end-to-end QoS to applications.

The other approach for providing QoS over the Internet is traffic engineering. Clearly, if there is sufficient bandwidth the best-effort packet transport suffices.

With the rapid deployment of optical networks, even to within a few hundred meters of network access points, it is becoming easier and less expensive to quickly deploy additional bandwidth where there are bottlenecks. Hence, it has been argued that it suffices to manage QoS by network and traffic engineering. The network topology (node placement, node interconnection, and link capacities) should be properly designed with the expected traffic in mind, and it should be tolerant of link failures. It also should be possible to deploy new bandwidth when needed. Furthermore, in the operating network the traffic should be carefully routed to prevent the formation of bottlenecks, and the routing should be monitored and revised from time to time. Traffic engineering alone, of course, cannot address the problem of congestion in access networks.

Finally, to end this section, we turn to the components of Internet hosts and routers. Figure 2.29 shows the protocols implemented in a typical Internet host. This host can handle email because it has SMTP (Simple Message Transfer Protocol) and can browse the Web because it has HTTP (Hyper text Transfer Protocol); both of these protocols run over TCP. The host can also participate in a packet voice call, something that requires the RTP (Real Time Transport) protocol, which in turn runs on UDP's simple datagram service.

Now let us look at the components of an Internet router (see Figure 2.30). A router is a packet switch and hence moves packets between several links. In the figure, the router shown has three links: a link to a *wide area network* (WAN), a link into a LAN, and a *dial-up link* (into which a connection can be made over the telephone network). There is a link protocol for each physical link: HDLC (High-Level Data Link Control) for the WAN link, the IEEE standard link protocol

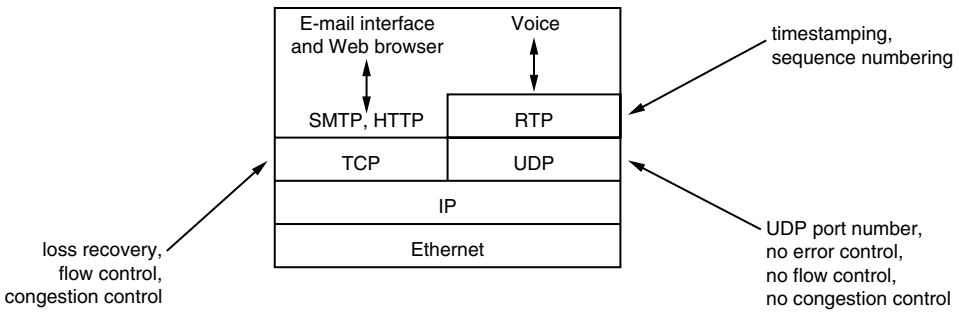

Figure 2.29 The typical protocols in an end system (or host) attached to the Internet.

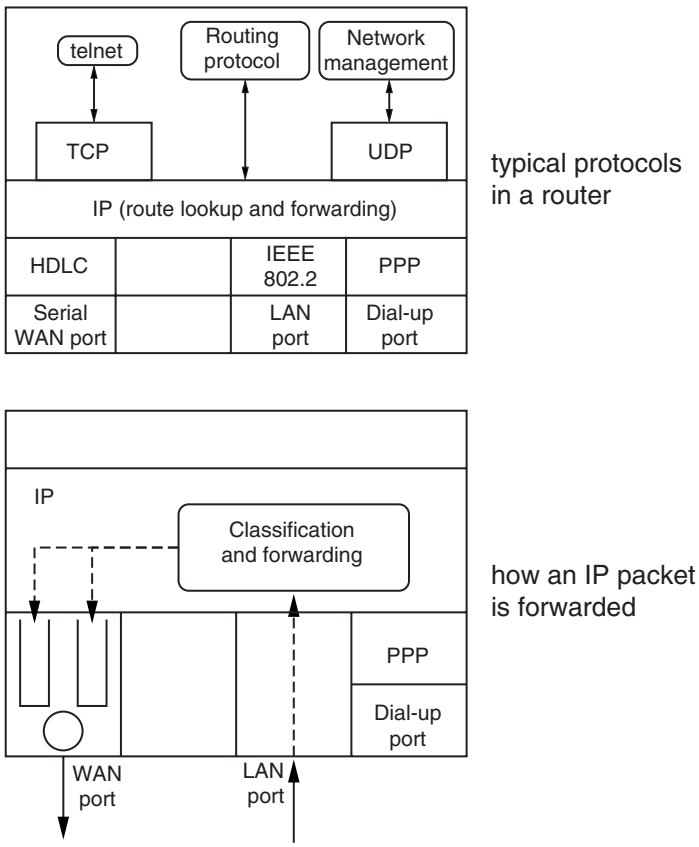

Figure 2.30 The protocols in a router (top) and the way an IP packet is forwarded from one interface of a router to another (bottom).

(IEEE 802.2) for the LAN link, and PPP (Point to Point Protocol) for the dial-up link. The IP layer runs across all these link layers and forwards packets between them. The packet switching could simply be done by reading into and copying out of the processor memory, or there could be a hardware switching fabric in a high-capacity router; the switch is not shown. In addition to these components (which were also depicted in Figure 2.27), there is a routing protocol (shown here directly over IP, as is the case for OSPF). There is *telnet*, a protocol for permitting an administrator to log in to the router to configure its parameters; because telnet

requires TCP, TCP must also be implemented on a typical router. Furthermore, there is a network management protocol that is used to monitor the traffic in the router and the status of its links. As mentioned earlier, SNMP is the commonly used protocol for network monitoring and management in the Internet, and, as shown in Figure 2.30, it operates with UDP as the transport protocol. The bottom part of Figure 2.30 shows what is involved in routing a packet from one port to another. Not only must IP do forwarding, but there is also the need to classify the packets into QoS classes to appropriately queue them at the output port; two queues are shown in the figure, perhaps a high-priority queue for voice packets and a low-priority one for data packets.

2.3.6 Asynchronous Transfer Mode (ATM) Networks

The development of ATM technology was motivated by the desire to implement a Broadband Integrated Services Digital Network (B-ISDN), which is a generalization of ISDN (see Section 2.3.3). Instead of access rates in multiples of 64 Kbps as in ISDN, B-ISDN supports access at almost any rate that the user desires. ATM offers a specific technique of packet switching and multiplexing onto a communication link. The packets generated by sources of information are, in general, of variable size. The most prominent feature of ATM is that it is designed to multiplex, switch, and transport small, fixed-size packets called *cells*. Each cell in an ATM network is 53 octets long and consists of a 5-octet header and a 48-octet payload.

The small size of the ATM cell can offer useful advantages. The fixed size helps in *cell delineation* at very high speeds. That is, identifying the boundaries of a cell becomes simpler because of the fixed size. Also, cell switching at very high speeds becomes possible because the units to be switched (i.e., cells) are always of the same size.

To see another advantage, suppose that packetized voice is being carried on the network along with bulk file transfer traffic and that the two information flows are to be multiplexed onto the same link. In such a case, it is natural to give priority to the voice flow so that the cell carrying digitized voice is transmitted as soon as possible. If the first cell generated from the packetized voice application arrives just after the first cell from the file transfer application, then the voice cell will have to wait at most one cell transmission time, while the file transfer cell is being transmitted. Because cell sizes are small and fixed, it is clear that the waiting time for a delay-sensitive application such as packetized voice is kept small. This would not have been possible for arbitrary packet sizes. However, it is also worth noting that the small cell size in ATM helps more at lower link speeds

than at higher link speeds. As link speed increases, the time taken to transmit large packets reduces, and the nonpreemption delay seen by a voice cell may not be significant.

On the downside, a 5-octet header in a 53-octet cell implies a fixed overhead of 9.4%. Owing to the fixed cell size, it is not possible to amortize the overhead over large packet sizes.

The word "Asynchronous" in ATM distinguishes it from Synchronous Transfer Mode (STM) techniques. You can see the distinction by considering the ways in which STM and ATM utilize the raw transmission capacity provided by the physical layer below (see Figure 2.31). As mentioned in Section 2.3.1, the capacity is provided by a digital transmission standard, in which frames are defined. When the STM is used, a particular connection's data can be found at the same relative positions in successive frames, as shown in Figure 2.31. However, when ATM is used, the data is inserted into the next available free slot, and specific positions are not maintained. It is possible that some frames do not contain any data from a particular connection because the connection had no data to send. In this case, the slots can be used to carry traffic from other connections. On the other hand, it is possible that a connection generates a burst of data so that the full burst cannot be

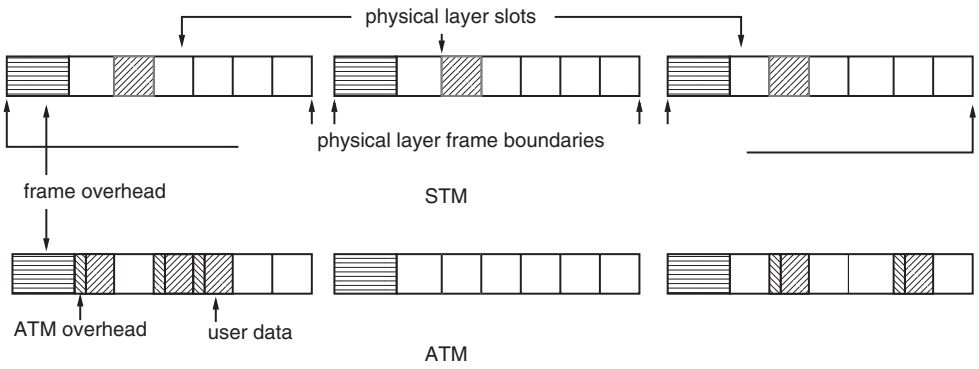

Figure 2.31 The top panel shows a particular connection's traffic carried using STM on a digital bit carrier. The traffic occupies the same relative positions in successive frames. The corresponding slots are shaded. The bottom panel shows a connection's traffic carried using ATM on the same carrier. The traffic is pushed into the frame without maintaining the same relative position in every frame. When ATM is used, part of the bandwidth resources are wasted because each ATM cell has an overhead.

carried in one frame; this causes queueing at the interface between the ATM and physical layers. The excess data would then be carried in free slots in the ensuing frames.

Of course, we cannot expect that the applications that are the sources of information will generate fixed-size packets. This is particularly true because many existing applications that were designed before the advent of ATM made no attempt to generate fixed-size packets. But if ATM technology is to be used, all units to be transported must be 48 octets long. We achieve this by using an ATM *adaptation layer* (AAL), whose purpose is to segment large packets into cells at the transmitting end and then reassemble the original packets from the corresponding cells at the receiving end. Thus, the AAL makes it possible for legacy applications to utilize ATM transport without requiring any change in the applications.

From the point of view of the OSI model (see section 2.3.2), ATM technology is a way to implement *both* the network layer (layer 3) and as the link layer (layer 2). First, ATM does not require any particular link layer technology (for example, Ethernet) to carry the *protocol data units* generated by it. In fact, ATM defines its own link layer. In contrast, in the Internet, the IP layer is strictly at layer 3 and requires an underlying link layer to carry the packets generated by it. Moreover, ATM provides full routing capabilities through a mesh of ATM switches. For this purpose, the Private Network-to-Network Interface (PNNI) Protocol has been defined in the ATM Forum (the standardization body for ATM technology). Hence, ATM technology also implements layer 3 functionality, of which routing is a typical example.

As in any layered communication architecture, a layer may offer connection-oriented or connectionless services to the layer above it. The ATM layer was designed to offer connection-oriented services to the AAL above. *Connection-oriented* service implies that an association between the two communicating end points must be set up before information transfer begins. Also, as part of the connection setup procedure, a *traffic contract* is negotiated between the ATM layer and its user. This contract characterizes the cell traffic that the ATM connection is supposed to carry and also specifies the QoS expected by compliant traffic. Because a commitment to QoS is being made, the ATM layer must to reserve adequate resources in the network. If sufficient resources are not available at all ATM switches along the path, then a connection setup request can be blocked.

Going up the protocol stack, the service offered to the user of the AAL can be connection-oriented or connectionless, as shown in Figure 2.32. Thus, applications that expect connection-oriented service as well as those that expect connectionless service can be seamlessly carried over the ATM network.

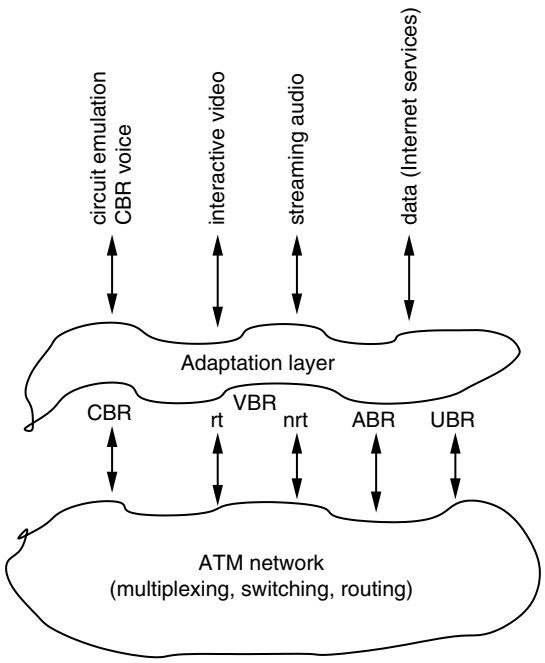

Figure 2.32 The ATM layer offers connection-oriented services to the AAL above it. But the service offered to the user of the AAL can be connection-oriented or connectionless. Different AALs provide different services. The AALs that have been standardized are AAL 1, AAL 2, AAL 3/4, and AAL 5. Some applications that utilize the services of the AAL are also shown.

Unlike the Internet, ATM networking technology was designed with QoS in mind from its conception. As we have explained, the small cell size was chosen to reduce packetization and nonpreemption delays when carrying packetized voice over an integrated services packet network. In addition, the ATM Forum has defined five types of standardized service categories that can be offered by the ATM layer (see Figure 2.32). The Constant Bit Rate (CBR) service is a vehicle for carrying traffic that has a regular periodic structure. Traffic that is less regular can utilize the *Variable Bit Rate* (VBR) service, of which there are two types: *non-real-time* and *real-time*. If the application generating the variable-bit-rate traffic can characterize it in terms of the maximum burst size and the peak and average rates, then an appropriate VBR connection can be set up. For example, a packet voice

application can often specify its maximum burst size by considering the peak rate at which the voice coder emits bytes and the duration of a talkspurt. Delay-sensitive applications such as interactive video would choose a real-time VBR connection, whereas delay-tolerant applications such as streaming audio and streaming video can be carried by a non-real-time VBR connection.

Whenever the ATM network accepts a request for setting up a CBR or VBR connection, it must to commit bandwidth and buffer resources for carrying the corresponding traffic. Because a given ATM switch normally handles many connection requests, groups of connections must be multiplexed onto outgoing links. Each CBR source must be allocated a bandwidth equal to its declared rate. For VBR sources, however, the resource allocation algorithms running on the switch can take advantage of the statistical nature of traffic generation and can multiplex more connections than simple peak-rate allocation would suggest. Note that, because resources are limited, an ATM switch cannot keep accepting CBR and VBR connection requests indefinitely. This implies that the resource allocation algorithm must also have connection admission control (CAC) functionality so that it can refuse requests when resources are overallocated.

Typical data applications, however, do not have a simple way of characterizing the traffic they generated. Moreover, data traffic can tolerate variable delays, and applications can actually modify their data transfer rates according to network conditions. These factors indicate that it is hard to carry data traffic efficiently using the CBR and VBR services. The Available Bit Rate (ABR) and Unspecified Bit Rate (UBR) services are designed to carry data traffic.

The ABR service indicates to the application the amount of the bandwidth available in the network for carrying its traffic. The source of the traffic is then expected to modify its rate of data generation accordingly, in response to feedback from the network regarding resource availability. In practice, resource management (RM) cells, generated at regular intervals by traffic sources, are used by the network to indicate the amount of available resources. A network switch can provide information indirectly, by indicating that it is getting congested; alternatively, it can explicitly indicate the data rate that it can handle. Together, the traffic source using the ABR service and the network switches form a closed-loop feedback system (see Chapter 7). Users of the ABR service can expect some level of performance guarantees from the network; for example, a source can ask for and obtain a *minimum cell rate* (MCR) guarantee.

From this discussion, it appears that applications using the VBR service generate variable-bit-rate traffic, and so do applications using the ABR service. What, then, is the difference between the two? Consider a long file being transferred across the network. The significance of the long file is that the source

always has data to send. Now, variability in ABR traffic occurs because the source adapts its rate of traffic generation in response to network feedback. On the other hand, variability in VBR traffic is completely independent of network conditions. It is an intrinsic property of the source, determined by its own characteristics, such as distributions of talkspurts and silence periods.

The UBR service is also meant for carrying data traffic. It is a typical best-effort service without any guarantees. Essentially, UBR traffic is treated as low-priority traffic that is carried using network resources that are not demanded by other categories of higher priority. The network makes no effort to indicate resource availability to a UBR source. Because no resources are committed, the bit rate is left unspecified.

As mentioned before, the ATM layer offers connection-oriented services to layers above it, and hence, before data transfer can begin, a connection between the communicating end points *at the ATM layer* must be established. This ATM-level connection must extend right through the ATM network, from one end point to the other. This is called a virtual circuit (VC).

Because the VC extends across the complete ATM network, it crosses several ATM switches. Hence, a VC consists of several *legs*, with a leg being the segment of the connection between a pair of neighboring switches. To identify a VC, therefore, an ATM switch must maintain an association between the incoming leg of the VC on an input port of the switch and the outgoing leg of the VC on an output port. At the time the connection is established, each ATM switch assigns *local identifiers* to the VC legs and stores the labels in a table in its memory. The idea is shown in Figure 2.33. For example, the first row of the table shows that for a VC passing through this switch, the incoming leg on input port 1 has been assigned

Input port	Input label	Output port	Output label
1	22	4	16
6	49	9	27
4	13	7	45

Figure 2.33 An ATM switch maintains a map between labels identifying the incoming and outgoing legs of a VC passing through it. This table shows that the label 22 on input port 1 is associated with the label 16 on output port 4.

the label 22, and it is associated with the outgoing leg label 16 on output port 4. Note that the labels are meaningful only to the switch that assigns them. Thus, labels have local significance only. A user of an ATM VC does not need to know the complete sequence of labels to identify the VC. It is sufficient for the user to insert into the cell the correct label on the *first* leg only. The tables maintained in the ATM switches ensure that the cell is forwarded correctly along the VC that has already been set up. In fact, as the cell proceeds along the VC, the ATM switches remove and insert labels by consulting the forwarding table. This is called *label swapping*.

ATM uses a hierarchical labeling scheme having two levels. A label consists of the *virtual path identifier* (VPI) at the "higher" level and the *virtual circuit identifier* (VCI) at the "lower" level. The label associated with a VC at a particular switch port is specified by the VPI–VCI combination. However, the hierarchy indicates that a group of VCs that share the same VPI, but differ in the VCI, can be treated as a unit for switching. For example, if there are 10 VCs, with the VPI of each being 22 and the VCIs being $5, 6, \ldots, 14$, then this group can be treated as a single unit by virtue of the common VPI. Such a bundle of VCs is called a *virtual path* (VP). The idea is shown in Figure 2.34.

When a cell arrives at a port, the switch searches its table to identify the outgoing port and the new label that must be inserted. As the number of VCs passing through the switch increases, the time taken to search the table goes up. This indicates a possible scalability issue: As the number of VCs increases, a switch may not be able to cope with cells arriving rapidly, because it does not have sufficient time to search the large table in its memory. In this situation, the VPI serves as a handy tool for aggregation. If only the VPI is considered as a label instead of the VPI–VCI combination, there will be fewer entries in the

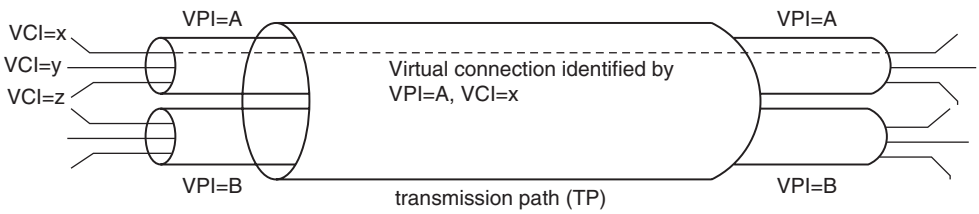

Figure 2.34 A schematic showing the relationship between VCs, VPs, and the transmission path (TP). The transmission path is a concatenation of one or more physical links.

table and searching will take less time. We can think of VPI-based switching as switching at coarse granuarity, whereas VPI–VCI-based switching occurs at finer granularity. Switching devices that consider only VPIs instead of the VPI–VCI combination are called crossconnects. Because the label associated with a connection at a crossconnect is the VPI only, label swapping means swapping of the VPIs alone; the VCI carried in the cell header is not modified by a crossconnect.

VCs can be classified according to whether they are *permanent virtual circuits* or *switched virtual circuits* (PVCs or SVCs). (A corresponding classification can be made for VPs.) A PVC between two end points is set up manually by a network operator. VPI and VCI values are defined for the permanent connection, and the values are entered in the table of each switch in the path. PVCs normally last for long periods: days or months. In contrast, an SVC is set up for a specific call by means of signaling between the user and the network and lasts for the duration of the call: minutes or hours. Thus, the timescales associated with setting up and clearing PVCs and SVCs are very different.

Standard interfaces have been defined in the ATM architecture so that devices from different manufacturers can interoperate. Figure 2.35 shows the main interfaces.

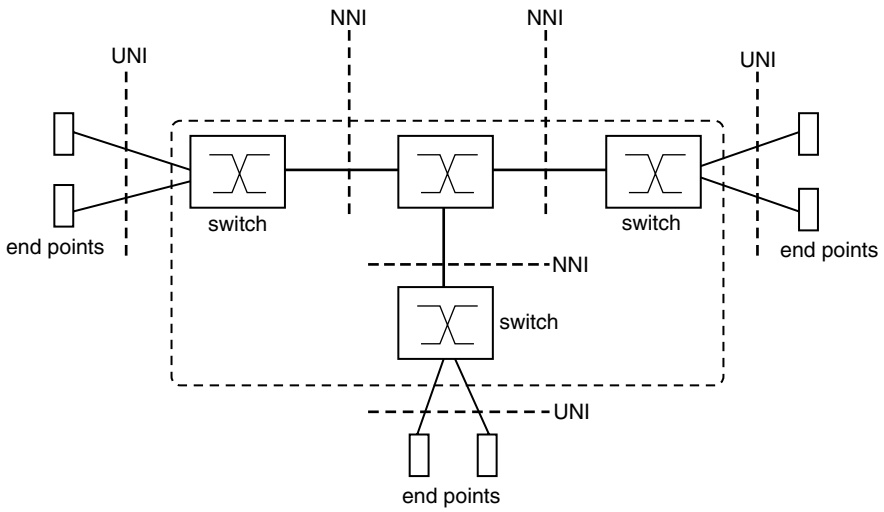

Figure 2.35 The user network interfaces and network node interfaces in an ATM network.

The *user network interface* (UNI), as the name implies, is the standard interface between the end-user equipment and the ATM devices inside the network. Communication across the UNI is structured in several planes (see Figure 2.22). The user plane is one in which user information is transmitted. Signaling information between the user and the network, used in establishing and releasing VCs, is exchanged in the control plane. Finally, the management plane is used for conveying various types of information related to management of the end-user equipment and ATM devices in the network.

The end result of standardization is the specification of protocols according to which peer entities at a specific layer interact. For example, it has been decided that when the end user wishes to initiate a switched virtual circuit across the UNI on the control plane, the relevant protocol will be DSS2 (Digital Subscriber Signaling System No. 2, described in ITU-T Q.2931) at the application layer. This is a part of the SS7 suite of signaling protocols, which is used in the telephone and ISDN networks (see section 2.3.3).

The ATM *network node interface* (NNI) is used between nodes (ATM crossconnects and ATM switches) in the network. As with the UNI, the purpose of standardization here is to fix protocols at different layers so that compliant devices from different manufacturers can interoperate. For example, on the control plane, the protocol that two ATM switches should talk in order to set up a switched virtual circuit is B-ISUP MTP 3 (Broadband Integrated Services User Part, Message Transfer Protocol 3). As discussed in Section 2.3.3, the SSP functionality in the ATM switch is responsible for creating and processing the MTP3 signaling packet as well as transmitting it over the signaling network.

2.4 Summary and Our Way Forward

In Section 2.1 we argue that networking is basically about resource sharing, with the basic functional elements being multiplexing, switching, routing, and management. Section 2.3 provides a fairly detailed overview of the manner in which many of the telecommunication technologies carry out the functions of networking.

From this discussion we argue that the layered model will not serve us well in organizing our understanding of telecommunication network design and analysis. Rather, we will be well served by studying the various functions, the way these functions can be performed, and the way interact or are isolated. As an example of isolation of the design of the functional blocks, recall the discussion of switches for circuit-multiplexed networks. The call-blocking probability depends on the blocking probabilities in the switches and the links on the path.

The switch-blocking probability is made negligible compared with the link-blocking probability and is ignored in the link capacity design. The problem of designing switches with low blocking is taken up as a separate problem. In packet networks, in principle, routing and multiplexing can be considered together, but in practice they are treated as being separated by operating over different timescales. On the other hand, in wireless networks over a limited radio spectrum and time-varying channels, the functions at several layers may need to be studied together. Thus, we adopt the functional approach, and the rest of this book is divided into three parts: one each to discuss multiplexing (Part I), switching (Part II), and routing (Part III). Our approach to the presentation of the material is that instead of just formally developing a battery of models and the associated results, we use engineering issues to motivate the presentation of the theory. The results obtained from the theory are then used to suggest solutions and to explain how these engineering issues are addressed in current practice. We expect that this approach will provide a more useful presentation for the network practitioner and the practical performance analyst.

2.5 Notes on the Literature

In this chapter we provide a point of view about communication networking. We explain the "big picture" of networking as it is practiced today. Finally, we provide additional perspectives to support our approach in adopting a functional (rather than layered) approach in organizing this book. The following is a representative survey of books that helps to supplement the overview of networking that we have provided.

Descriptive, textbook treatments of the vast area of computer networks are provided in the books by Tanenbaum [282], Keshav [169], Peterson and Davie [238], and Kurose and Ross [187]. The current most authoritative book on optical networking is the one by Ramaswami and Sivarajan [245]. A comprehensive book on the basics of wireless mobile communications is the one by Goodman [124]. Detailed coverage of the OSI model is provided in Jain's and Agarwala's book [152].

One of the best resources for understanding the telephone network is the Bell Systems operations handbook [252]. The book by Bellamy [24] is a good source book for all aspects of current practice in digital telephone networks, including access network design, backbone transport systems, and capacity design. It also covers some of the newer access technologies such as xDSL, as well as wireless and transport technologies such as SONET and SDH. The handbook by Freeman [109] is another source that also has

a discussion of the engineering of the analog networks. SS7 is very well covered by van Bosse [290].

The book by Stallings [272] provides an overview of ISDN, Frame Relay and ATM networks. For ATM networking alone the books by de Prycker [240] and by Höndel et al. [135] provide very detailed coverage of the concepts and standards. For Internet technology the book by Keshav [169] provides an insightful treatment. The most authoritative coverage of Internet protocols is found in the books by Comer and Stevens [70] and by Stevens [275].

The book by Rose [257] is the first on SNMP. The books by Subramanian [280] and Stallings [273] describe the most recent versions of the SNMP and RMON standards in detail. Of course, the Requests for Comments (RFCs) from the Internet Engineering Task Force (IETF) are the original source for all the SNMP standards.

The books by Kershenbaum [168] and Cahn [50] are good references on the topic of network topology design, which is not covered in this book.

Problems

The following can be called "toy" problems. They require reader you to carry out elementary calculations for some simple models that arise in the context of the functional elements discussed in this chapter. Solving these problems will help illustrate some engineering issues and will also review some of the basic analytical prerequisites required for reading this book.

2.1 Consider a channel in which bit errors occur independently from bit to bit, and the bit error probability (or bit error rate) is p. This is called the Bernoulli packet error model. Show that the probability that a K-bit packet is received in error is approximated by Kp. Examine the validity of the approximation for $K = 10,000$ bits, for $p = 10^{-7}$, and for $p = 10^{-4}$, and discuss. If the packet length is a random variable, K, with $f(k)$ being the probability that the packet has k bits, show that the packet error probability is approximated by $p\mathsf{E}(K)$.

2.2 A simple model to introduce memory in the error process is as follows: A bit error follows another bit error with probability p, and a correct bit follows another correct bit with probability q. The channel state can be represented by a two-state Markov chain, as shown in Figure 2.36. If bits

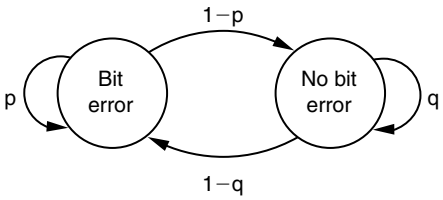

Figure 2.36 Markov error model for problem 2.2.

are being continuously transmitted on the channel, find the probability, π_e, that a random bit is in error. Find the probability that a K-bit packet, randomly located in time, is in error. For what values of p and q do we recover the Bernoulli packet error model with bit error probability p?

2.3 A P bit packet is to be transported over L store-and-forward links; d_l and C_l are the propagation delay and bit rate, respectively, on link l. Assuming zero queueing and processing delays on the links, argue that the end-to-end delay would be $\sum_{l=1}^{L}\left(\frac{P+H}{C_l}+d_l\right)$, where H is the number of header bits in a packet. If an M bit file is to be transmitted as K packets and if $C_l = C$ for all l, show that the end-to-end file transfer completion time is $\frac{1}{C}\left[\left(\frac{M}{K}+H\right)(L-1)+\sum_{l=1}^{L} d_l + M + KH\right]$. Discuss the trade-offs of choosing K. Show that the K that minimizes the delay is $\sqrt{\frac{M}{H}(L-1)}$.

2.4 If the links in problem 2.3 used cut-through switching rather than store-and-forward, obtain the end-to-end delays for the packet and the file with K packets. Assume $C_l = C$ for all l.

2.5 A circuit-multiplexed point-to-point link can accommodate one call at any time. The arrival times of calls X and Y are uniformly distributed in the time interval $[0, 5]$. The holding time of call X is one unit, and that of Y is two units. An arriving call that finds a busy trunk is lost. Find the blocking probability for calls X and Y.

2.6 A source generates bits according to a Poisson process of rate λ. The bits from the source are to be transmitted as K bit packets. The time to accumulate a full packet will be called the packetization delay. Find the distribution of the packetization delay.

2.7 A video source generates K fixed-length packets per frame, which are then transmitted over a link. The frame rate is 25 frames per second; that is every 40 ms the source outputs K packets into the link's buffer. The transmission time of each packet on the link is 1 ms. If $K \leq 40$, find the average packet delay and the time average occupancy of the link buffer. Assume that a packet is stored in the buffer until its last bit has been transmitted. Now assume that K is a random variable with probability mass function $\Pr(K = k) = \alpha^{k-1}(1-\alpha)$ for $k = 1, 2, \ldots$, with $0 < \alpha < 1$. In each frame, packets in excess of 40 are lost. Find the probability of losing packets in a frame and the average number of packets lost per frame. Also, find the average packet delay and the average buffer occupancy.

2.8 In a network with distributed multiplexing using random access, assume that the distance between all node pairs is a. Can you think of a physical network in which this could be possible?

2.9 In a slotted multiaccess system, K nodes are each attempting transmission with probability p in each slot independent of the other nodes. If N is the random number of slots consumed before the first success, find the probability mass function of N.

2.10 Three kinds of polling can be identified: token ring network, in which N nodes are arranged in a physical ring and a token is passed around the nodes to control access to the ring; *hub polling* on a *token bus* network, in which the token is passed over the broadcast bus to the "next node" in the sequence; and centralized roll call polling, where a central node sends an explicit query to each node in the network and receives an ACK/NACK. Identify the minimum delays between consecutive opportunities to transmit (called the *walk time*) in each of the three types of polling.

2.11 $N + 1$ nodes, numbered $0, \ldots, N$, are located on a straight line starting at the origin, and node i is at distance i from the origin. Each node is connected to a switch nearest to it. Assuming $N = 2K$, what would be the total length of the transmission links with only one switch at location K? Repeat the calculation for a network with $N = 4K + 1$, switches at K and $3K + 1$, and a transmission link connecting these switches.

2.12 To get a perspective on packet and call timescales, consider a switch with 10 100 Mbps links. If the packet lengths can vary from 500 bits to 12,000 bits, with an average packet length of 1000 bits, determine the

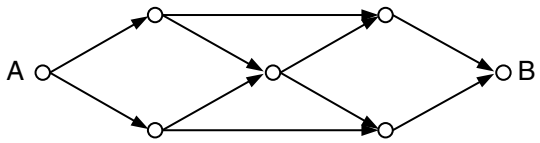

Figure 2.37 Network for problem 2.14.

maximum and average packet arrival rate to the switch. If stream traffic requires 50 Kbps per call, with a call being active for 200 seconds, what is the maximum rate at which accepted calls arrive? Compare the mean time between packets and between call arrivals.

2.13 Consider a hierarchical construction of a network. At the lowest level, the nodes are formed into groups of K_1 nodes and are connected to one level-1 switch; K_2 level-1 switches are grouped together to be connected to one level-2 switch. Let there be L levels of this hierarchical construction. For $K_i = K$, and $i = 1,\ldots,L$, find the number of switches in the network and the number of links. Note that the links interconnecting the switches at the higher levels will be longer, but because they aggregate more traffic, they will also be of increasing capacities.

2.14 How many possible routes are there between a node pair in (a) a linear network where the nodes are arranged in a straight line, (b) a ring topology network, (c) a star network, and (d) a fully connected network, and (e) between nodes A and B in the network shown in Figure 2.37?

2.15 One link in a network is failure-prone, and the other links are stable. The network topology is sampled every second, and, if the topology has changed, the routes are recalculated. Each second, the failure-prone link changes from up to down with probability p, and from down to up with probability q. What is the rate at which the routes in the network are calculated?

2.16 On a link the mean delay can be approximated by $\frac{1}{1-\rho}$, where ρ is the utilization of the link. If the network monitoring and management traffic increases the utilization by α, find the percentage increase in the mean delay and plot the increase as a function of ρ. What do you observe?