CHAPTER 3

QUALITY OF SERVICE IN IP NETWORKS

JOBERTO SÉRGIO BARBOSA MARTINS

INTRODUCTION

Transmission control protocol/IP (TCP/IP) has become a de facto platform for most of the computer systems in the world. Besides that, it is a common well-known fact that IP has to be somehow "adjusted" to work adequately with multimedia and other general applications, becoming very popular in networks, including the Internet and other TCP/IP networks. QoS provisioning to application and end users in IP networks then becomes the challenging task for network designers, managers, researchers and, generically, technical staff.

The objective of this chapter is to elaborate on this subject by presenting an introductory overview of the main technical solutions available for QoS in IP networks. We will first elaborate on the importance of this field and then review the most important solutions and propositions for QoS provisioning in IP networks.

The plan of the chapter is as follows:

Section 3.1, "IP Context and Quality of Service," introduces the importance of the field, identifies the overall perspectives and makes an attempt to foresee IP evolution.

Section 3.2, "Quality of Service," examines some common basic principles for QoS, presents a taxonomy for applications, and introduces the most important basic parameters frequently used to characterize, analyze, and manage QoS requirements.

Section 3.3, "Quality of Service: Approaches and Initiatives for IP Networks," summarizes the main approaches used to control and manage QoS and presents some additional basic concepts.

Managing IP Networks. Edited by Saleh Aidarous and Thomas Plevyak
ISBN 0-471-39299-5

Section 3.4, "Packet Conditioning, Queue Scheduling and Congestion Control in Routers," explores the techniques used by routers for QoS provisioning.

Sections 3.5 and 3.6 sequentially present the Integrated Service Architecture and the Differentiated Services Architecture, examining their goals and implementation issues.

Section 3.7, "Multiprotocol Label Switching," idescribes multiprotocol label switching (MPLS) technology in the context of the IP network.

Selected references are listed at the end of the chapter in order to provide guidance for additional reading.

3.1 IP CONTEXT AND QUALITY OF SERVICE

Quality of service (QoS) is a very important issue for computer networks in general and for IP networks in particular.

Over the past few years, data networks have been growing at enormous rates while, simultaneously, a totally new set of applications has emerged. Generically, these new applications can be broadly grouped as multimedia applications, since they manipulate many different media types, e.g., real-time applications, mission-critical applications, or another specific denomination (Fig. 3.1). This new set of applications needs to be supported by the "conventional" data networks in use, which, in turn, are mainly TCP/IP networks. Therefore, an understanding of the QoS concept and solutions is essential to design, implement, operate, maintain and manage TCP/IP networks adequately in this new context.

The discussion about quality of service at first requires an answer to this very simple question: How can the term "quality of service" be defined considering the application

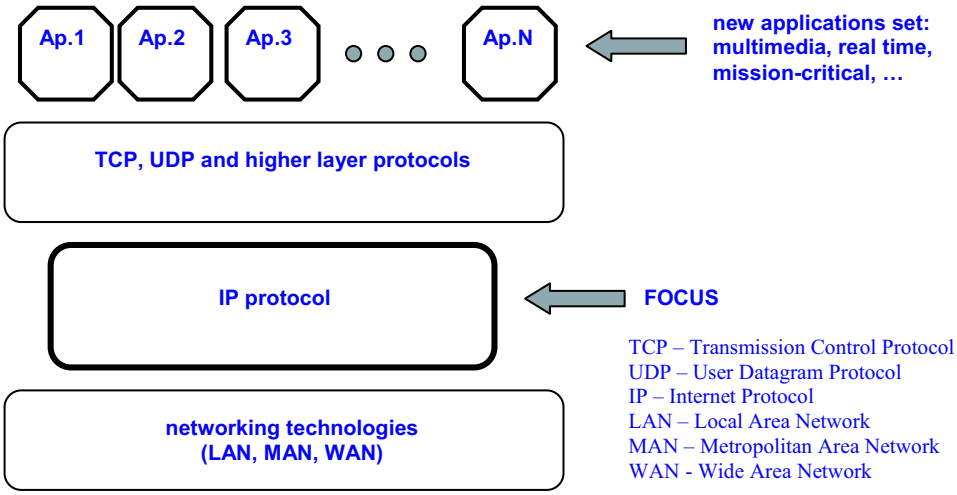

Figure 3.1 QoS and TCP/IP.

scenario just mentioned? In the following introductory sections we elaborate on the basic concepts about QoS before discussing the strategies used in IP networks to achieve a "QoS-ready network."

3.1.1 IP Network Context and Perspectives

To understand how the QoS issue is important requires a general overview of the widespread use of computer networks and their importance in our lives and society. This discussion is beyond the scope of this text, but the general context and the perspectives involved are easily identifiable. In effect, there is a plethora of networked users, and certainly none would possibly appreciate the chaotic effect of turning off all these very practical and troublesome "appliances." Industry, banks, retail stores, universities, schools, in short, all segments of our "civilized and efficient" society, will simply not work as expected without the "networks."

The estimated installed number of IP users is impressive and, beyond that, is continuously growing in all numbers available. The very strong growth of IP-installed base and widespread use are measurable effects of well-known facts:

- The explosive growth of the Internet global network and its continuous dissemination among end users all over the world;
- The de facto standardization of IP as a network platform for supporting distributed applications by companies and institutions in almost all areas of business.

Certainly, the use of IP as a network platform (Figure 3.1) for distributed applications could be understood as a consequence of Internet success. For developers, IP is an extremely natural "business option," as the number of users is large and, fueled by the World Wide Web (WWW), will continue growing over the long term.

At this point, a fundamental question may be asked: Is IP *the platform* for supporting distributed applications in local, metropolitan, and wide-area networks (WANs)?

There are various approaches to answering this question, and ours will focus on QoS issues. In this context there are other important technological options to consider that are discussed in the following section.

As far as the IP perspective is concerned, all indicators point to the widespread use of this solution in all application areas.

3.1.2 Switched Networks, IP Networks, and Applications

There is general agreement among computer scientists and specialists that cell, frame, or packet switching is a better option when compared to circuit switching or message switching.

As a matter of fact, there is a global trend to employ switching technologies in all networking areas, such as:

- Backbones for local, metropolitan and WANs
- Optical communications
- Supporting multimedia, voice, fax, video-conferencing, mobile, and industrial applications, among others

When analyzing available switching technology alternatives, designers frequently consider certain options:

- Ethernet is an important switching technology option for local areas
- Asynchronous transfer mode (ATM)
- Frame relay
- IP

In the context of our discussion, an important point to understand is how user applications are supported, considering the switching technologies just mentioned.

There are two basic approaches to discuss:

1. The application support is based on level 2 switching.
2. The application support is based on level 3 switching.

In the first approach, ATM, frame relay, or another level 2 switching technology, is used in the backbone. User applications communicate and are effectively distributed across the backbone, directly using the services and features available for the chosen technology. When the chosen technology is ATM, the application would control and use an ATM virtual circuit mesh on top of the backbone. Voice over ATM (VoATM) and voice over frame relay (VoFR) are examples of frequently used applications following the level 2 switching approach.

The favorable point in using this approach is the set of technical advantages of the chosen technology. For instance, ATM will provide a great deal of control to user applications for the characteristics of the communication channels. The application quality of service needs would be mapped to specific services on the ATM backbone. For frame relay, another set of favorable characteristics would be available to applications, such as, controlled bandwidth utilization by committing to the information rate and flow control mechanisms to avoid congestion. These advantages are extensively discussed in many books treating metropolitan and WAN technologies.

The level 3 switching applications support is basically an IP-based approach. Applications are on top of the IP network, and use its services and features through a dual choice of transport services (TCP or user datagram protocol (UDP)) discussed later in this chapter. In this context, the application is fully dependent upon packet switching datagram services and characteristics. This means, in principle, no guarantees for bandwidth, delay, jitter, and others communication channel characteristics. As a matter of fact, there are more technical drawbacks than straight advantages in using native level 3 IP switching support for applications.

The favorable points in using IP-based switching to support end-user applications are the following:

- Massive Internet and corporate IP use results in having IP in almost all computers, or in other words, high availability.
- High availability implies reduced costs as a consequence of good market share.
- Level 2 switching advantages could always be preserved in terms of backbones used by IP switching as a “transport mechanism” for packets.

Level 3 switching is technically far less satisfactory for supporting multimedia, real-time, and other applications that require a better control of network parameters. But the fact is, level 3 switching will prevail as *the alternative* for all applications.

In effect, the indicated advantages are apparently pushing IP switching in such a way that it will have to support all types of applications.

Level 2 switching apparently will not prevail as a general solution for supporting applications in all networking areas (local, metropolitan, and wide area). To understand this trend, we should consider the following:

- Level 2 switching is an excellent solution for backbones, but it does not get to the end user with enough capillarity.
- Level 2 switching is often much more expensive.

So, pragmatically, we will consider level 2 switching technologies like ATM and frame relay as mainly representative for backbone implementations and specific applications support. We will concentrate on IP as the "global end-user platform" in the near future. This being so, a big technical challenge is presented: How to obtain QoS guarantees on top of IP networks? That is the subject discussed next and the technical focus of this chapter.

3.1.3 IP and Quality of Service: Basic Issues

IP was conceived two decades ago to improve a networking scenario of links with very low performance communication and few hosts. This being the case, IP was designed to target simplicity and use a best-effort model, which has properties such as:

- Connectionless
- Routing forwarding decisions taken on a hop-by-hop basis (simplicity)
- No recovery mechanism for errors (let to TCP)
- Support for diverse layer 2 technologies
- Very little control and management over protocol behavior at network nodes

Nowadays, the scenario is quite different, with very high performance communication links being used for router interconnection and many thousands of hosts to interconnect. Their number is not precisely known, but is certainly increasing continuously. In this scenario, new applications with explicit QoS requirements have to run over IP and, as a consequence, QoS has to be addressed somehow by the protocol.

QoS issues are directly addressed in IP version 4 (IPv4) basically by defining the "Type of Service" (ToS) byte in its header (Fig. 3.2).

The ToS byte is defined as follows (Fig. 3.3). Precedence bits (3 bits) specify the priority associated with that particular packet while the remaining bits (bit 3 to 6) specify requirements that could be used for route definition during routing processing. The encoded QoS information is as follows:

IP Precedence (bits 0 to 2):

000 to 111 → priorities

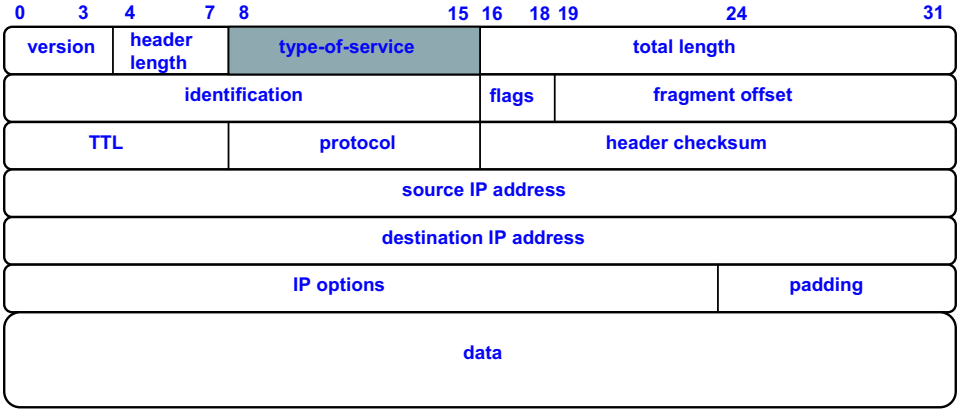

Figure 3.2 IP version 4 packet header.

Type of Service (bits 3 to 6):

All zero—normal service

Bit 3—minimize delay

Bit 4—maximize throughput

Bit 5—maximize reliability

Bit 6—minimize monetary cost

The ToS byte was not effectively used in practice by routers until the recent Internet explosion. In this new scenario, IP Precedence bits have been extensively used to provide a simple priority mechanism for differentiating among packet flows through the network.

IPv4 basic routing processing presents some problems for high-performance routers, which have an impact on the overall network QoS provided. In effect, the IPv4 header was not optimized for gigabit or terabit packet processing and, as such, presents overhead problems. These include:

- Address matches for packet forwarding are processed by using mainly matches to network address prefixes of variable length using a table look-up search method.
- Fragmentation, when necessary, is an important overhead component for routers through the packets path.

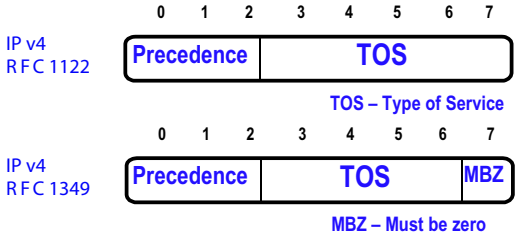

Figure 3.3 ToS in IP header.

- Checksum computation was necessary at every hop due to TTL (time-to-life) manipulation.
- IP options processing implies a variable header length and represents an additional overhead for router packet processing, specially when options are present in the header.

IP version 6 (IPv6), sometimes denoted IP new generation (IPng), is the new IP. IPv6 design addresses the indicated problems of IPv4 and, as such, IPv6 better supports the requirements for high-performance IP forwarding in routers and switches. It is important to mention that IPv6 design addresses many other technical aspects, which are beyond the scope of this chapter. These include:

- A solution to the IP address exhaustion problem, which resulted from explosive Internet growth all around the world
- A security framework at layer 3, targeted at supporting applications
- An overall IP processing optimization (fewer parameters in the header, optional headers, and no fragmentation, among other improvements)
- Other more specific improvements, like autoconfiguration, well-structured address hierarchy, and easy migration path form IPv4, just to mention a few.

The format of the IPv6 header is illustrated in Figure 3.4. IPv6 addresses directly QoS issues by defining specific fields in its header. These include:

- A “flow label”
- A “traffic class”

The flow-label byte may have a general-purpose use, but the basic principle it supports is the concept of flow manipulation. In IPv6, flow identification (flow label) may be used to efficiently manipulate the packet. For instance, a labeled IPv6 packet can be forwarded by

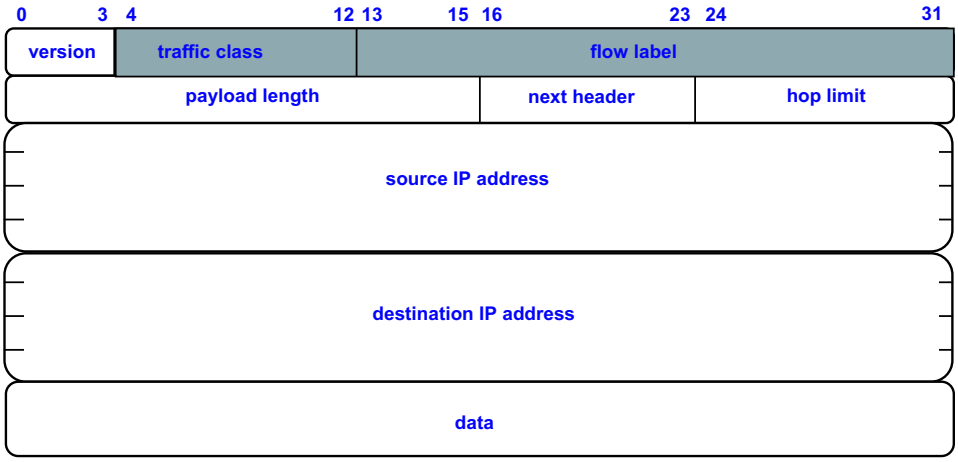

Figure 3.4 IPv6 header format.

using the label, disregarding any other field in the IPv6 header. As we discuss in the following sections, "label identification" may be used by solutions like integrated services architecture (IntServ) and MPLS to implement QoS-aware solutions.

The "traffic class" byte is also oriented to support QoS. It allows applications to specify priorities for the packets they generate. In this sense, IPv6 introduces the principle of "class" which, in practice, corresponds to considering packets from different flows belonging to the same aggregate or grouping. This is another important concept used by QoS solutions like, for instance, the Differentiated Service Architecture (DiffServ).

3.2 QUALITY OF SERVICE

QoS is a fundamental issue for many applications. Once we examine some common definitions about QoS, we will develop a more precise analysis on how to control and manage IP networks in order to obtain the expected operating characteristics or, in other words, the expected QoS.

3.2.1 Quality of Service: Trying a Definition

The definition of QoS depends on the perspective adopted. It is possible to define QoS based on either the user point-of-view, the application point-of-view, or the network and equipment point-of-view, among some of the mostly used perspectives. The following statement is an attempt to provide a more monolithic understanding of QoS, considering the scope of our discussion:

> Quality of service is an application requirement expressed by a set of characterization parameters (typically, a *service-level agreement (SLA)*), which should be provided by the network on an *end-to-end basis,* in order to preserve an adequate application operational behavior and end-user "satisfaction."

SLA is not a unique way to express QoS requirements, but it is certainly the one mostly used. Later in this chapter we will identify other alternatives for expressing QoS requirements.

End-to-end QoS is a challenging requirement for designers. Like a long chain, QoS guarantees are achieved by a set of equipment and services through the network. In the event that one of these equipments or services fail, the entire chain is compromised. Unique and monolithic approaches to provide end-to-end QoS are indeed one of the most promising research and development areas for this subject.

3.2.2 Quality of Service: How to Proceed?

Once we have defined an end-to-end meaning for quality of service, we need to refine the concept by pointing out what technical issues, control procedures, and management actions are necessary to handle it.

Typically, implementing a QoS-ready network requires a rather broad set of techniques, control procedures, and management actions. These include:

- Identifying applications requirements.
- Understanding how network components (routers, switches, hosts, and other devices) behave as far as QoS is concerned.

- Requesting specific QoS in networks and equipment using an appropriate method and formulation.
- Making use of signaling protocols.
- Controlling the router operation for QoS-dependable applications.
- Integrating alternative QoS enforcement control options in complex and heterogeneous networks, such as the Internet, for instance.

Applications should be clearly understood as far as their needs are concerned. The overall network and equipment behavior is fundamental in distributing the need for QoS through the network being used. We must have a clear method of expressing QoS needs in terms of applications requirements and, beyond that, we need to map it on the networks and equipment involved. Signaling protocols are the way in which user requirements are translated in specific network requirements to the equipments adopted. Finally, the router is a very important element for QoS enforcement strategies, since it handles IP packets. Thus, router control and queue management techniques are essential for a successful QoS-ready network. In the following section these issues are briefly discussed.

3.2.3 Application's Taxonomy: Quality of Service

As far as QoS is concerned, applications behave differently and, as such, may be grouped in classes. In this section we identify how they differ and what possible classification could be adopted.

In global terms, applications are classified in two main groups (Fig. 3.5):

1. Adaptable applications
2. Real-time applications

Adaptable applications are also frequently denoted as elastic, satisfactory, and flexible applications. They have the basic characteristic of being able to tolerate fluctuations in network performance and continue to work satisfactorily.

Real-time applications do not tolerate variations in network performance. The basic characteristic of real-time applications is their stringent requirements for QoS parameters, such as packet loss and delay. Variations on the required QoS parameters result in serious degradation of application behavior, and consequently reduce user satisfaction. A good example of real-time application is medical imaging. A system supporting a real-time surgical operation should maintain network performance parameters as required to maintain

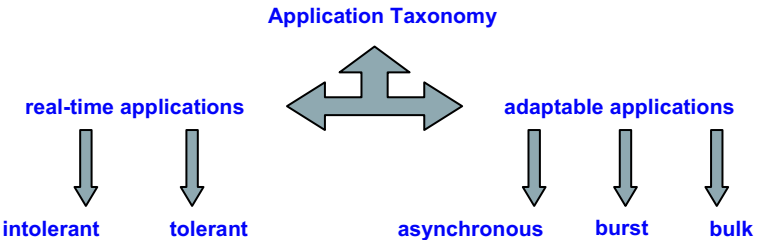

Figure 3.5. Application taxonomy.

perfect images, thus preserving the health and tranquility of the patient. In these applications, QoS enforcement and network performance relying on agreed values is a must.

The adaptable applications mentioned so far can be further refined in another classification as follows:

- Asynchronous application
- Bulk transfer applications
- Burst applications

Asynchronous applications, as their name implies, have a rather asynchronous behavior between their peers. Application peers have to communicate, but there are no strict requirements established. In other words, the requirements are "soft" and easily accomplished as long as the network is operational. The cited requirements concern delay, loss, and jitter in any combination. Thus, the term "asynchronous" can be understood in a broader perspective, meaning flexible applications that are capable of adapting to various network conditions.

A good example of asynchronous applications is email. It is desirable to have email transfers processed as soon as possible, but it is not necessary to enforce bounds to the communication exchange between mail servers. This means that other more demanding applications may borrow network resources from email applications, as long as they can wait for available resources without significant loss. Also, if the network drops packets with email, the application will recover from this situation and preserve the application service.

An interesting aspect of asynchronous applications is how asynchronous they really are. The required response time (which is one possible parameter to identify how asynchronously peers communicate) can vary significantly for different asynchronous applications. As an example, email data transfers, which are TCP-based, can wait for minutes before reacting to any impediment to sending messages. In another situation, hypertext transfer protocol (HTTP) transfers may have their response time dictated by external factors like the maximum waiting time to display a Web page to a customer. Generally, it is assumed that asynchronous applications support network performance variations and that their time limits are typically beyond the minimum the network is able to guarantee.

Burst and bulk applications behave differently in the way they generate data (packets). Burst applications, as the name suggests, produce bursts or peaks of data. The instantaneous data rate may substantially vary from the average data rate for this type of application.

Bulk-transfer applications are more reliable with respect to data generation. In bulk transfers, there is a huge volume of data, and also the instantaneous data rate does not significantly vary from the average data rate for long periods of time. File transfer is one example of bulk application.

The burst or bulk characteristic of the application is important for buffer dimensioning and management. Buffers in routers and switches are a valuable resource that has to be used efficiently. To know beforehand that some supported applications generate data in peaks is fundamental to determining buffer size or, alternatively, to managing and smooth peaks in order to regulate the flow of data in network nodes.

Real-time applications can also be further refined and grouped as follows:

- Real-time tolerant applications
- Real-time intolerant applications

Tolerant applications do have real-time requirements, but can wisely support "fluctuations" of specified QoS parameters without compromising the user's "satisfaction." For instance, Moving Picture Experts Group (MPEG) video applications have the possibility of guaranteeing "graceful degradation" in case network performance suffers fluctuations.

Intolerant real-time applications (also called hard real-time applications) do not accept QoS parameter variability. This means that the requirements are absolutely stringent and violations on parameter agreement (SLA violation) might compromise the "satisfaction" (QoS) of the user.

3.2.4 Quality of Service Parameters

QoS has to be expressed in precise terms. This is an important issue, especially for planners and designers, in order to have a common understanding of the requirements involved.

The QoS requirements for applications are frequently expressed in terms of:

- Bandwidth
- Latency
- Jitter
- Packet loss
- Reliability

The following discussion will consider the definition of these parameters, the factors in a network that affect them, and the possibilities for managing and controlling them.

Figure 3.6 depicts a very simple network comprising two logical LANs interconnected by a router based WAN. There are two applications communicating with QoS requirements on an end-to-end basis. This simple setup will be used to exemplify the QoS parameters just listed, their dependency, and their relationship with network equipment and services.

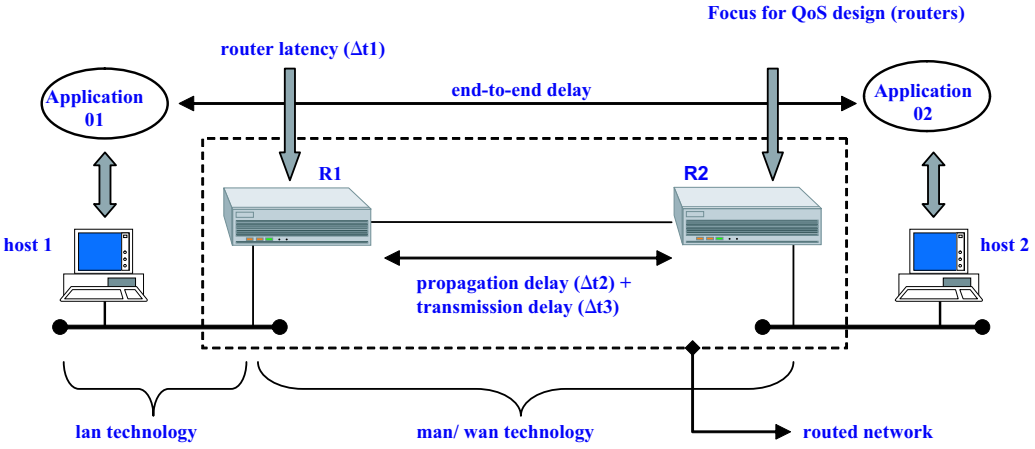

Figure 3.6 QoS parameters in IP networks: an example.

3.2.4.1 Bandwidth Bandwidth is a straightforward parameter representing the amount of *capacity* the network has to guarantee for the application on an end-to-end basis. In simple terms, the bandwidth parameter expresses the *data rate* that the network is supposed to guarantee for the application.

Bandwidth requirements have been identified for the most commonly used applications. Table 3.1 shows typical bandwidths for network applications. Typically, bandwidth requirements are expressed in units of packets per second (PPS) or bits per second. This metric reflects once again the global trend to adopt layer 3 (IP) switching for supporting end-user applications. There are also other parameters that are used to express bandwidth requirements. These include:

- Minimum data rate
- Maximum data rate
- Committed average data rate
- Burst size (expressed in amount of data and/or burst time duration)

In effect, overall bandwidth requirements have to be identified by network managers. Methodologies to identify these requirements for a set of user applications distributed over the network (physical topology dependency) are available in the literature [1]. Subsequently, the network has to be provisioned, as far as bandwidth is concerned. There are various alternative ways to provision a network. These include:

- Contracting services
- Acquiring adequate equipment
- Choosing network technology which matches requirement needs

Bandwidth requirements are not a critical problem for LAN design. This fact is due to technology evolution from shared to switched and their low cost. As an example, single-user ethernet interfaces at 10 Mbps, 100 Mbps, 1 Gbps and, soon, 10 Gbps, are large pipes that give designers enough flexibility and accommodate user needs.

Bandwidth requirements could be a point of concern for WAN and metropolitan area network (MAN) design. The reasons are the following:

Table 3.1 Typical Bandwidth for Some Networking Applications

Application	Bandwidth (Typical)
Voice	6 kbps to 64 kbps
Whiteboard	10 kbps to 100 kbps
Web applications	10 kbps to 500 kbps
File transfer	10 kbps to 1 Mbps
Video (Streaming)	100 kbps to 1 Mbps
Video-conferencing	128 kbps to 1 Mbps
Video MPEG	1 Mbps to 10 Mbps
Medical images	10 Mbps to 100 Mbps
Virtual reality	> 80 Mbps

- Overall communications cost
- Bandwidth limitations on router interconnections
- Availability

Bandwidth is usually an expensive and limited resource for router interconnection. High-speed technologies like synchronous optical networks (SONET), DWDM, and ATM, capable of delivering hundreds of megabits per second, are certainly available, but their use is still precisely adjusted to application needs. In other words, this means that a precise or even an underdimensioned design is mostly a common practice. Also, there is sometimes a rigid budget to maintain, which finally results in using much less bandwidth than necessary. These approaches result in a more affordable global overall cost for network operation, but may compromise QoS requirements. Besides that, availability of high-speed data communication services may be a problem for remote sites.

The concerns about bandwidth limitation for network design just discussed have a direct impact on router operation. In effect, the router might become a bottleneck, since the available bandwidth for their interconnections is not necessarily dimensioned to guarantee the flow of packets from all applications simultaneously at all times. IP processing in routers has to handle packets efficiently to guarantee overall QoS requirements. The issues on router IP processing and router control are discussed further in Section 3.4.

In the near future, it might be possible to have enough bandwidth on all router interconnections for all types of networks (private networks, Internet, etc.). In this desirable and optimum scenario, the point of concern for QoS would be the IP processing capacity of the equipment involved (router, multilayer switch, routing switch, layer 3 switch, or any other type of equipment for switching IP packets).

3.2.4.2 Latency and Delay Latency is a general denomination for the various types of delays experienced by packets when traveling through the network.

Although "latency" and "delay" can be used indistinctly to express slowness in the packet's journey through the network, "latency" is often used in relation to equipment and network services (e.g., switch latency, router latency), while "delay" is often used in data communications (e.g., transmission delay, propagation delay).

Latency and delays can be expressed using different terminology. Figure 3.7 illustrates the terms considered for the scope of the discussion in this text.

Network Latency In general terms, network latency (NL) can be considered as the result of summing up all end-to-end delays produced by network and end-user equipment (hosts). This includes, as an example, delays introduced by network equipment (hubs, LAN switches, and routers), delays introduced by network service

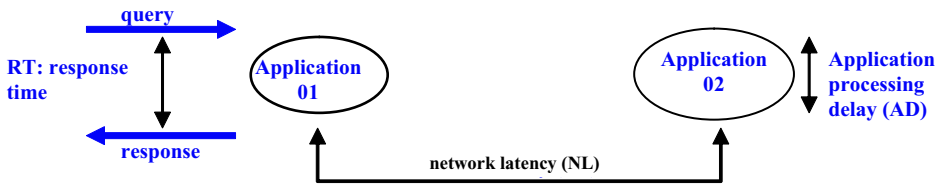

Figure 3.7 Latency and delays.

providers (carriers), delays introduced by signals propagating in a communications medium (fiber, wireless, twisted-pair, and coaxial), and the network stack processing delay at the hosts (Fig. 3.8). The network stack processing delay at the hosts is frequently ignored by designers, but may become an important component when low-performance hosts are using high-performance networks.

Application Processing Delays As the name suggests, the application processing delay (AD) indicates the amount of delay introduced by data processing in the application program. As illustrated in Figure 3.8, the AD may include a protocol component. In effect, it is assumed that the AD incorporates delays introduced by the application code, which, typically, has components such as:

- The application program itself
- The protocol layers reaching up to the application programming interface (API) used

For most TCP/IP networks, this interface is located at layer 4 (TCP/UDP API). The AD is totally dependent on the end-user application program, its computer performance, and local operating system.

End-to-End Delay For the scope of this discussion, end-to-end delay (EE) is equivalent to the previously defined NL, since it incorporates all delays introduced by the network itself and the host's network stack. The EE delay approximately represents the overall time a packet takes from leaving its source to arrival at its destination, except for the processing time in higher layers in the host. This parameter is frequently used to specify QoS requirements for user-to-user applications like, for instance, VoIP.

Response Time Response time (RT) is a typical way of expressing QoS requirements for client/server applications. RT has end-user significance and depends on both round-trip NL and ADs at both communicating parties' ends, as follows:

$$RT = \Sigma\ (NL_{round\text{-}trip}, AD_{source}, AD_{destination})$$

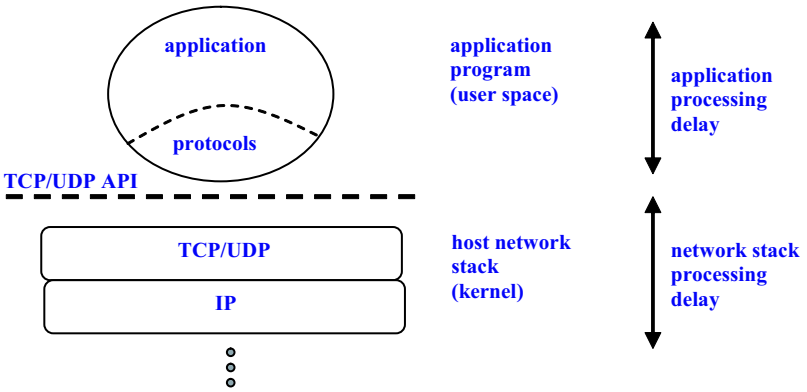

Figure 3.8 Host network stack and application program.

From the application point-of-view, network latency is frequently the main component of the response time. Therefore, let us consider in detail the delay components influencing it. They are the following:

- Propagation delays
- Transmission speed
- Equipment latency
- Communication service latency

3.2.4.2.1 Network Latency Components Here is a brief discussion presenting the main NL components.

Propagation Delay Propagations delays are immutable physical parameters. They represent the time the signals (electrical, optical, or electromagnetic) take to propagate in the media being used in the physical network. For long runs and satellite networks, these delays reach hundreds of milliseconds, and may become critical for certain applications and have to be considered as a NL component. For LANs these values are normally ignored, since they are very small when compared with other delay components.

Transmission Speed The second factor contributing to NL is the transmission speed used by routers and communications equipment. The transmission delay resulting from packet transmission will contribute to the NL.

The transmission speed used in the various network links are defined by project and provisioned for the network by means of any public data communications services carrier or, alternatively, private network. In any event, this is a manageable parameter for QoS provisioning. The points to consider in defining data communication services for IP networks are:

- Router links in local area design are high speed and do not contribute significantly to the overall network latency.
- Router links in long-distance design may become a major source of delay for the overall network latency.

Cost issues and limited availability of services are the main causes of delays in long-distance router links. As an example, 1000-byte packets routed through 1.5-Mbps links, would experience 5.3-ms transmission delay per router link, assuming all links use the same transmission speed. The cumulative effect of transmission delay in multihop router paths can result in long delays. In brief, propagation delay and transmission speed are important design issues for long-distance and metropolitan networks.

Equipment Latency The third network latency component is due to processing in network equipment. Examples of network equipment contributing to increased network delay include:

- Routers
- Hubs and LAN switches

- Firewalls
- Remote-access servers

All network equipment traversed by the packet introduces a delay component typically denoted as *equipment latency* (router latency, LAN switch latency, and so on). This delay is basically due to packet, frame, or higher layer processing within the equipment. In general terms, equipment latency may be considered as the time period between when the packet enters and exits the equipment.

The impact of this delay component on overall delay depends on equipment type and its internal design characteristics. As an example, let us consider a brief discussion of the most important equipment:

- LAN switches normally introduce delays of a few microseconds and, as such, they have very low impact on overall delay. As technology is always in evolution, it is expected that this equipment will not represent a point of concern for QoS.
- Routers, depending on their internal implementation, introduce processing delays that could vary from microseconds to milliseconds. Low-performance routers therefore introduce important delay components, and high-performance routers are equivalent to LAN switch when considering delay components.

Since routers are fundamental equipment for packet processing, their implementation has been the subject of much attention from manufactures. In effect, since QoS emerged as a must for network design, routers have evolved and many techniques have been applied to their implementation. These include:

- Switching techniques (router switch model).
- Integrated multilayer implementation (integrates switching of layer 2 and 3 in a single piece of equipment).
- Hardware integration, especially by using application-specific integrated circuits (ASIC) techniques.
- Multiprocessing implementation.

In general, all equipment should be considered potential delay component elements. QoS design demands careful identification of the equipment's latency in order to guarantee overall delay requirements.

Communication Services Latency Another delay component for applications is communication service latency. Normally, this component plays an important part in overall delay. Communication services are requested by an SLA, and latency is one of the specified QoS parameters. Other parameters typically include bandwidth, packet loss, and availability. As an example, using a frame relay point-to-point communication link between routers R1 and R2 (Figure 3.6) may introduce delays of hundreds of milliseconds for packets. Network design with QoS guarantees should consider public communication services as a potential delay component.

3.2.4.3 Jitter Jitter is another important QoS parameter. In the context of IP, jitter can be defined as the delay variation experienced by packets (packet-delay variation).

Jitter control is important for applications whose correct processing depends on packets being delivered in guaranteed time intervals by the network. As an example, VoIP and fax-over-IP (FoIP) applications do not behave adequately if packets do not arrive at a given rate at the destination for decoding.

As discussed in earlier sections, equipment introduces delay for packets that are variable. Factors influencing the variability of equipment latency are:

- Variable processing time at intermediate equipment (routers, switches, and others)
- Variable public communication services latency
- Congestion
- Other factors related to network operation

Considering the processing and latency variability, it is evident that time processing is different for each incoming packet due to the unpredictable behavior of the distribution of resources in the equipment. Also, it is important to mention that congestion is a critical issue for jitter. Once the network or its equipment reaches a congested state, processing delays increase, contributing to increased jitter.

The Figure 3.9 illustrates the jitter effect for applications. In this simple example, it is important to notice that jitter not only causes variability in packet delivery but also may cause packets to be received out of order.

For TCP-based applications, this last problem is promptly corrected by the TCP protocol. For UDP-based applications (which correspond to a large number of real-time and multimedia applications), there must exist another high-level protocol or procedure to eventually recover from this situation. As an example, real-time protocol (RTP) is used in VoIP applications to help the application in fixing this problem.

Once the jitter can be controlled and kept within defined bounds, there must be an additional mechanism to guarantee that packets will be received in precise time windows for applications. The most common mechanism to eliminate jitter or delay fluctuations is *jitter buffer*. The basic idea is to create a buffer that will keep incoming packets for processing in such a way that the receiver will always have an incoming packet ready for processing. The controlled jitter bounds is the basic parameter used to dimension the required buffer in this approach.

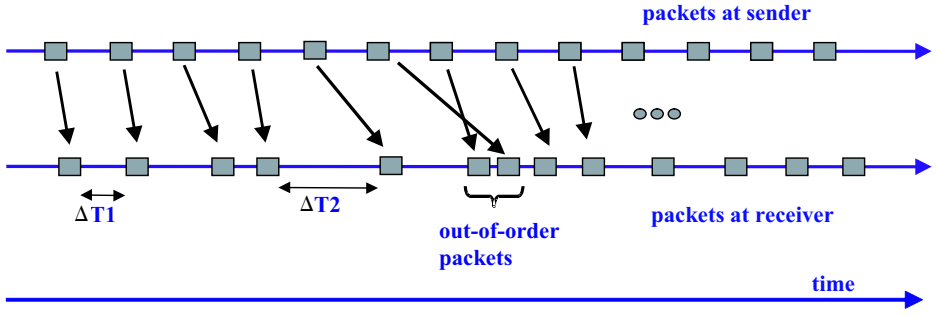

Figure 3.9 Jitter effect for applications.

3.2.4.4 Packet Loss Packet loss occurs in networks mainly due to factors such as:

- Dropping packets at routers
- Transmission errors
- Routing processing decisions

Packets dropped during congestion or due to resource limitation in routers is the main contributing factor for this specific QoS parameter.

Transmission error is a much less important factor contributing to packet loss, since error rates have been considerably reduced by the use of fiber optics and other communication techniques for interfaces and systems. In effect, transmission error rates fall below 10^{-9} for many communication systems.

Packet loss due to routing processing decisions corresponds to the situation in which a packet is routed incorrectly and, because of this, is dropped somewhere in the network. These misrouted-packet possibilities are a less important factor contributing to packet loss.

Packet loss is a very important parameter for certain applications. As an example, VoIP applications will tolerate packet loss up to a certain limit (typically 5%). Beyond that, the recovered voice quality becomes inadequate for user perception.

3.2.4.5 Availability and Reliability Availability and reliability are related parameters considered for QoS. In practice, availability and reliability are parameters typically considered during the network design phase. When considering network operation, these parameters specify directly or indirectly the time constraints for equipment and systems to execute without disruption or failure. For end-user applications, the execution without disruption or failure is dependent on such factors as:

- Network equipment reliability (private networks)
- Public service availability and reliability (when used)

When using public services, SLA should specify availability and/or reliability of parameters required for network operation.

Computer networks are frequently the mission-critical infrastructure for companies and institutions in areas such as electronic commerce, banking, industrial, and retail. In this scenario, availability and reliability are critical issues. As an example, availability requirements above 99.9% are common for Web applications, client/server applications, VoIP applications, and other applications that strongly interact with users.

3.3 QUALITY OF SERVICE: APPROACHES AND INITIATIVES FOR IP NETWORKS

In this section we discuss some basic definitions concerning QoS and we indicate the main existing and proposed alternatives to enforcing QoS requirements in IP networks. The Internet Engineering Task Force (IETF) proposed most of these initiatives, all of which have a very important role in IP network evolution. Besides IETF, other forums, such as the ATM Forum, Frame Relay Forum, WWW, and private companies have contributed, but less intensively, to promoting solutions to QoS enforcement in IP networks.

3.3.1 Quality of Service: Some Basic Definitions

For the purpose of discussing QoS, some basic definitions, which are commonly used with standardized strategies, will be briefly introduced. Also, alternative QoS approaches, like overdimensioning, will be briefly revisited in order to provide a contextual overview of the set of technical alternatives involved.

3.3.1.1 Flow-Based and Class-Based IP Processing In general, quality of service packet processing in routers and switches can use two different approaches:

- Flow-based packet processing
- Class-based packet processing

Before detailing these approaches, we must first give the meaning of the terms "flow" and "class." For the purposes of this chapter, a flow is identified as follows:

- A flow corresponds to a unidirectional (source-to-destination) data stream between applications.
- A flow is typically identified by parameters such as IP addresses (source and/or destination) and port numbers (source and/or destination). Alternatively, a flow can be identified by a label like the flow label used by IPv6 and resource reservation protocol (RSVP) (Fig. 3.10).

Using the concept and identification just described, a flow may correspond, for instance, to packets exchanged by end-user applications. A flow is a fine-grain concept with high capillarity for applications and its associated QoS.

The important concept to retain for now is that the term flow is associated with high capillarity. This means that, eventually, additional parameters may be applied for flow identification. As an example, the ToS parameter in an IP header can be used together with an address and port number to further refine flow's identification.

In our discussion, the term "packets flow" represents the packets being delivered to the network, and that are delivered for one (unicast) or multiple destinations (multicast and broadcast). In this chapter we will use the terms "flow" and "packets flow" interchangeably.

Flows and packets can be identified and grouped together in different ways; for instance, packets originating from a specific machine, a subnetwork, or any other combina-

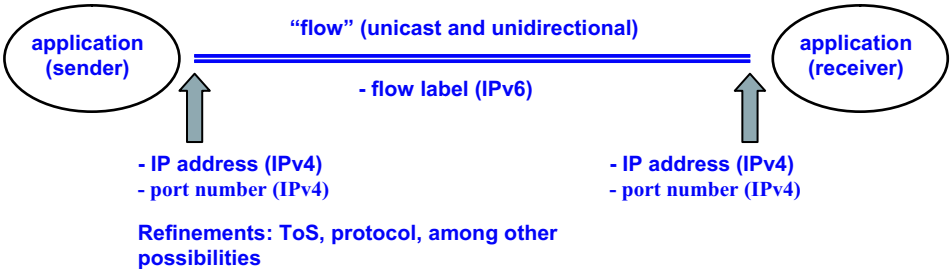

Figure 3.10 A flow between applications.

tions of machines and subnetworks. Also, IP header parameters like addresses, port numbers, and protocols can be used to identify a specific group of packets belonging to an "aggregate" or assigned to a "class." The term "class" or "aggregate" represents a group of packets with some common identification (IP prefix, from the same machine, etc.), which, in effect, corresponds to two or more flows.

For the purposes of this chapter, aggregate and class are terms representing the same basic idea. The term class is often used to indicate a group to which the packet belongs, and the term aggregate typically represents a group of packets flowing through the network that belong to the given class.

For now, let us discuss the main characteristics when flow-based and class-based packet processing is used in strategies to enforce QoS.

In flow-based packet processing, the processing is done at the flow level. In other words, packets belonging to a specific pair of applications (sender/receiver) are processed to enforce the expected QoS.

The main advantages of flow-based packet processing are:

- High capillarity, since processing can be tuned to end-users individually.
- Good resource distribution, since applications can be tuned individually.

The main drawbacks of flow-based packet processing are:

- Flow identification with high capillarity typically requires additional processing.
- Does not scale well, since flow-based packet processing is subject to "state explosion" in big networks with many users.
- The amount of per-flow information to be maintained by the network nodes may also become a scaling problem.

In class-based packet processing, the processing is done for an aggregate or class of packets. Flows are not considered individually anymore.

The main processing advantages of a class-based packet are:

- Class-based processing scales well with many users since, for instance, state information is kept to a minimum.
- Flow aggregates reduce the analysis of packet header parameters by routers, and consequently reduces overhead.
- Processing by class eases operation management and price policy implementation.

The main processing drawback of a class-based packet is related to the fact that individual users cannot be tuned adequately as far as their specific requirements are concerned.

3.3.1.2 Token Bucket IP quality of service control approaches like the IntServ and the differentiated service architecture (DiffServ) use, typically, a token-bucket scheme. We will therefore introduce the basic concepts related to this scheme.

QoS strategies need to keep track (monitoring, controlling, etc.) of the characteristics of different packet streams, such as average data rate, peak data rate, and bursts. Identification of these characteristics is necessary for the network, because the traffic from most

applications is quite variable and, besides that, the traffic does not necessarily follow predefined service contracts due to the behavior of intrinsic applications. Networks thus need to have a scheme to limit traffic at network borders, to shape bursts, and to identify out-of-profile traffic, among other needs.

The token-bucket scheme can be used to address some of the listed network needs. In its most basic operation, the token-bucket principle is as follows (Fig. 3.11):

- Tokens representing "credit" are generated at contracted rate "r."
- Tokens/credits accumulate in the bucket till the limit "b" (the bucket size).
- Packets require credit to enter the network.
- Packets arriving at average rate "p" get access to the network with credit that is available at the bucket.
- Packets experience a predefined action, in case the accumulated credit is insufficient.

This very simple scheme can be used to create some necessary functionality in networks like policy modules, packet markers, and shapers.

As illustrated in Figure 3.12, a token-bucket filter implements a "policy module" that limits out-of-bound traffic by dropping packets that do not obtain enough credit when arriving at network entry point. The operation is as follows:

- Tokens representing credits are generated at contracted rate "r," which is supposed to be the average contracted rate for the incoming packet stream.
- Tokens/credits accumulate in the bucket until the limit "b" (the bucket size), which allows the instantaneous packet input rate to vary (input data rate fluctuation).
- Arriving packets require credit proportional to their size to get access to the network.
- Any arriving packet is dropped, in case the accumulated credit in the bucket is insufficient at arrival time.

The token-bucket scheme has advantages. For example, it can be used by the network to enforce a traffic contract, by identifying conforming and nonconforming packets. Also,

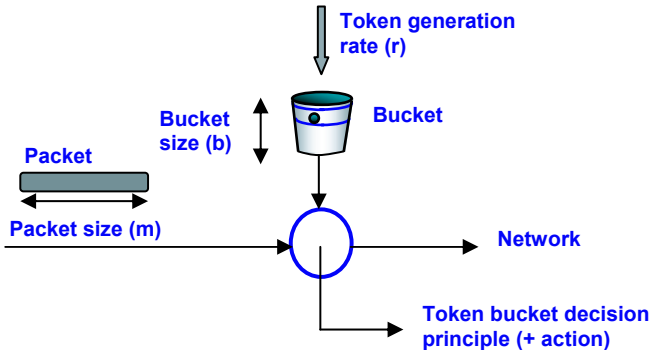

Figure 3.11 Token-bucket principle.

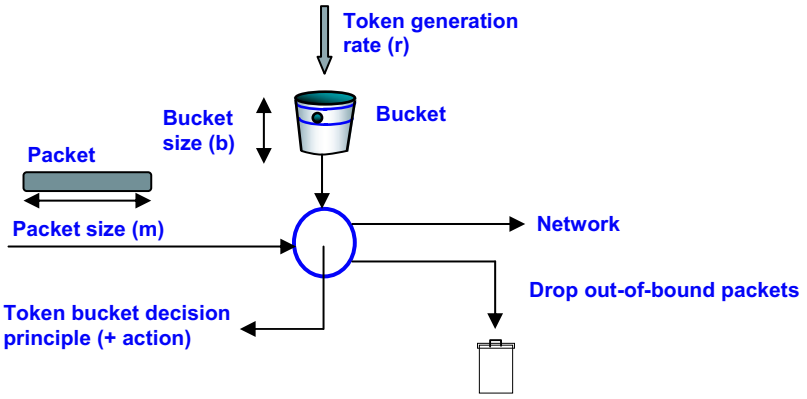

Figure 3.12 Token-bucket as a policy module.

the token-bucket scheme provides a boundary on the average rate and limits the maximum amount of traffic during any period of time. The last mentioned advantage is important because it adequately limits the buffer size used in devices like routers and switches.

Slight variations of the preceding operation and the concatenation of parameterized token-buckets produce shapers, markers, limiters, and other modules, which are essential to QoS control and management.

3.3.2 IP Quality of Service: How to Handle It? Having examined some common definitions relating to IP QoS, we will now develop a more precise view of how to enforce QoS requirements in IP networks. The most common alternatives for enforcing QoS parameters in IP networks are the following:

- Overdimensioning
- Queue and congestion management in routers and switches
- Integrated services architecture
- Differentiated services architecture
- Multiprotocol label switching
- Proprietary and hybrid solutions

Each one of these solutions has its own set of characteristics that we will describe and discuss in the remainder of this chapter.

3.3.3 Do We Need Quality of Service Control?

Networking technologies have evolved significantly in the last few years. This evolution becomes apparent when we consider, for instance, the data rates supported by the actual and under-development technologies. Following is a brief review of frequently used technologies:

- Ethernet has evolved from a 10-Mbps shared-hub solution to a switched solution (eventually dedicated) ranging from 100 Mbps, 1 Gbps, and beyond to 10 Gpbs.

- ATM supports data rates that typically range from 1.5 Mbps to 622 Mbps.
- SONET and synchronous digital hierarchy (SDH) can, typically, deliver data rates ranging from 52 Mbps to 622 Mbps and beyond to 2.4 Gbps.
- DWDM is another high-speed technology with data rates that can achieve hundreds of Gbps and beyond to Tbps (terabits per second, or 10^{12}).

Besides the increase in supported data rates, another important aspect of networking technology's evolution is that the cost per bit has been considerably reduced over the last few years. In principle, this aspect stimulates the widespread use of the technologies.

This being so, a fundamental question can be asked: Do we really need do deploy any form of *QoS control* in networks?

The point is: with high-speed technologies available, we could just provide as much bandwidth as necessary for applications. This approach, called overprovisioning or overdimensioning, is discussed next.

3.3.3.1 Overdimensioning Approach. Strictly speaking, overdimensioning is not a technique for QoS control. In effect, the basic idea is to provide as much bandwidth as necessary for any application. Since technologies have evolved and are capable of delivering hundreds and even thousands of megabits per second, overdimensioning might be applicable in some circumstances.

The context for overdimensioning utilization is mainly the private and local-area networking. In this context, it is a possible alternative, considering that:

- Users are, in principle, well known (number, traffic characteristics).
- The global cost for installing the network is, apart from network operations and management maintenance costs, mainly the measurable initial investment.

For wide and metropolitan area networking, the use of such big pipes typically implies higher monthly costs for public services, which prevents the use of this approach for network provision.

The possibility of using overdimensioning may also be discussed using another perspective. The question is: In order to guarantee overdimensioned output links, is it enough for applications to behave as expected?

Intuitively, the straightforward answer is yes, since we assume that large output pipes will guarantee that routed or switched packets and cells will be delivered very quickly. In routers and switches, this behavior guarantees that output queues will be nearly empty and, thus, delays are reduced and the jitter is minimized in the network.

The preceding supposition raises two additional basic questions:

- Do the networking devices (routers and switches) have enough processing capacity so that we can assume minimal latency and processing time?
- Can we be sure that users and applications will not evolve in parallel to consume the "pretended" overdimensioned resources?

For the first question it is necessary to understand the existing limitations for network devices. Focusing on routers, the fact is that processing capacity and network design features have, for the moment, eliminated eventual internal-capacity bottlenecks in these devices. High throughput routers and level 3 switches, for instance, can deliver thousand of

packets with minimal latency (on the order of microseconds) and, as such, do not yet represent a serious concern for QoS. Also, virtual networking and careful design can overcome excessive traffic concentration at some network points, which would create a potential bottleneck. As such, the actual router limitation is the output interface capacity, often limited in metropolitan and wide-area design by cost or availability. Beside this, there is no clear evidence that "free bandwidth" will emerge as a network design scenario in the near future.

Analysis of the second point may be better tackled by looking back at the network evolution. The fact is that historically user-application demands have always risen to meet network technology's evolution. In other words, network evolution has been followed by new applications with tougher requirements concerning data rate, delays, and jitter. In this evolving scenario, it may happen that the overdimensioning approach does not work at all and the network designer will have to adopt QoS strategies to adjust the network to attain user demands.

In conclusion, the fundamental resource is the output link rate. Therefore, network designers and managers will need algorithms, techniques, and end-to-end strategies to help them in controlling requirements and delivering QoS for end-user applications.

3.4 PACKET CONDITIONING, QUEUE SCHEDULING, AND CONGESTION CONTROL IN ROUTERS

As stated earlier, routers are a point of concern for QoS since they could become potential bottlenecks for packet flow. In effect, for high packet volumes flowing through routers, low-speed links used in interconnections and low processing capacity, among other factors, may cause packets to slow down or even to be dropped at routers.

As illustrated in Figure 3.13, basic router operation can be described in very simple terms:

- Packets are picked up at input queues, where they arrive asynchronously or, eventually, synchronously.
- Packets are processed according to basic IP routing operation (TTL control, address handling, and so on).
- Packets are switched to output queues for transmission.

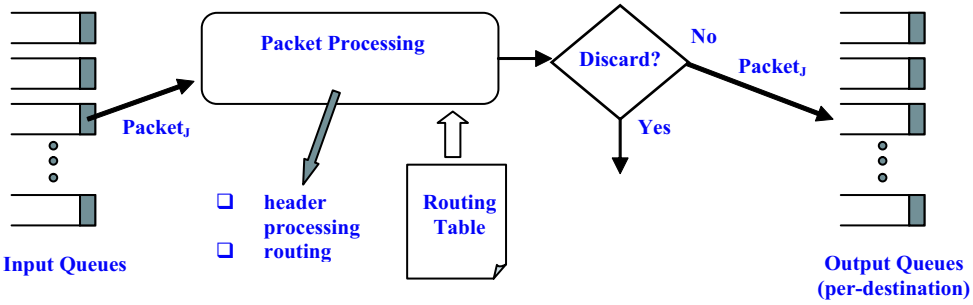

Figure 3.13 Basic router operation.

This very simple view of the router operation has a set of potential problems for QoS. Here is a short summary of some of the problems encountered in routers:

- Packet latency in routers is dependent, for instance, on router processing capacity and transmission speed at output interfaces. Low-capacity routers will lead to greater router latency since packets will take longer to be internally processed. Also, low-speed interfaces will lead to bigger router latency because packets wait longer to be transmitted on output queues.
- Packet loss has a great deal of dependency on router capacity, queue memory size, and transmission speed at output interfaces. Packets may be lost by combining situations, for instance, like queue overflow due to low transmissions speed or queue overflow due to peaks of input traffic.
- Jitter is another QoS parameter influenced by router operation. Since there is no prior guarantee that packets from the same sources will be handled "homogeneously" (given the same amount of router capacity, processed at same time, and so on), internal packet delays in routers tend to be quite variable, thus increasing the jitter.

Overall router operation can be managed by applying a set of basic control methods to packets and queues such as:

- Packet conditioning
- Queue scheduling
- Congestion control

In the remainder of this section we will elaborate on each of these control methods in more detail and review their application to routers.

3.4.1 Packet Conditioning

In general terms, packet conditioning corresponds to the set of actions carried out on packets in an attempt to establish and preserve their relative importance or priority in relation to any other packet in the network.

Packet conditioning can be accomplished by the following techniques:

- Packet prioritization
- Packet classification and markup
- Traffic shaping
- Policing

It is important to observe that these functions are, in practice, typically used in conjunction. Depending on the QoS strategy used, these functionalities may be combined to achieve the expected result. So, classification may precede traffic shaping or prioritization may result from traffic shaping results. This being so, the following discussion will focus on the description of basic functionalities, and their interoperation will be elaborated later when we discuss the scheduling algorithms and QoS strategies.

In an attempt to give some insight on how these functionalities might interoperate, Figure 3.14 shows a typical router implementation. Packet prioritization is one of the most

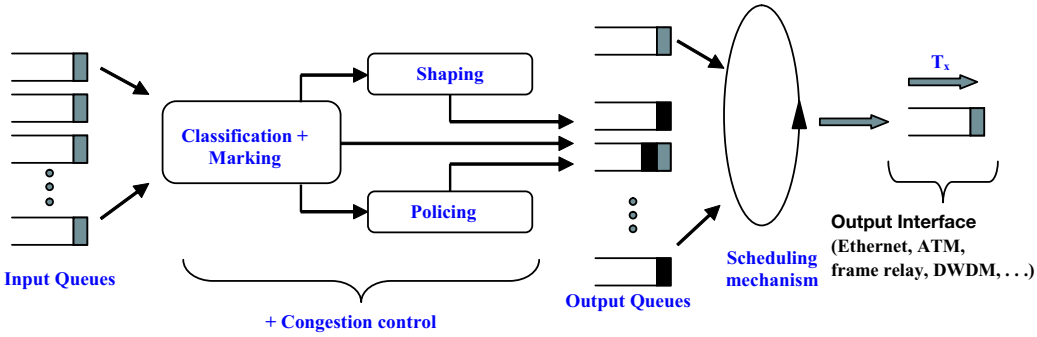

Figure 3.14 Packet conditioning, scheduling, and congestion control in typical interoperation.

frequently used and straightforward packet conditioning techniques. Packets are "marked" as having a specific priority and, as an example, will receive network resources like memory and bandwidth depending on its priority.

In IP routers, the scheduling algorithms used to decide which packet would be serviced next typically employ packet prioritization. Alternatively, packet prioritization also may be used to decide which packet will be dropped during congestion. Figure 3.15 illustrates QoS and routing functionalities using priority-based decisions.

Packet classification and markup is another fundamental and very frequently used conditioning technique. The idea is to investigate packet content (IP header, TCP/UDP header, and so on) in order to identify the flow or application to which that packet belongs, and therefore infer the packet's characteristics. As a very simple example, this technique allows a router to know that for the two packets waiting for service, one carries synchronous time-sensitive traffic and the other carries asynchronous traffic. This information is valuable to various aspects of router operation.

Figure 3.16 illustrates the basic principles for classification and markup functions. In routers, packet classifications typically occur when moving packets from input queues to output queues. The markup technique, normally associated with packet classification, allows some sort of relative identification of packets within the router, or within a network of routers and switches. There are various approaches to marking packets (labeling pack-

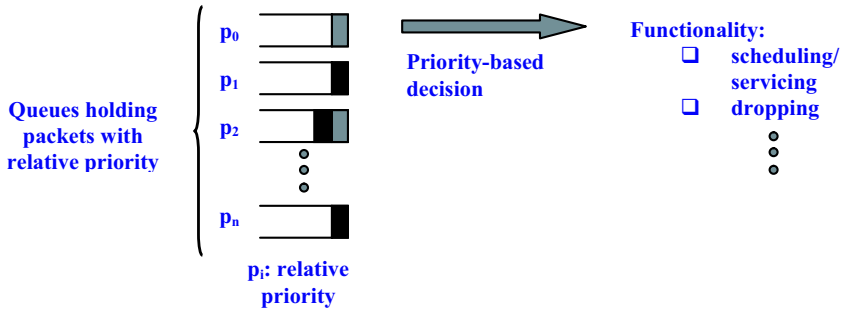

Figure 3.15 Typical functionalities using priority-based decisions.

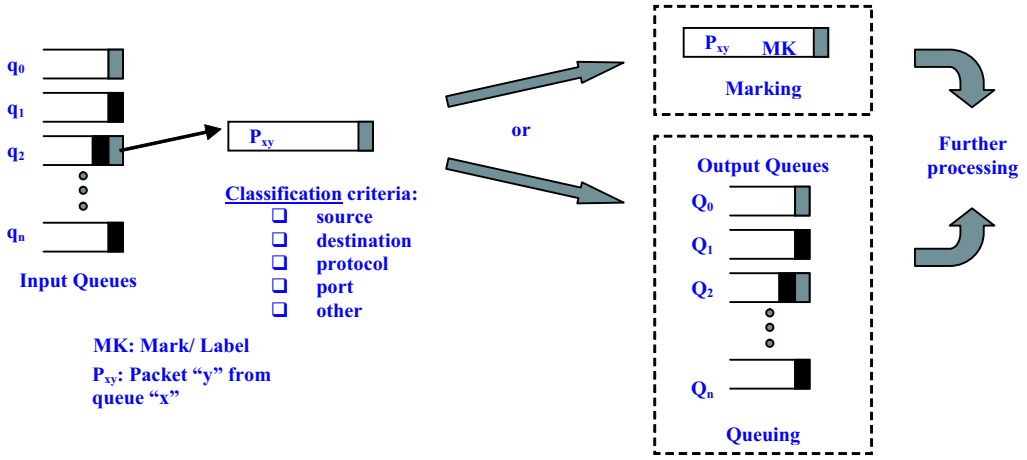

Figure 3.16 Classification and markup functions.

ets) and distributing labeling information (RSVP, MPLS, etc.) that will be further discussed later in this chapter.

Packet classification and packet markup are separate functions, but intrinsically associated. It means that for markup, some classification has to be executed, but classification does not necessarily apply in marking or labeling packets. The way these functions interoperate depends on the QoS strategy used.

Traffic shaping is a very important technique necessary in many scheduling algorithms in routers. When applied to a classified stream or flow of packets, traffic shaping allows, among other possibilities, the control of packet bursts. The incoming traffic is "shaped" by defining upper bounds, for instance, on the packet's average rate and burst. The token-bucket filter is a typical example of shaper functionality implementation.

In routers, traffic shaping is typically used to limit incoming traffic to predetermined profiles. Alternatively, traffic shaping also can be used at network borders to deliver packets according to predetermined flow characteristics. In effect, predetermined short- and long-term traffic behavior is an essential condition for many algorithms used in routers.

Traffic-shaping techniques also can be used, for instance, in conjunction with a monitoring function to make decisions about dropping packets. In this particular situation, the bursts of traffic are detected. Depending on the accumulated volume of traffic generated, new out-of-bound packets might be elected to be discarded or not.

Before discussing policing functionality in the router context, it is important to understand the general meaning of the term "policy." Generally speaking, policy is a set of rules associated with services (Fig. 3.17). The principle is such that these rules define the criteria for obtaining related services. In other words, policy describes the set of conditions that must be satisfied before actions related to the service can be taken. Policy is applied to many aspects of computers, such as security, management, and administrative issues.

In the specific context of router operation, typical examples of possible policy rules are:

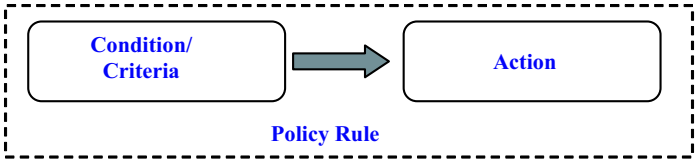

Figure 3.17 Policy rule.

- Mark packets out-of-bounds with respect to negotiated SLA for dropping.
- Drop packets over the agreed medium rate.
- Lower the priority of packets over the agreed burst size.

In the first example of a policy rule the condition is: "packets are out-of-bounds with respect to negotiated SLA"; while the action is: "mark packets for dropping".

Policy management is a broad technical issue, and for further reading a visit to the IETF Policy Framework Working Group Web site (www.ietf.org) is recommended.

3.4.2 Queue Scheduling

Queue scheduling corresponds to the set of algorithms or mechanisms used to control the way packets are sent from an input to output queue in routers. The scheduler analyzes packets from queues for eligibility. The best eligible packet per queue and per algorithm is scheduled and serviced.

In IP routers, the queue scheduling algorithms are targeted to optimize IP operation and improve QoS parameters.

The main goals for queue scheduling mechanisms used in IP routers are the following:

- To share bandwidth fairly.
- To guarantee QoS parameters like bandwidth and delay for certain types of applications, for instance, time-sensitive ones.
- To reduce jitter.
- To prevent bandwidth starvation among IP users.

There are many alternatives for scheduling IP packets on router queues, most of which have proprietary implementations. Some of the most frequently used algorithms found in IP routers are:

- First-in first-out (FIFO)
- Round-robin (RR) and weighted round-robin queuing (WRR)
- Priority queuing (PQ)
- Weighted-fair queuing (WFQ).

3.4.2.1 First-In First-Out First-in first-out (FIFO), also called first-come first-served algorithm (FCFS), is the simplest and most typical default solution found in routers. As the name suggests, packets are switched from input to output queues in order

of arrival independently of their size, type, contents, or any other packet characteristics (Fig. 3.18).

Considering IP routing, the FIFO algorithm's main advantage is simplicity and that it behaves well in the absence of network congestion. It consumes far less processing than any subsequently discussed algorithms, and thus can be used with very high-speed interfaces.

With heavy traffic or during congestion, the FIFO algorithm creates a potential problem for QoS enforcement in routers.

First, since it does not distinguish packet lengths, small packets may experience long delays while long packets are transmitted. Assuming the short packets are generated by a real-time or QoS-demanding application, this characteristic contributes to increased router blindness to critical applications.

Second, packet dropping is executed at random, as far as the type of packet or type of packet content is concerned. Usually, routers using FIFO execute tail drop (arbitrarily). This characteristic is very undesirable or even unacceptable for most critical applications since it increases packet loss without criteria. Besides that, in this operation, packets with low priority may pass through while critical packets are being dropped out.

In brief, the disadvantages of FIFO include unfairness and the impossibility of guaranteeing bandwidth or delay parameters for QoS enforcement over routers.

3.4.2.2 Priority Queuing Priority queuing (PQ) is a very common solution in routers. In PQ, packets are typically sent to n queues, which are assigned relative priorities (0 to $n - 1$). These priorities set a relative ordering (high to low) for servicing the queues for a given interface. In PQ, packets in higher priority queues are serviced first. Besides that, packets in queue k are serviced only if queues 0 to $K - 1$ are empty. Figure 3.19 illustrates PQ operation.

Considering IP routing, the main advantages of PQ are high throughput, lower delay, and higher bandwidth to packets in higher priority queues.

Queues in PQ may experience resource starvation, which is one of its most serious problems. Since there is an order for servicing queues, it may happen that low-priority

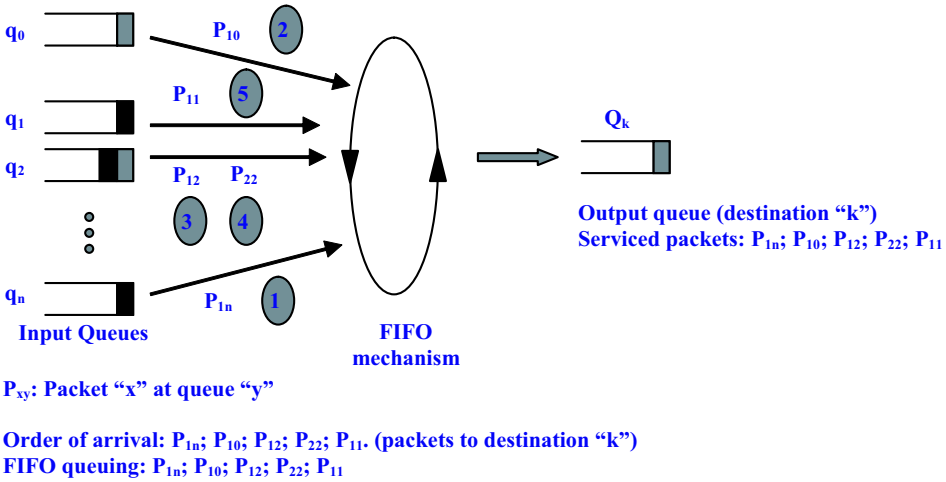

Figure 3.18 Basic FIFO operation.

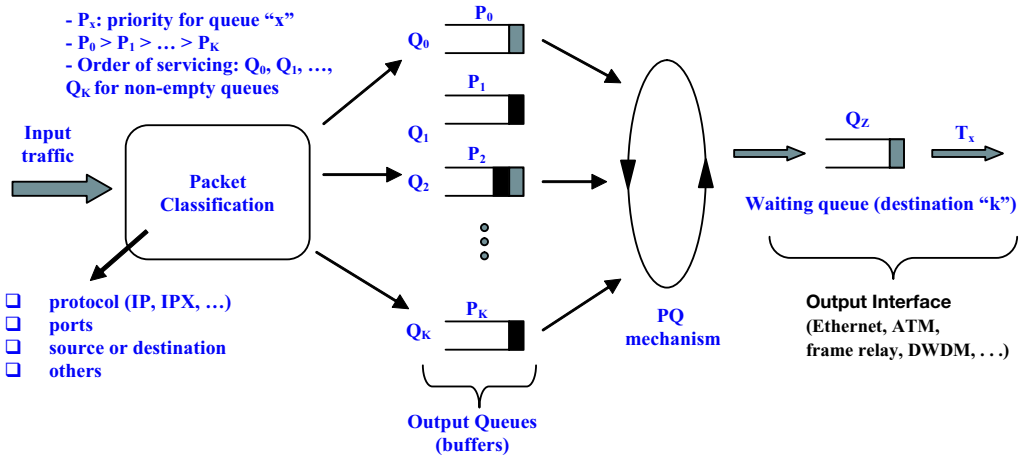

Figure 3.19 Priority queuing operation.

queues starve for resources, which happens when any highest-priority queue remains occupied. There is, therefore, a compromise between the traffic receiving the lowest possible delay and the low-priority traffic, which should receive minimum resources. Often, the use of PQ assumes that admission policies and control are used to limit the amount of high-priority traffic in high-priority queues, somehow regulating the starvation problem.

In PQ, several levels of granularity for traffic classification are possible. For instance, traffic may be prioritized by the type of packet (IP, IPX, or other), or by the type of application (VoIP, HTTP, FTP, etc.). In the first case, IP packets may be delivered before IPX packets and, in the second case, packets carrying VoIP application data identified by the RTP protocol may go ahead of packets carrying HTTP application data. Higher levels of granularity have the disadvantage of increasing the packet processing overhead for classification and queuing, and, as such, have often to be limited in practical PQ applications. One of the resulting considerations of the previous discussion is that PQ does not scale well for higher levels of granularity.

In general terms, PQ can be considered to be the most "primitive" method to differentiate traffic for scheduling. For this reason, more "refined" algorithms exist and are the focus of the following discussion.

3.4.2.3 Round-Robin and Weighted Round-Robin Queuing In round-robin (RR), packets are classified and sent to m queues. Classification, as seen previously, may be performed by type of packet, type of application or any other packet characteristic. The queues in the basic RR algorithm are serviced in order (0 to $m - 1$), one packet at a time (Fig. 3.20).

RR solves the starvation problem, since all queues are periodically served by the RR discipline, but introduces new problems.

First, RR does not consider packet size, and is consequently unfair. In some cases, small critical packets may wait for long periods in queues while large noncritical packets are serviced. Second, RR cannot provide guarantees for bandwidth or delays.

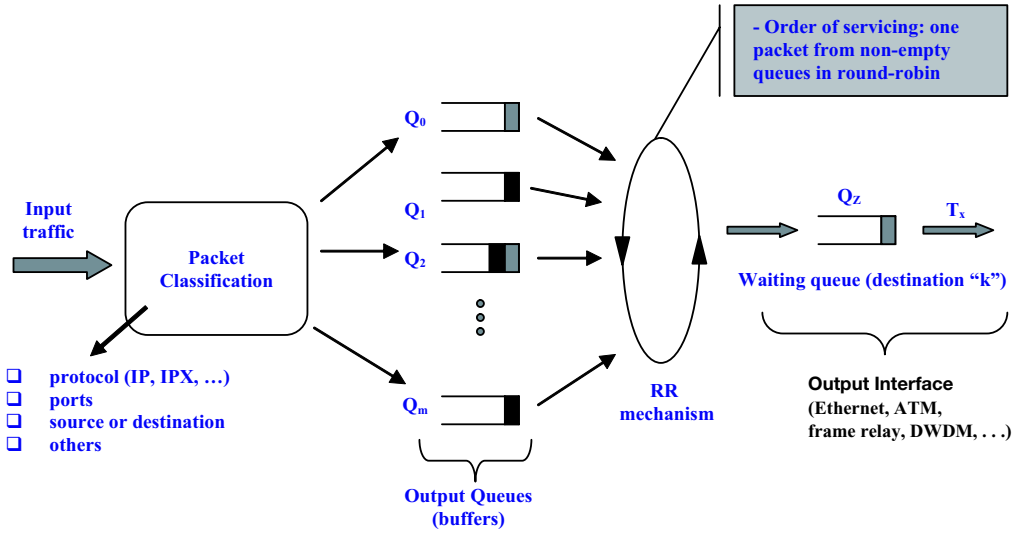

Figure 3.20 Round-robin operation.

Weighted round-robin (WRR) is one possible variation of the RR discipline. In WRR, more than one packet can be serviced in turn for a particular queue. As in the RR algorithm, if packet sizes are different, the algorithm deviates from a strict relative resource allocation strategy and is therefore unfair.

3.4.2.4 Weighted-Fair Queuing Weighted-fair queuing (WFQ) is another algorithm used in routers in an attempt to provide fairness, to deliver predictable behavior to packet flow, and to avoid the resource starvation problem.

WFQ is an approximation of the generalized processor sharing (GPS) algorithm. GPS is an algorithm that provides fairness by visiting each active (nonempty) queue in turn and servicing it bit by bit (infinitesimal service). The basic problem with GPS is that it does not have a feasible implementation, since it can serve only whole packets.

In WFQ, packets are served in order of their "finish times." To compute finish times, the WFQ operation maintains two variables (assuming weights are 1 for all queues):

- The current round number
- The finish time (packet state information maintained per queue)

The round number, as the name suggests, corresponds to the number of rounds of service completed by the round-robin scheduler, supposing that the algorithm serves one bit from each active queue. The round number reflects the amount of traffic served for the active queues. The round number may be fractional, where fractions represent partially completed rounds.

To show the computation of the "finish time" (F_i) at queue, i, suppose a packet of length L, arrives to an empty queue when the round number is R. The finish time for this packet corresponds to the time of transmitting its last bit ($F_i = R + L$). This packet will be served by WFQ discipline only when its finish time is the smallest among the queues.

When a packet arrives at an active queue (not empty), its finish time is equal to the previous packet finish time plus its size.

The following expression computes the finish time for a packet in a given queue, *i*:

$$F_i(x, t) = \max \{F_i(x - 1, t), R(t)\} + S_i(x, t)$$

where

$F_i(x, t)$ = finish time for packet "x" on queue "i" at arrival time "t";
$R(t)$ = round number at packet arrival time "t";
$S_i(x, t)$ = Size of packet x arriving on queue i at time t.

The round number is updated on the arrival or departure of each packet. Packet departure resulting in empty queues increases the round rate for servicing. On packet departure, the queue is considered inactive when the largest finish time in the queue is smaller than the round number. On packet arrival, the round number is recalculated and the finish time is computed.

When weights are allocated to queues, the computation of the finish time takes the weighted size packets into account as follows:

$$F_i(x, t) = \max \{F_i(x - 1, t), R(t)\} + S_i(x, t)/w(i)$$

where $w(i)$: weight on queue i. The larger the weight, the smaller the resulting finish time for a given packet size, which in practice means that more bandwidth is effectively allocated for that queue.

In brief, WFQ operation can be resumed as follows:

- On packet arrival, classification allocates the packet to a queue.
- The round number is recomputed.
- The packet finish time is computed (relative to either its previous packet on queue or the round number).
- Service the packet with the smaller finish time.

On packet departure, the WFQ algorithm services the packet with the lowest finish time. When queues are empty the algorithm goes idle.

Figure 3.21 illustrates the basic WFQ operation. In brief, the general characteristics of WFQ are fair in that:

- Traffic with small sized packets are not compromised and have an effective priority.
- Traffic of large packets does not hog bandwidth.
- The use of weights for queues allows an intrinsic allocation of more resources (bandwidth) for time sensitive traffic.

WFQ also provides performance guarantees:

- Bandwidth bound
- End-to-end delay bound

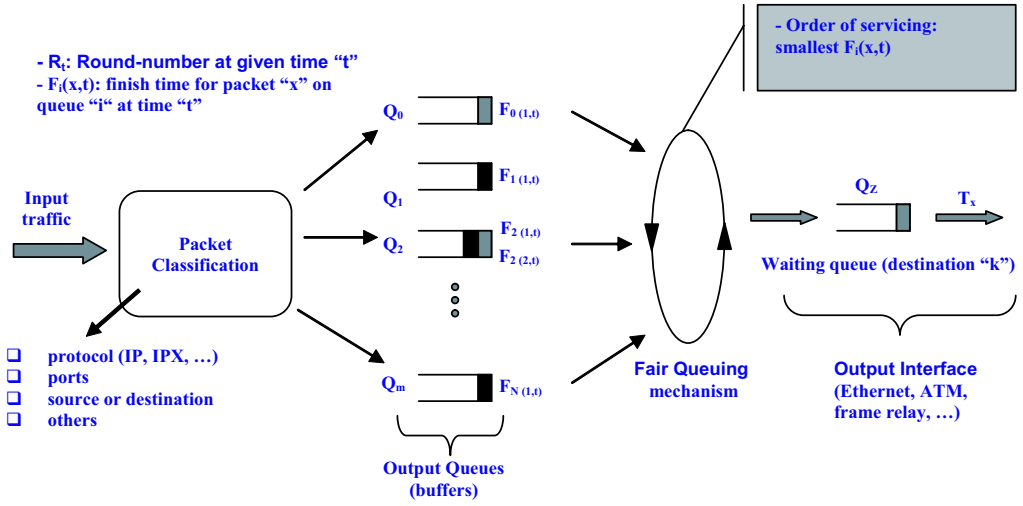

Figure 3.21 The WFQ operation.

Bandwidth bound is achieved by rating link capacity, thus allowing queues to receive minimum amounts of the available bandwidth. End-to-end delay bound is achieved if queues are regulated, for instance, by a token-bucket filter; otherwise, too many packets will arrive and will be stuck in queues.

3.4.2.5 Other Scheduling Approaches There are additional technical alternatives for queue scheduling, such as:

- Virtual clock
- Deficit round-robin (DRR)
- Distributed weighted-fair queuing (DWFQ)
- Class-based queuing (CBQ).

These alternatives are either variations of the basic queue scheduling algorithms previously discussed, proprietary implementations of generic solutions, or new ideas, which may be implemented in routers. For further details of these solutions, the references at the end of this chapter should be consulted.

3.4.3 Congestion Control

Congestion control is the general term for the set of mechanisms used to both prevent and eliminate congestion in networks.

Congestion occurs because packets can go "from any input queue, to any output queue," at any moment. In other words, for an unpredictable period, T, up to $(n - 1)$ of n possible input queues could be directed to a single output queue. Since bandwidth is normally limited in packet-switched networks and IP routers, in particular, congestion can occur at any time.

Congestion control techniques are installed capabilities available on routers and switches that have the ability to control flow and deal with excessive traffic during congestion periods.

One of the first basic points concerning congestion is the perception of its effects on data flows. Congestion may influence basic QoS parameters:

- Congestion can increases packet loss
- Congestion can increases delay and jitter

During congestion periods, packets are dropped out, causing packet loss to increase for the corresponding data flows. Depending on the level of congestion, all flows may experience packet losses.

Transport protocols (TCP or UDP) are also influenced by congestion. In TCP, lost packets are interpreted as signaling congestion and cause resynchronization among TCP peers. In effect, lost packets trigger the TCP slow-start algorithm and other procedures to adapt TCP flow to the underlying network throughput capacity. TCP slow-start triggering results in delay increase and jitter variation.

In UDP, delay and jitter variation result mainly from either additional packet processing and overload at output queues or resynchronization procedures, whenever they are used at the application level. Longer output queues result in longer delay for packets waiting on these queues.

The congestion control techniques differ in the threshold the moment they start to deal with the congestion problem. It is worthwhile to distinguish between congestion management and congestion avoidance techniques. *Congestion management* deals with techniques and strategies used to eliminate congestion once it begins. Congestion management comprises reactive techniques, which typically try to manage congestion by reducing data flow intensities over the network. *Congestion avoidance* comprises techniques and strategies that try to prevent congestion from occurring. These strategies are proactive techniques that basically try to avoid the nasty effects of congestion on data flows.

Another important aspect of congestion is the identification or detection of a congestion situation or congestion period. The point is: How can congestion be identified, for instance, in a router?

Typically, an excessive queue length can signal a congestion situation. In other words, the queue length is normally the most usual parameter used to identify congestion. The actual optimal queue length depends on a number of factors, such as traffic type (bursty, adaptative, etc.).

The basic approaches used to control congestion are the following:

- Tail drop
- Random early detection (RED)
- Explicit congestion notification (ECN)

Tail drop is a straightforward mechanism in which incoming packets are dropped if the output queue for the packet is full. RED is a general technique in which packets are discarded with a certain probability in order to prevent congestion [2]. ECN is another technique proposed for routers to modify a congestion situation to ECN-capable end systems.

3.4.3.1 Tail Drop As previously indicated in this section, tail drop is a straightforward mechanism in which incoming packets are dropped if the output queue overflows. Tail drop is a very simple procedure, but it has disadvantages, among which are the unfair queue utilization with FIFO queue management, and TCP synchronization and recovery problems.

The main disadvantages with TCP are the global synchronization problem [3] and the recovery procedure for multiple packet losses in a single window. If many TCP hosts are sending packets simultaneously during a congestion period, tail drop will discard packets, and, consequently, TCP hosts will stop sending them in an attempt at synchronization using the slow-start algorithm. Also, multiple packet drops in the same TCP window will cause long delays on recovering. The unfair queue utilization problem occurs when a small set of flows "monopolize" the link by generating packets at a higher rate (lockout). In this condition, packets from "well behaved" flows will have a greater probability of being discarded.

3.4.3.2 Random Early Detection Random early detection (RED) is a recommended active queue management technique that attempts to prevent highly congested situations. The basic idea is to discard packets early and, in doing so, to avoid the output queues to become completely congested (queue overflow for long periods). RED discards packets proportionally from competing flows according to their average bandwidth usage once a precongestion situation is detected. As illustrated in Figure 3.22, this precongestion situation is identified by having the weighted average output queue size between a minimal (Q_{min}) and a maximum value (Q_{max}).

The RED algorithm uses a weighted average of the total queue length instead of current queue length to determine when to drop packets. This prevents RED from reacting from packet bursts, effectively allowing reaction to long-term flows.

Packets are sent to an output queue whenever the weighted-average queue length is smaller than Q_{min}. Early drop verification is performed on packets whenever the weighted-average queue length is greater than Q_{min} and smaller than Q_{max}. A forced drop will occur whenever the weighted average queue length is greater than Q_{max}.

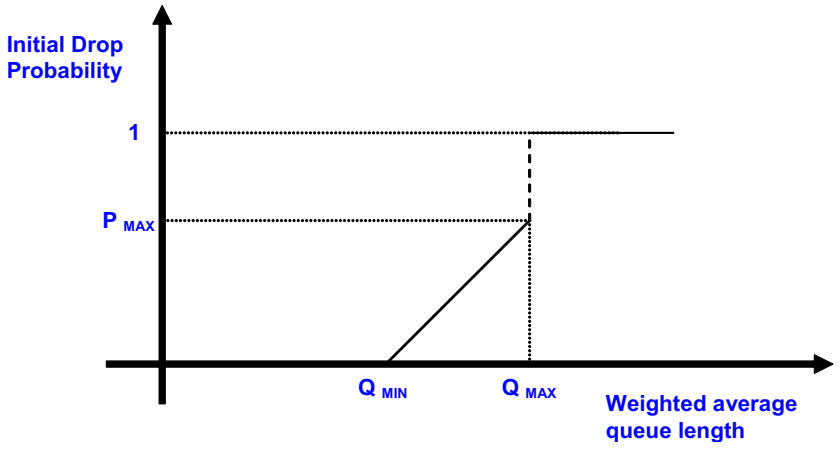

Figure 3.22 RED thresholds.

The main RED control parameters are illustrated in Table 3.2. In RED, the probability of early drop packet discarding depends on various parameters. The packet drop probability calculation follows [2]:

$$P_b = P_{\max} (\text{avg} - Q_{\min})/(Q_{\max} - Q_{\min}) \quad (3.1)$$

$$P_a = P_b / (1 - \text{count} * P_b) \quad (3.2)$$

Where:

P_b = initial drop probability
P_a = actual drop probability
avg = weighted-average queue length
$P_{\max}$ = maximum drop probability
count = number of enqueued packets since the last dropped packet

It follows that initial probability varies from 0 (zero) to $P_{\max}$, and the actual drop probability increases with the number of enqueued packets since the last one was dropped. The general effect is that higher rate flows will have a higher number of dropped packets, since their packets arrive at a higher rate than slower rate flows.

RED operation assumes a higher-level protocol (transport); otherwise, the application eventually will somehow react to packet loss as an indication of congestion.

With TCP, the slow-start algorithm is activated and causes the reduction of packets flowing among TCP peers for the TCP connection that experienced a packet loss. Generally, it is expected that this effect will help reduce the precongestion condition and bring the total amount of enqueued packets to below the minimal threshold ($Q_{\min}$). In the case where congestion persists, packets are systematically discarded after the maximum threshold ($Q_{\max}$) is reached.

There are variations of RED in which the thresholds are chosen according to traffic priority, or the discard probability is adjusted to guarantee fairness, among other possibilities. Examples of these variations are weighted RED (WRED), distributed weighted RED (DWRED), fair RED (FRED), stabilized RED (SRED), and balanced RED, among other alternatives [4].

RED has no effect on UDP data flows. This is because, contrary to TCP, UDP does not back off during congestion. Also, some TCP flows may behave as a nonresponsive flow or, in other words, as an aggressive flow. That happens, for instance, with optimized TCP implementations that do not implement TCP congestion avoidance mechanisms, as proposed by IETF [5].

Table 3.2 RED Control Parameters

RED Parameter	Description
Q_{LEN}	Maximum queue length
$Q_{\min}$	Queue length threshold for triggering probabilistic drops
$Q_{\max}$	Queue length threshold for triggering forced drops
$P_{\max}$	Maximum probability of early dropping

3.4.3.3 Explicit Congestion Notification Explicit congestion notification (ECN) is another alternative for preventing congestion; it allows routers to set a congestion signaling bit (congestion experienced bit (CE)) in packets from ECN-capable transport protocols. The basic idea behind ECN is to signal congestion explicitly, instead of signaling congestion by dropping packets.

Using ECN instead of packet drop as a mechanism to signal congestion has the following advantages:

- Signaling congestion reduces packet loss.
- ECN allows different actions on end nodes.
- Coupled with active queue management, in which routers detect congestion and precongestion situations, ECN effectively separates the policies for queuing and dropping from the policies for indicating congestion.

To this end, ECN signaling must be carried in IP packets. The bits used are illustrated in Figure 3.23.

The ECN-capable transport bit is set by the data sender to report that ECN capability is available on the endpoints. The CE bit is set by the router to indicate congestion to the endpoints.

Basic ECN operation is very simple, as illustrated in Figure 3.24. Once a router receives a packet from an ECN-capable transport source and congestion is detected, the CE bit is set and the packet is forwarded to its destination (in RED, the packet would be even-

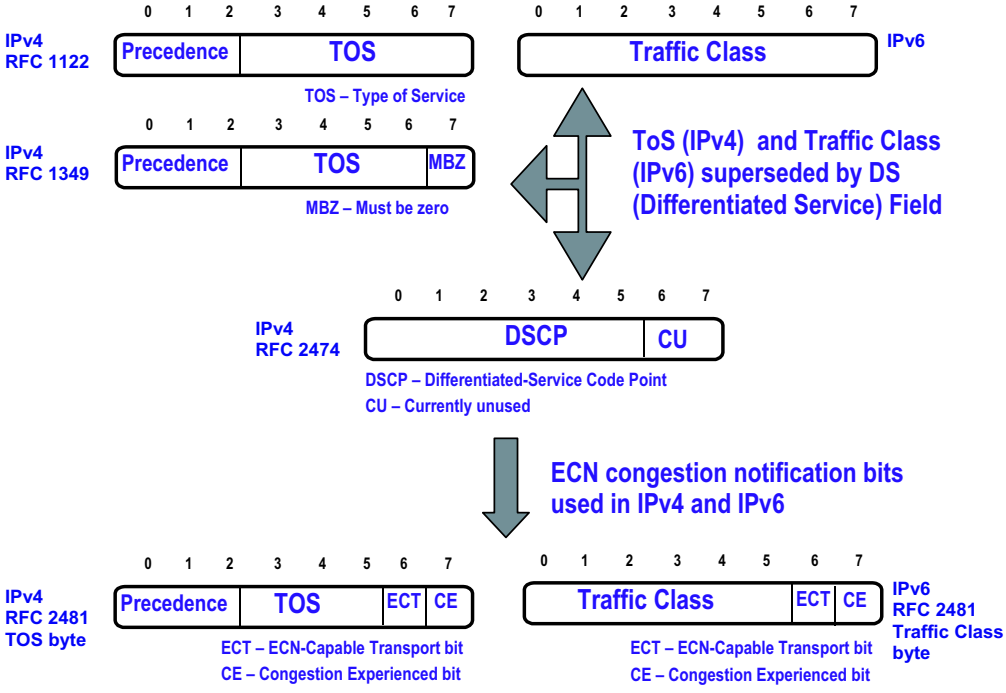

Figure 3.23 ECN bits in IPv4 and IPv6 packet headers.

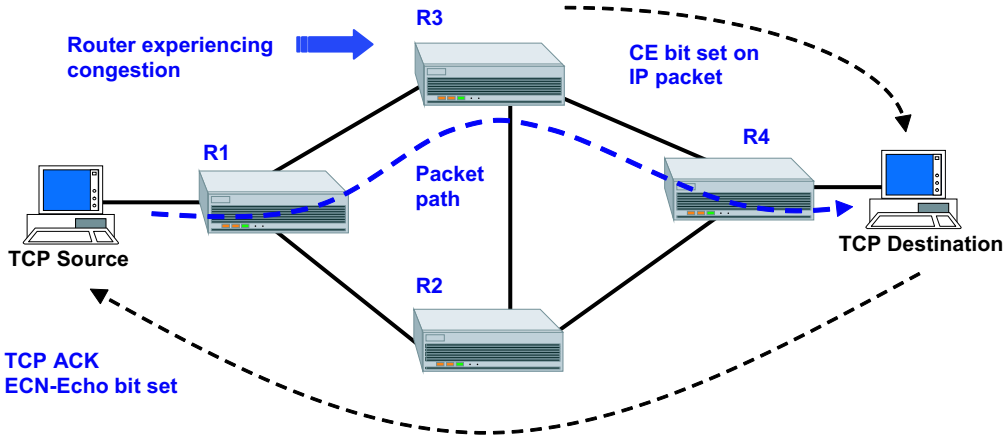

Figure 3.24 ECN notification between IP and TCP peers.

tually dropped). At the destination, ECN assumes that the transport protocol (for instance, TCP) will signal back to source the congestion indication. Upon receipt of that signal, the source will execute essentially an equivalent congestion-control algorithm as that used for packet dropping. This last condition is essential for the coexistence between ECN and non-ECN endpoints and nodes, allowing the gradual adoption of this solution.

As indicated earlier, the transport protocol must support ECN. In TCP, three mechanisms are required:

- Negotiation between endpoints to identify ECN capabilities.
- An ECN-echo flag is designated to signal back to the TCP source the congestion indication bit (CE).
- A congestion window reduced (CWR) flag is designated to indicate when the TCP destination should stop sending the congestion indication bit (CE).

The ECN-echo flag is set by the TCP destination in the next TCP ACK (ECN-echo ACK packet) whenever a CE bit is received. When a TCP ACK with ECN-echo set is received by the TCP source, it triggers the TCP congestion control algorithm:

- TCP halves the congestion window (cwnd)
- TCP reduces the slow start threshold (ssthresh)

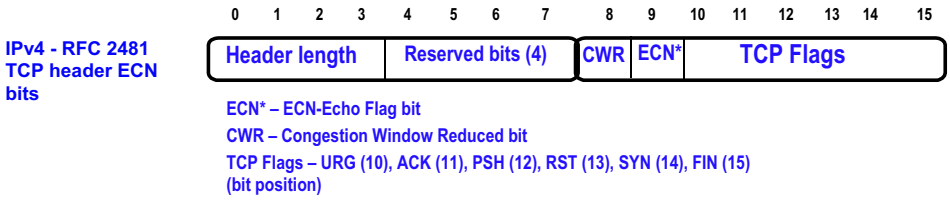

Figure 3.25 ECN bits in TCP header.

After these steps cause the TCP data flow to back off, the TCP source will set the CWR bit in the next TCP data packet in order to signal to the TCP destination to stop sending TCP ACK with ECN-echo.

In the TCP header, the ECN bits are positioned as indicated in Figure 3.25.

3.5 INTEGRATED SERVICES ARCHITECTURE

In this section we will introduce the main initiatives being proposed by the IETF, and we will elaborate on the IntServ [6].

3.5.1 IETF Quality of Service Strategies and Initiatives

The IETF has been actively involved in the development of a standardized support for multimedia applications over IP networks and, in particular, for the Internet.

For the scope of this discussion, we will consider three major standardized alternatives, which can be used to provide QoS for multimedia applications:

- The IntServ
- The DiffServ
- The MPLS solution

This section presents and discusses IntServ, a proposed IETF strategy oriented to provide QoS to individual flows.

3.5.2 Goals and Basics

The main idea behind the IntServ architecture initiative is to expand the Internet basic service model.

The basic Internet model is based on best-effort services. Best-effort services do not guarantee QoS parameters like bandwidth and delay and, thus, are very limited for supporting many user applications.

The IntServ architecture proposes an "Integrated Service Model." In this model, individual flows are identified and specific QoS flow requirements may be enforced on an end-to-end basis. IntServ effectively allows an application to request QoS from the network with high capillarity and thus is of great importance to end users.

In IntServ, all types of digitally represented information (conventional data, voice, video, text, graphics, and so on) are transported in a single and unique network infrastructure, with their QoS requirements being either controlled or guaranteed.

Obviously, the benefits of this new set of capabilities have a great impact on IP networks:

- For corporate networks, merging different networks onto fewer or, eventually, just one network, may result in significant economic benefits.
- The Internet represents the possibility of supporting multimedia applications with high capillarity.

It is important to observe that this goal is not a new idea. The integrated services digital network (ISDN) in the 1970s and the broadband ISDN (B-ISDN) in the 1980s were earlier attempts to provide a common or integrated infrastructure for a wide range of applications. IntServ, the new architecture proposed, targets IP networks and, in particular, the Internet.

The point now is how to implement the "Integrated Services Architecture" or, in other words, how to put this new model into practice?

The proposed new capabilities of integrated services require the following basic functionalities:

- Network nodes, mainly routers, must be able to support and control QoS delivered for IP packets.
- Application of QoS requirements must be signaled to all network nodes involved in the end-to-end path from source to destination.

The first functionality mentioned earlier corresponds to the "service" supported by the IP network nodes (routers) and must be precisely defined. The following services have been defined for IntServ compliant networks:

- Guaranteed service
- Controlled-load service

The second functionality can be provided by a signaling protocol. RSVP [7] is the protocol frequently used in IntServ implementations (Fig. 3.26). Thus, RSVP can be considered as an essential component of the IntServ initiative, responsible for signaling QoS requirements for nodes in the network's infrastructure. RSVP characteristics are detailed in Section 3.5.5, and have great importance on the overall IntServ applicability.

Before discussing RSVP in more detail, we will explore the IntServ architecture, its characteristics, and basic services.

3.5.3 Integrated Service Architecture: Characteristics and Services

There is a basic conception principle adopted by the IntServ architecture: "The Quality of Service in IntServ compliant network infrastructures is supported by a *resource reservation mechanism* applied to packet flows."

This conception principle requires IntServ nodes and users to follow an operational model summarized as follows:

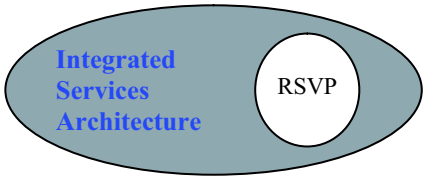

Figure 3.26 IntServ and RSVP.

- First, QoS parameters have to be identified by users in order to be properly requested from network nodes.
- Once users identify their QoS application requirements, they must request IntServ nodes to support them. This is done, for instance, by requesting services using the RSVP signaling protocol between end-user and network nodes.
- Since IntServ uses a reservation-based model, nodes must perform validation steps before granting access to the required QoS parameters. In other words, network nodes must perform admission control before accepting any new flow of packets between users.
- Once access is granted to the network, IntServ nodes must continuously identify flows, provide the required QoS for the flows, verify conformance to flow characteristics, and police traffic flows.

Other important characteristics of IntServ are:

- IntServ uses state information to keep track of IP packet flows through the network. State information is kept on IntServ routers and is necessary for service processing. This characteristic differs considerably from the classic IP assumption in which packets are processed independently at every node, and it has some scalability implications discussed later.
- IntServ does not attempt to alter basic packet routing. In effect, routing paths are processed and maintained using any routing protocol (unicast or multicast) like open shortest path first (OSPF) and border gateway protocol (BGP), among other alternatives.
- Best effort traffic is the default service for packets that are not part of defined flows. Also, best effort traffic has to be considered in the overall IntServ design. In effect, the allocated resources for IntServ services (bandwidth, memory) should not cause best effort flows to starve.

In the next section, flow definition and flow requirements that the IntServ will support are discussed.

3.5.3.1 Flow Definition and Requirements

Data flows have to be characterized as signaled source, destination, and network nodes. Figure 3.27 illustrates a typical flow specification signaling for a given packet flow between a source and a destination.

TSpec defines the source packet's flow specification, giving characteristics of the data traffic to be handled. TSpec indicates transmitter configuration parameters such as peak rate, average transmission rate, and packet size.

RSpec defines the receiver packet's flow specification. RSpec indicates the desired service or, alternatively, the receiver configuration. Parameters defined in RSpec are the service requested and its characterization, for instance, in terms of delay bounds.

3.5.4 Services

The services defined in IntServ architecture have the philosophical intention of eliminating, as much as possible, the randomness characteristics of delay, jitter, and bandwidth

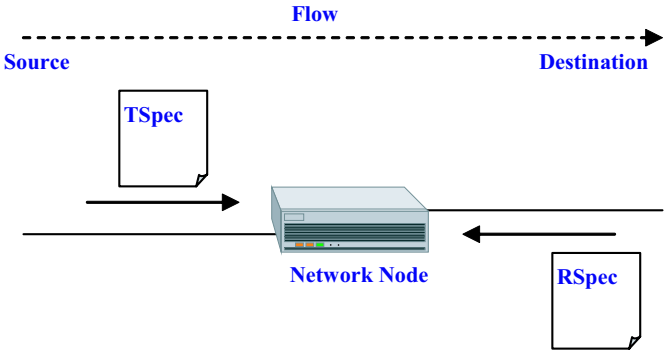

Figure 3.27 IntServ flow specification.

found in basic IP services. In other words, IntServ services will attempt to guarantee some QoS parameters on an end-to-end basis.

As stated earlier, there are two basic services defined in IntServ architecture, which are also called service elements:

- Guaranteed service
- Controlled-load service

3.5.4.1 Guaranteed Service Guaranteed service guarantees limited delays for packets on a policed flow with no queuing loss. The resulting end-to-end behavior for this service is such that a maximum delay bound is guaranteed all along the application's flow path. Guaranteed service also guarantees bandwidth limits, since there is an agreement about the maximum flow bandwidth the user is supposed to deliver at any time. This property of the flow is actually policed at every network node.

Guaranteed service can also be understood as the equivalent of a conventional leased-line service where there is a precise definition for the available bandwidth and corresponding transmitting delay.

IntServ specifies requirements for achieving a guaranteed reservation, but does not specify how to set up these requirements. Various setup methods can be used to guarantee reservations in the IntServ model. Here are some alternatives:

- To use a signaling protocol, for instance, like RSVP.
- To use a management protocol, for instance, like SNMP to properly configure reservation parameters at every network node.
- To manually configure the nodes.

It is important to observe that only queuing delay limits are guaranteed by IntServ. As discussed in Section 3.2.4.2, end-to-end networking delay is a more general definition and has three basic components:

- Propagation delays
- Transmission delays
- Processing delays

Transmission delays refer mostly to the time required to send and receive packets over data communications links, and can be easily calculated once the link properties (speed, packet length, technology) being used are specified.

Processing delays comprise both frame processing delay, packet processing delay, and upper-level processing delay at hosts. In routers, it is a reasonable approximation to consider the processing delay component to be the summation of queuing delay processing and packet encapsulating delay processing since most of the processing involved is related to queuing mechanisms and packet encapsulation. Guaranteed service, therefore, could also be roughly understood as a solution that defines limits on processing delay for routing.

IntServ users must properly identify the queue limit when making a request of the network by analyzing the various delay components present in a network end-to-end path.

Guaranteed service is intended for hard real-time applications, which need to have guarantees, for instance, that packets will arrive at the destination within a maximal arrival time limit. Multimedia applications are potential users of IntServ guaranteed service. VoIP, for instance, is an example of an application that has well-defined parameters for packet arrival time and, as such, could benefit from this service.

3.5.4.2 Controlled-Load Service The controlled-load service element is intended to guarantee QoS under overloaded traffic conditions. The basic idea of this service is that the QoS provided to the user's flow is quite close to the QoS this flow would receive under unloaded conditions, even if the network element were actually overloaded. Also, the applicability of this service is based on the observation that for the specific flow using controlled-load service the IP network works adequately if it is not heavily loaded. Therefore, the controlled-load services are adequate only for certain classes of application, in particular, those applications (like the Web-based ones) that are elastic or adaptable to modest fluctuations in network performance.

The principle behind controlled-load service is the service element's ability to keep QoS guarantees under heavily loaded conditions by controlling flow admissions and adopting queuing mechanisms with priorities to differentiate flows.

Controlled-load QoS guarantees in terms of parameters should be understood as follows:

- A high percentage of packets is successfully transmitted.
- The transit delay variation experienced by packets is kept small for a high percentage of transmitted packets.

In general terms, the behavior of a series of controlled-load service elements (routers, switch routers, or subnetworks) always resembles the best effort service provided by the same network element under unloaded conditions. If overload occurs, the QoS of controlled-load flows will behave better in relation to best effort serviced flows.

An Implementation of controlled-load services, for instance, may use a queuing mechanism with priorities to distinguish between controlled-load flows and best effort flows.

Applications should request controlled-load services to obtain, for instance, a more "stable" behavior under overloaded traffic conditions. Since the reservation-style used in IntServ is dynamic, any application capable of self-monitoring performance can benefit from IntServ controlled-load service. In this case, anytime the application monitors an un-

acceptable performance degradation when using a best effort service, it could dynamically switch to controlled-load service in order to improve its performance. Controlled-load service is also recommended for applications whose traffic characteristics are reasonably approximated by a TSpec. Applications characterized by more "precise" TSpecs generate less out-of-profile packets, and consequently have a more stable behavior guaranteed by network design.

In summary, IntServ architecture provides two basic solutions for QoS as follows:

- A leased-line style solution, oriented to hard real-time applications having strict delay requirements, called guaranteed service.
- A less strict solution, whose main objective is to keep the approved flows QoS parameters more invariant to instantaneous loading conditions on network nodes, known as controlled-load service.

3.5.5 Resource Reservation Protocol

The resource reservation protocol (RSVP) is a signaling protocol used by senders and receivers in a path to reserve the network resources that they need to guarantee proper application functioning. After proper setup, applications using RSVP perform with QoS.

What does the term "signaling" mean for RSVP and the integrated service model?

RSVP is not a transport protocol or, in other words, it does not transport data. In effect, RSVP carries a series of commands requesting and granting the resources (signaling) necessary for the proper execution of applications. IntServ supports requests for QoS parameters like minimal bandwidth and maximum delay. The reservation commands carried by RSVP are described in Section 3.5.5.3.

The RSVP protocol has various important characteristics. A brief discussion of each one follows.

Receiver-Oriented Protocol RSVP is a receiver-oriented protocol, which means that receivers request reservation services. This is a flexible solution, since receivers know exactly what QoS they need, independently of transmitter capacity. It often happens that a receiver requires much fewer networking resources than the transmitter's full capacity. As an example, a video flow of 1.5 Mbps may be supported by a video server, but the personal computer receiving this data flow is only capable of handling 258 kbps of video flow due to processing limitations. In this case, RSVP receiver-oriented characteristics allows the flow to be adjusted to the receiver's capacity, thereby reducing utilization of network resources.

Soft State RSVP uses soft state to keep track of flow reservations and their characteristics. Each network node maintains state control for all defined flows with reservations. Periodically, RSVP sends refresh messages (REFRESH) to maintain the state in all the nodes along the used path. The principle is that if no refresh message arrives in time, the actual state is destroyed and a new one has to be built upon. Soft state supports the dynamics of RSVP. In effect, users can change their reservations anytime either by application or end-user requirements. Also, new users can be added to flows dynamically. Reservations can also change due to routing reconfiguration or route failure. RSVP soft-state characteristics provide flexibility and dynamics to the network operation.

RSVP Data Flow RSVP data flows are called "sessions." The receiver side of RSVP sessions is identified by:

- Destination address (unicast or multicast);
- Protocol identification; and
- Destination port (optional).

Each data flow is considered independently of any other data flow in RSVP. The destination port should be thought of as a general identification used to separate flow information at the transport or application level. Frequently, TCP and UDP ports are used as destination port parameters.

Simplex operation RSVP is a simplex signaling protocol. Specifically, reservations are requested separately for each flow direction (source to destination and destination to source). This characteristic is relevant since, for many applications, the flow resulting from the user's communications is not balanced. As an example, in client/server applications the server-to-client flow normally carries much more application data than the flows in the opposite direction.

RSVP and Routing Protocols RSVP does not attempt to route IP packets or flows. RSVP uses routing tables created by current routing protocols (OSPF, BPG, RIP, MOSPF, etc.) to find paths between users through the network. Also, RSVP is designed to operate with any future unicast and multicast routing protocol. These considerations have two implications. First, reservations requested using RSVP do not necessarily cause any traffic engineering action on the network. In this context, reservations are only used to allocate available resources independently of any global network optimization. Second, in case of route failure or route modification, the reservation process has to be repeated, since the reservation is set up for a routing path between sender and receiver.

IPv4 and IPv6 Both IPv4 and IPv6 protocols support RSVP. As stated earlier, RSVP uses these routing protocols by reading their routing database for forwarding packets. IPv6 packets can be tagged with "flow labels" in their headers. As the name suggests, flow labels can be used to mark packets as belonging to a certain flow. As such, this label can be used by classifiers in RSVP to identify which service has to be delivered to the packet.

3.5.5.1 RSVP Functionalities

RSVP operation requires the following functionalities on network nodes:

- Admission control
- Policy control
- Classifier
- Scheduling

The interoperation of these functions on hosts and routers is illustrated in Figure 3.28.

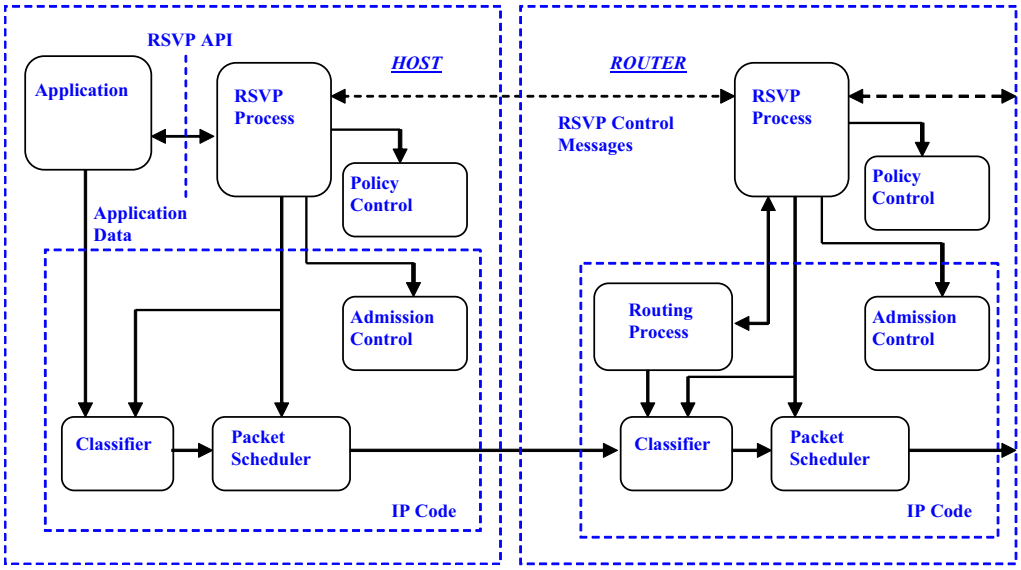

Figure 3.28 RSVP functionalities in hosts and routers.

The admission-control functionality is invoked for any new request for service. It is a decision module necessary for verifying the availability of resources at every node for new incoming requests during reservation setup.

The policy control module determines whether the user requesting a reservation has the necessary permissions. The policy control module is also invoked during reservation setup.

The classifier and packet scheduler are modules invoked during data exchange between senders and receivers. The classifier executes packet classification. Packets are associated with their specific flow and, as such, they will receive a particular QoS during forwarding processing at the packet scheduler module. Specifically, the packet scheduler identifies the packet flow and, eventually, the route information associated with the packet.

The packet scheduler module implements the scheduling mechanism for packet flows according to the service requested by the user.

In the hosts, the RSVP requests generated by applications are passed to RSVP processes through an API. This API is also responsible for passing any information or signaling generated by the RSVP process to the application programs.

Both the RSVP process and policy control modules are a user process in many RSVP implementations. The other modules composing the RSVP—the classifier, admission control, and packet scheduling—strongly interact with the IP code. As such, these modules are typically system modules within the operating system or kernel.

3.5.5.2 RSVP Operation Now that we have examined the basic RSVP functionalities, let us develop a more precise understanding of its operation. The basic operation principle used by RSVP is that senders announce their flows to all possible receivers and receivers subsequently reserve resources for these flows according to their needs.

RSVP signaling protocol operates as illustrated in Figure 3.29, and its operation is summarized below.

First of all, senders (the origin of flows) must characterize the packet's flow that they will generate. The TSpec identifies the traffic generated by any sender in IntServ.

A sender must inform all possible unicast or multicast receivers about its packet flow. *PATH messages* containing a TSpec are then send from the sender to all receivers. PATH messages are thus used to "inform" routers and end users of possible flows.

When unicast routing is being used, there is a one-to-one relationship between sender and receiver. When multicast routing is being used, the PATH message will follow the multicast tree, ultimately reaching all active receivers for that tree. As we stated before, RSVP uses the "normal " routing protocols (unicast or multicast) supported by network nodes.

PATH messages will then follow the routing "path" provided by the normal IP routing process all the way downstream from the sender to the receivers. All RSVP-compliant routers downstream will establish a "path identification" that will keep information about the upstream sender of the PATH message. Specifically, they know the address of the upstream router from which they receive the PATH message. This information is necessary for establishing a "reservation path" along the network for the TSpec flow.

The receiver reserves resources at each network node (for instance, routers) by sending a *RESV (reservation request) message* along the upstream "path" to the sender. The RESV message will flow upstream, hop-by-hop, through the previous "marked PATH" defined by the PATH message.

The RESV message defines the service requested (guaranteed or controlled-load) and carries the *RSVP reservation* composed by the sender flow specification, called TSpec, the receiver specification, called RSpec, and a filter specification called "Filter Spec" (Fig. 3.30).

The TSpec identifies and characterizes the transmitter's flow to which the reservation and requested service should be applied. Receivers recover TSpec objects from PATH messages sent earlier by transmitters.

The RSpec specifies the requested QoS for the flow. RSpec includes information such as the service requested and parameters needed for that service.

The "Filter Spec" defines IPv4 or IPv6 filters to specify the flow (addresses, port numbers, and flow descriptor—IPv6) to which the reservation applies. The filter specification is used to set parameters in the packet classifier (Figure 3.28). The filter specification al-

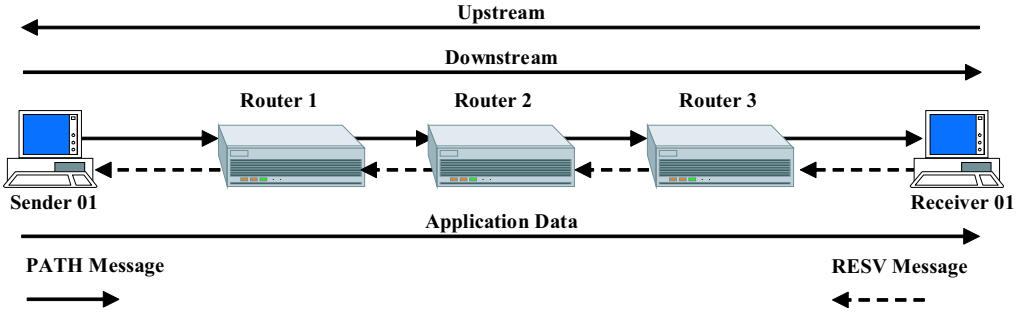

Figure 3.29 RSVP control messages.

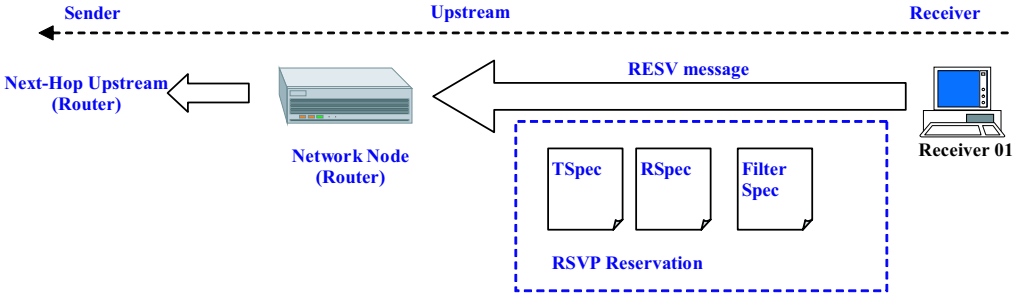

Figure 3.30 RSVP reservation with RESV message.

lows fine-coarse-grained filters. As an example, IPv6 packets with the same flow descriptor, but from different sources (different IPv6 addresses), might have different reservations and QoS guarantees in RSVP networks.

Upon reception of a RESV message, RSVP nodes must execute admission control. First, the user requesting a reservation is authenticated, and second, the node verifies if there are enough resources available to satisfy the reservation requested. In case of failure (authorization denied or unavailable resources), the node returns an error back to the receiver. Otherwise, the RESV message is sent upstream to the next network node in the path.

When the RESV message reaches the last network node and is accepted, a confirmation message is sent back to the receiver and the reservation setup process is completed. A confirmation message resulting from merging it with another already established reservation might also be sent by a node.

Receivers, senders, or intermediate nodes along the reserved path can initiate the teardown process. The teardown process removes reservations, thus releasing the corresponding allocated resources. Since the reservation is merged in the routers, the teardown process can result in selective pruning of state and resources. For instance, the teardown of one multicast receiver using less bandwidth than another multicast receiver causes the router to assign the allocated resources to the receiver remaining active.

There are two RSVP teardown messages, PATHTear and RESVTear. The first travels from receivers or routers, whichever one initiated the teardown process, in the direction of all senders. The second travels from senders or routers to all receivers in the reserved path.

3.5.5.3 RSVP Messages

The RSVP protocol uses the set of control messages indicated in Table 3.3 to allow users to make reservations.

PATH and RESV are the most important RSVP commands. A brief summary of their operation is in Table 3.4

3.5.5.3.1 PATH Message

To state their traffic specifications within an RSVP session, senders generate PATH messages. Since both unicast and multicast are supported, PATH messages can reach one or multiple receivers within an RSVP session.

PATH messages carry two basic objects (Fig. 3.31):

- TSpec (transmitter specification)
- ADSpec (advertising specification)

Table 3.3 RSVP Messages

Control Message	Action	Generation
PATH	Inform sender's traffic flows	Sender → Receiver
RESV	Reserve resources for sessions	Receiver → Sender
RESVConf	Confirm reservation request	Sender → Receiver Node → Receiver
PATHTear	Teardown process	Sender → Receiver Router → Receiver
RESVTear	Teardown process	Receiver → Sender Router → Sender
PATHErr	Error reporting in PATH message processing	Router → Sender
RESVErr	Error reporting in RESV message processing	Router → Receiver

As stated later, TSpec carries the senders data flow characteristics in terms of a token-bucket traffic model and identifies the sender, while ADSpec carries information generated by either senders or intermediate nodes all along the downstream path. The ADSpec objective is to support the identification of the QoS parameters, available services, and network characteristics that will be used by applications to properly request the network service. As an example, information carried by ADSpec include:

- The services (guaranteed, controlled-load) supported by nodes.
- If there are weak points for end-to-end QoS guarantees, such as a network node not supporting RSVP reservations.
- The minimal path maximum transfer unit (MTU), which is a necessary parameter to avoid packet fragmentation all along the reserved path.
- Identify the "C" and "D" delay bounds necessary for determining guaranteed service reservation parameters.

Table 3.4 RSVP Messages PATH and RESV

RSVP Message	RSVP Message Processing
PATH message	Send by sources of data flow Information provided to receivers: • source data flow characteristics • network resources Nodes processing: • identifies path upstream from receivers to source • finds route to receiver
RESV message	Send by data flow's receiver Information provided by receivers: • service requested • service parameters Nodes processing: • checks resources availability • establishes reservation

ADSpec information is altered and updated by routers all along the reserved path, but TSpec remains unaltered.

ADSpec provides an enhancement to the reservation setup process. In effect, receivers request reservations sending RESV messages that are either accepted or rejected at each node. Upon rejection, receivers have little recourse beyond guessing the next request, unless they have some previous knowledge of network parameters. ADSpec, also called "one pass with advertising" (OPWA), provides "knowledge" to receivers by collecting network characteristics and parameters.

Once PATH messages reach their destination(s), TSpec and ADSpec objects are passed to user application. The application then computes its RSpec reflecting its QoS needs based on the information gathered.

Figure 3.32 illustrates the exchange of PATH and RESV control messages during the reservation setup process.

3.5.5.4 Reservation Models Resource reservations in RSVP can have different styles or reservation models. The style or reservation model determines the treatment a RSVP node will apply to data flow sources within a RSVP session. Figure 3.33 illustrates possible alternatives. First, when multiple senders are in the same RSVP session, resources might or might not be shared among then. For instance, resources might be shared among data sources, or each source could use "separate" resources. Second, the sender's choice in RSVP reservations may be deterministic (for instance, including all source IP address) or, alternatively, reservations may use a wildcard representation to senders. The possible reservation styles actually defined are:

- Fixed-filter style
- Shared-filter style
- Wildcard-filter style

In the fixed-filter style (FF), "separate" reservations are created for "explicit" senders. In that case, different senders for the same session will have separate network resources reserved by the RSVP node. The RSVP representation for the FF style is:

$$FF(S\{Q\}) \tag{3.3}$$

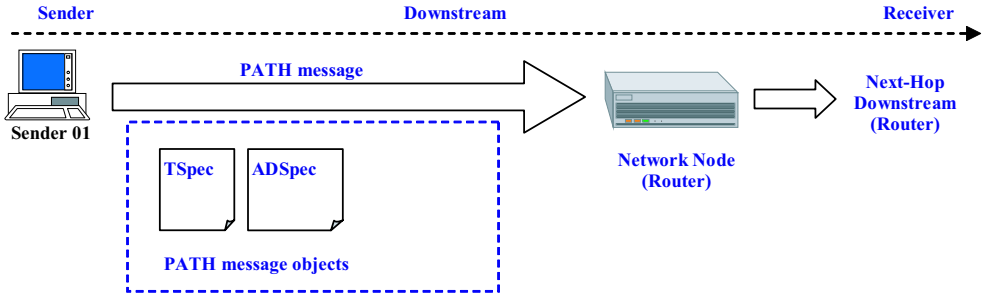

Figure 3.31 PATH message.

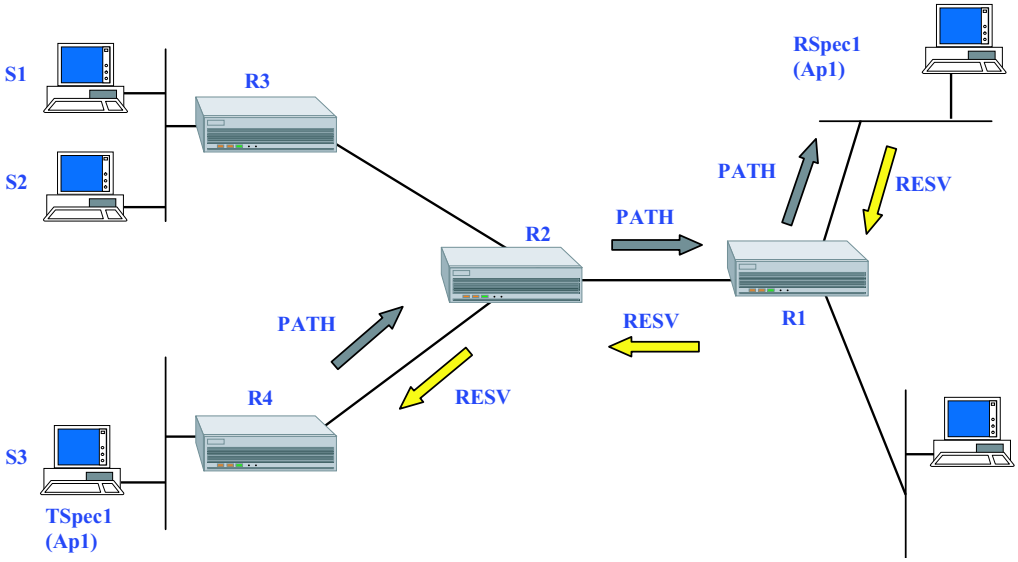

Figure 3.32 Control message exchange during setup process.

where

S = sender
Q = flow specification (flowspec)

In the shared filter style (SF), reservations (implicitly network resources) are "shared" among "explicit" senders. In this case, a group of senders "share" resources allocated for a session. The RSVP representation for the SF style is:

$$SF(S1\{Q\}, S2\{Q\}, \ldots) \tag{3.4}$$

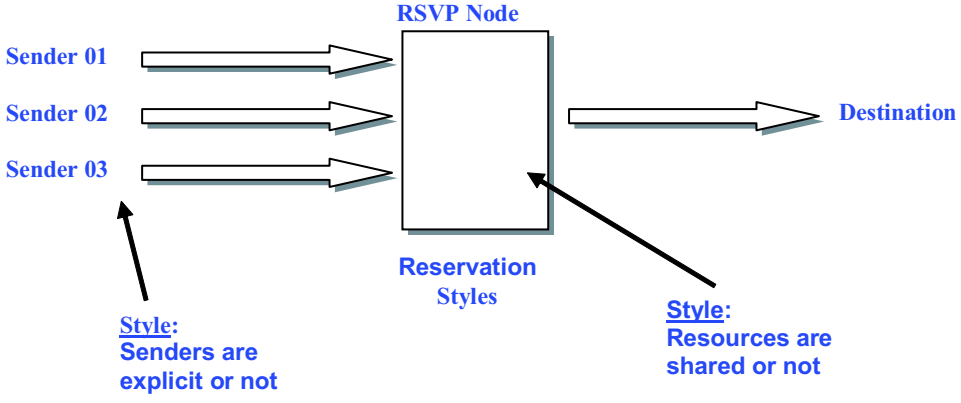

Figure 3.33 Alternative styles for RSVP reservations.

In the wildcard-filter style (WF), reservations are "shared" among wildcard-selected senders. WF reservations correspond to having a single reservation being used by flows from all upstream senders. With the WF style, new senders can join the session at any time and share the reserved resources. The RSVP representation for the wildcard-filter style is:

$$WF(*\{Q\}) \tag{3.5}$$

3.5.5.4.1 Reservation Merging The RSVP operation allows reservations to merge in an appropriate and efficient way. Reservation merging results in optimized network resource utilization. The point is that network resources (bandwidth, memory, etc.) are scarce and therefore they should not be wasted. As an example, Figure 3.34 illustrates reservation merges with different resources. At router R1 reservations are merged such that the same flows receive, when available, the highest level of requested resources. From R1 to R3, for instance, an upstream RESV message will carry an RSpec requesting QoS levels defined by Q1, which is assumed to be higher than Q2.

By definition, different reservation styles cannot be merged since erroneous situations may occur.

Figure 3.35 illustrates another example of reservation merge, where the WF style is used. Multicast applications could use this reservation style to reserve resources. In that case, the multicast receiver sends reservation request to any sender. The RESV message follows the multicast tree upstream to all possible active senders. From the RSVP point of view, active senders are those senders that previously sent PATH messages. Also, it is important to notice that RSVP allows many-to-many configurations, in which multiple senders are simultaneously sending flows to multiple receivers.

3.5.6 RSVP: Final Considerations

Now that we have examined the overall RSVP operation, let us discuss its application to networks in more detail. The following technical aspects are considered:

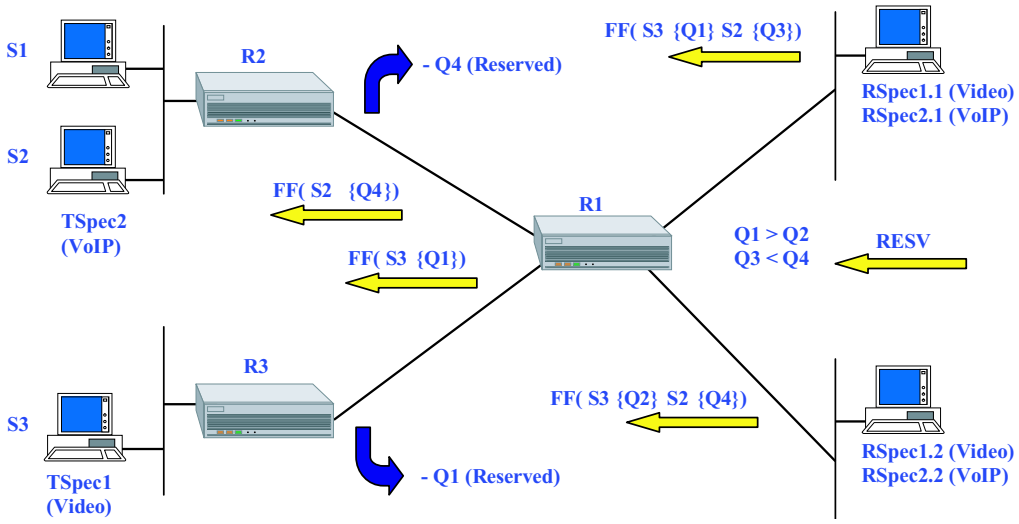

Figure 3.34 A RSVP reservation merge for the FF style.

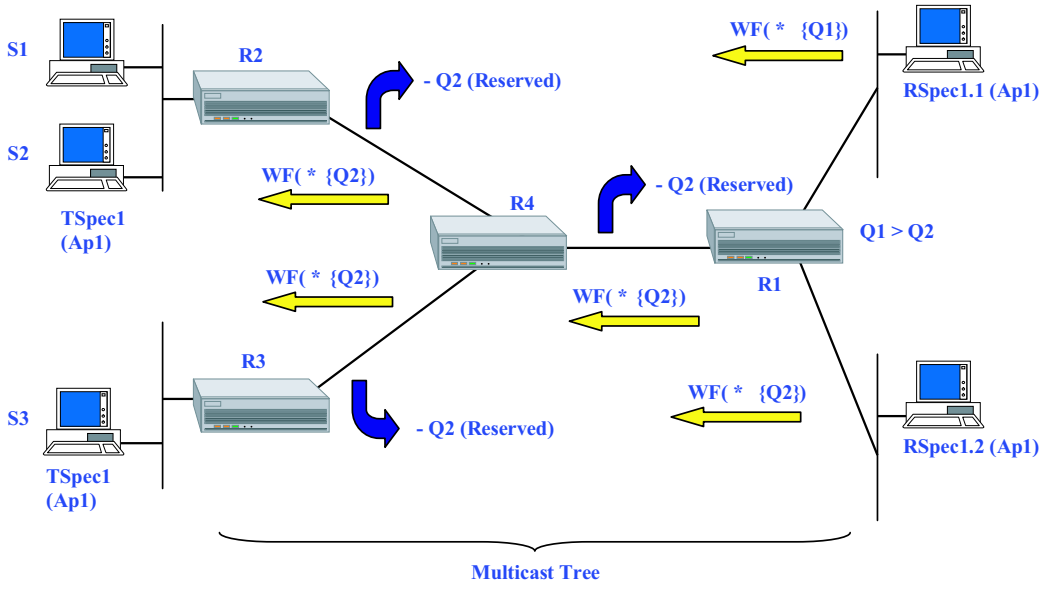

Figure 3.35 Reservation merge with the WF style.

- Signaling complexity
- State control
- Granularity
- Migration path

In brief, there is a scalability problem to handle when using RSVP as a generalized solution for supporting QoS guarantees over complex network infrastructures.

In effect, RSVP requires interior nodes to keep information on the state of each flow. In large networks, this leads to the so-called "state explosion problem," and effectively represents a potential problem for RSVP implementation. For reasonably sized networks, in which the number of total flows is somehow limited, the scalability problem does not represent a potential problem.

Another problem faced in large networks by RSVP is signaling complexity. Here, we have two basic considerations: the complexity involved in RSVP message processing and the number of messages necessary to set up and maintain a reservation. RSVP message processing requires a great deal of processing and may be a limitation for high-speed routers. The number of exchanged messages is another potential problem. As an example, reservations are kept in "soft" states and, as such, must be periodically refreshed. This characteristic limits RSVP applicability to networks in which the number of simultaneous flows is large.

A final concern about RSVP implementation is related to the migration path necessary for adopting the solution. In effect, RSVP requires a "system" to be installed. In other words, routers, hosts, and applications must be updated in order to guarantee an end-to-end support for this solution in the network. Since this requires time and investment, migration is not necessarily straightforward.

Granularity is an important characteristic of RSVP protocol and its corresponding integrated services model. In effect, real-time, mission-critical, and other new applications require QoS flows to be explicitly enforced. In this context, RSVP and the integrated service approach is certainly necessary.

In the context of complex networks, where applications require flow control, it may be possible that an IntServ solution will be integrated with other QoS initiatives in order to provide both flow control and high-performance processing at network nodes.

3.6 DIFFERENTIATED SERVICES ARCHITECTURE

There are three major standardized alternatives to support QoS enforcement for multimedia applications: the IntServ architecture, the differentiated services (DiffServ) architecture, and MPLS. This section presents DiffServ [8], a coarse-grained approach providing QoS to aggregated traffic. This initiative has been standardized in IETF.

3.6.1 Goals and Basics

The main idea of the DiffServ architecture initiative is to provide differentiated services to an "aggregate of flows" in an IP network.

In DiffServ, flows are not treated individually but are considered in groups. In effect, only a few services or behaviors are instantiated at network nodes and, as such, user's flows must be treated in aggregates if they are to to receive the available services. It is expected that DiffServ will support a large number of user flows, thus allowing this solution to adequately support QoS in very large networks, such as the Internet.

As in the previously discussed solution, the DiffServ initiative intends to expand the basic Internet best effort service model. In its basics, DiffServ allows all types of digitally represented information (conventional data, voice, video, text, graphics, and so on) to be transported in a single and unique IP network infrastructure, while their QoS requirements are being supported.

The question we can now ask is how to put this new proposition to work in practice.

DiffServ architecture proposes a "differentiated service model." This model requires the following basic functionalities to be implemented in IP networks:

- Network nodes must be able to support and control the QoS required for IP packet aggregates.
- IP packets must be "marked" with their QoS requirements.

The first function just mentioned corresponds to the "services" actually supported by IP network nodes in DiffServ. IETF defines the following services:

- The Expedited Forwarding (EF) service
- The Assured Forwarding (AF) service

The second function can be provided by simply "marking" packets according to their required QoS. This marking procedure in DiffServ is often realized at network borders, but it is also possible to mark or remark packets at nodes within the network.

As an illustration, the packet markings in DiffServ compliant networks are realized by the differentiated service byte (DS byte), which is equivalent to a packet tag (Figure 3.36). Once marked, a packet is said to belong to a specific "class" that will receive a specific service at network nodes. This simple yet powerful principle allows, for instance, service providers to classify user's flows in a limited number of classes and provide to these classes a set of basic common services.

In the next section, we will explore in more detail the model and the services supported by a DiffServ-compliant network and how these services are mapped to network node operation.

3.6.2 Differentiated Service Architecture

First, it is important to realize what the main focus for DiffServ architecture services is EF and AF and what the operational model defined by the DiffServ architecture is.

The DiffServ model has been proposed to mainly support the QoS services in complex networks with many network nodes, like routers, and many simultaneous user applications demanding QoS. This is a situation with potential scaling problems, which is why the DiffServ architecture is made to scale to a large number of users.

Roughly, the basic idea is to:

- Aggregate user's flow and application traffic streams in a small and manageable number of *traffic classes*.
- Process and forward packets according to their assigned classes.
- Avoid using signaling at every hop.

Traffic stream aggregation is an important aspect in DiffServ. Users of the DiffServ network-capable infrastructure benefit from the services they need by having their packets marked with DSCP. Marking can be done, for instance, at DiffServ network borders, although packets can be also marked elsewhere:

- In hosts
- At the first hop router after the host
- In a specific equipment such as a VoIP gateway

This simple approach has a number of advantageous features for large and complex networks:

- First of all, packet processing is considerably reduced in network core nodes with respect to the total number of end-user flows (also called microflows). This happens

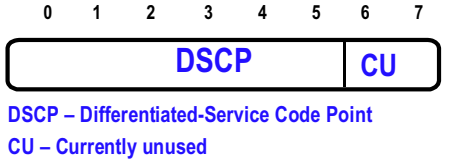

Figure 3.36 Differentiated services field (DS byte).

because there is no per-flow state, with packets being processed according to their class.

- Network provisioning is defined by the characteristics of the services provided for each class.

The DiffServ model presents potential benefits, for instance, in ISP networks. In effect, ISPs normally negotiate a service profile with their clients (users and applications). This negotiation is formalized by typically defining an SLA in which QoS or other traffic and service characteristics are agreed upon.

Once a user's SLA is defined, packets from users and applications have to be processed according to the services negotiated. For scaling reasons, SLAs ("service profiles") are negotiated and policed in the DiffServ network for a set of *aggregate flows.*

From the technical point of view, this means that ISP networks using the DiffServ model benefit from the fact that services provided in network nodes may be the same for various end-user flows (microflows), as long as packets from these flows are assigned for the same DiffServ class. Marking packets appropriately in a small number of predefined and negotiated classes is a more scalable solution, since there is less router processing involved.

Finally, ISP network service provisioning can be adjusted to the number of users, network resources available, and type of services supported by the network.

From the previous discussion we can write the basic conception principle used by DiffServ architecture:

> A reservationless model supports the QoS in DiffServ-compliant network infrastructures where *marked packets,* corresponding to a *group of end-user applications,* receive a predefined *service* or processing in network core nodes.

To understand this principle in more detail, let us consider more carefully the network nodes, the functionality, and the terminology adopted by DiffServ architecture. Figure 3.37 illustrates a typical DiffServ network or DiffServ domain.

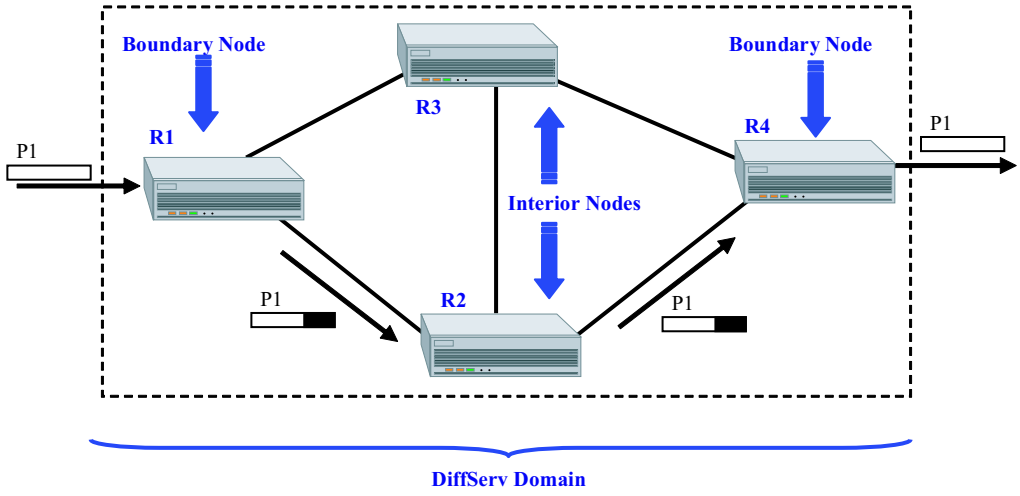

Figure 3.37 Basic DiffServ network elements.

First, let us consider the DiffServ *domain* concept. A DiffServ domain is a set of network nodes, typically routers, cooperating to provide predefined services (EF and AF) to packet streams classified into classes. There are two types of nodes in DiffServ domains: boundary nodes and interior nodes. Boundary nodes connect the DiffServ domain cloud to other domains. Interior nodes are connected to other interior nodes or boundary nodes and always belong to the same DiffServ domain. Boundary nodes can be either ingress nodes or egress nodes, depending on packet flow direction. As an example, the router located at the entry point is called an "ingress router" (Fig. 3.38). It typically implements a set of functions to allow "normal IP packets" to benefit from DiffServ services.

Core routers (internal routers) are mainly responsible for the optimized IP processing for marked packets. These marked packets are supposed to receive AF and EF services while traveling hop to hop through the DiffServ domain. In DiffServ terminology, the way the routers process the marked packets is termed "per-hop behavior" (PHB) (Fig. 3.38).

The router located at the outer edge of a DiffServ domain is called an "egress router." Egress routers process marked packets according to their required services and, may also unmark packets when they leave the DiffServ domain. Egress routers may also condition the outgoing traffic to make it compliant with interdomain traffic agreements.

3.6.3 Service-Level Specification and Traffic Conditioning Specification

As stated earlier, services are negotiated beforehand between customers and service providers and are typically defined in terms of service-level agreements (SLA) and traffic conditioning agreements (TCA) by the large majority of network service providers.

An SLA is a service contract between a customer and a service provider that typically defines all business and technical aspects of their interaction, such as:

- Services provided and their basic properties
- QoS parameters for each service

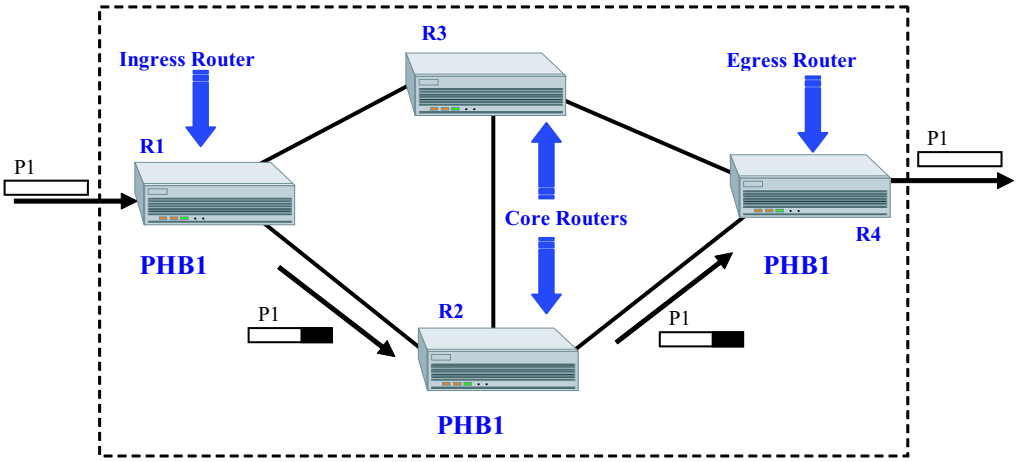

Figure 3.38 Per-hop behavior in DiffServ networks.

- Other technical aspects (availability, resilience, etc.)
- Rules for services (TCA)
- Contractual basis (pricing, discounts, etc.)
- Other technical or business aspects

TCAs can be understood as the definition of *requirements or rules for services*. In other words, it is necessary to precisely define the traffic stream (bandwidth, bursts, and other traffic characteristics) the service provider will handle for each customer and negotiated service. This is certainly essential for network resources provisioning and many other network operational characteristics, such as billing and fairness. An SLA may include the rules for services, which constitute the TCA.

SLA and TCA are general terms that encompass parameters and characteristics beyond the scope of DiffServ. This being so, new terms have been proposed for DiffServ-compliant networks.

Services provided by a DiffServ domain are defined in *service-level specifications* (SLS), and rules for this service are expressed in *traffic conditioning specifications* (TCS) (Figure 3.39).

SLS, like SLA, is a service contract that defines the service provided by the DiffServ domain, and typically specifies:

- The specific DiffServ service (AF, EF) the customer should receive
- Service parameter values
- The rules for that service (TCS)

TCS is an integral element of an SLS, and specifies traffic rules and requirements, such as:

- Traffic profiles
- Classifier rules
- Action on traffic streams (in-profile and out-of-profile traffic)

Traffic profiles specify the expected temporal properties of a traffic stream, the classifier rules specify how input traffic is classified, and actions are defined under various instantaneous and long-term traffic stream conditions.

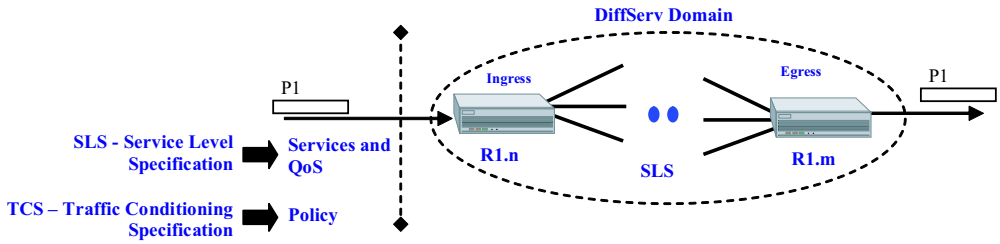

Figure 3.39 Service Level Specifications and Traffic Conditioning Specification in a DiffServ Domain.

In simple terms, DiffServ domains have an entry point and an exit point with respect to the stream of packets and negotiated services. In terms of the SLS, it defines the services that are implemented within the DiffServ domain.

In multi-domain DiffServ implementations, SLSs should be agreed at each network entry point and boundaries between DiffServ domains are not covered by SLSs (Figure 3.40).

3.6.4 DiffServ Functionalities

As mentioned earlier, ingress, core, and egress routers have to process marked packets within a DiffServ domain in order to implement predefined services (EF and AF) and to have PHBs.

In DiffServ architecture, the network nodes support the following functionalities:

- Packet classification
- Traffic monitoring
- Packet marking
- Traffic conditioning (shaping and dropping)
- Behavior classification
- Packet scheduling

Figure 3.41 illustrates a typical distribution of these functions among DiffServ network nodes. As illustrated in the figure, ingress routers are responsible for packet classification, stream monitoring, marking, and conditioning. Obviously, ingress routers also do forwarding processing for each packet according to its marked behavior. In most cases, core routers do only basic packet forwarding processing by identifying the correspondent packet behavior and scheduling it accordingly.

Egress routers, for instance, may optionally classify and condition packets at the DiffServ domain exit point for multidomain infrastructures. In this case, egress routers may be responsible for SLS/TCS conformance.

Let us now consider each one of these functions in more detail. Figures 3.42 and 3.43 illustrate how they interoperate in DiffServ routers.

Packet Classification The first functionality required is to classify packets in a traffic stream. The classifier is the entity within the ingress router that checks incoming packets against *service profiles.* Service profiles indicate the following:

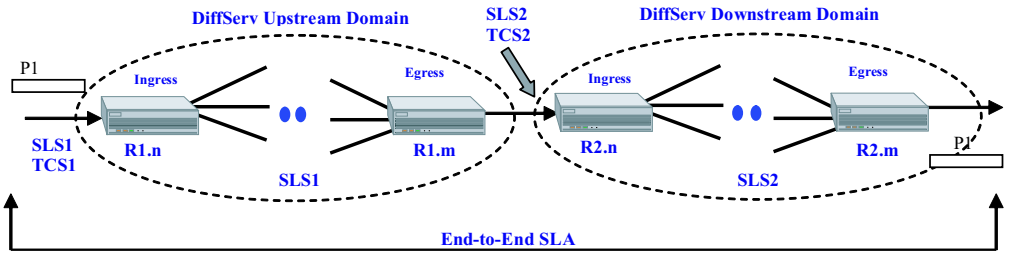

Figure 3.40 Multidomain DiffServ infrastructure.

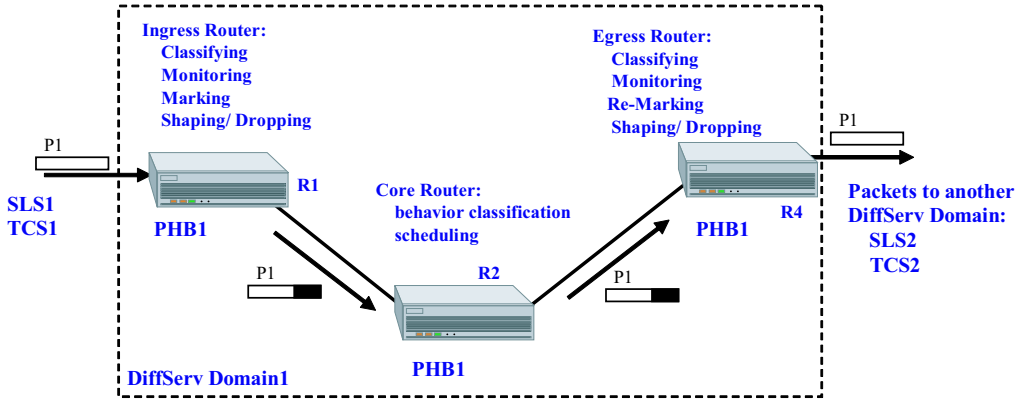

Figure 3.41 Typical functionalities in DiffServ routers.

- The flow of packets that will receive service
- The corresponding service.

Packets can be classified by IP address, protocol type, ToS, source port, destination port number, and other packet parameters or information. Multifield classification is allowed to obtain a fine-grain flow or microflow identification. There are two types of classifiers:

- Behavior aggregate (BA) classifier
- Multifield (MF) classifiers

The first classifies packets based on the DS code point only. The second classifies packets based on the value of a combination of packet header parameters, as indicated earlier. Packet classification is a time-consuming processing task realized once at the network border. Core routers do not normally execute this functionality.

Traffic Monitoring Traffic monitoring is a functionality intended to measure and keep track of the temporal properties (rate, bursts, etc.) of classified traffic streams.

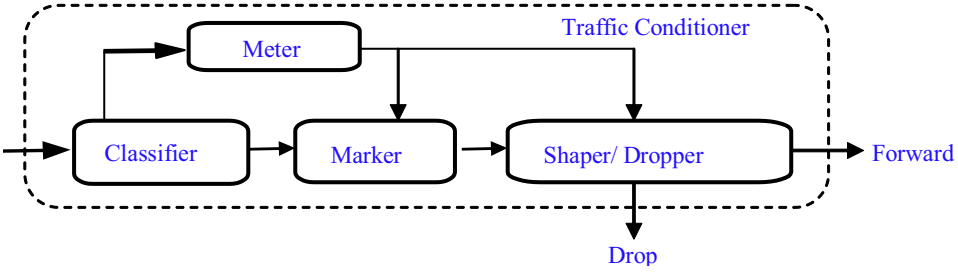

Figure 3.42 Typical ingress router functionalities.

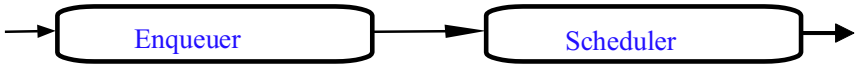

Figure 3.43 Typical core nodes functionalities.

TCS defines the expected temporal properties of the monitored traffic streams. Essentially, traffic monitoring determines whether packets for a selected traffic streams are in-profile or out-of-profile. This classification state can be used by the marking and conditioning functions (shaping and dropping) in many other ways. Some possibilities are:

- In-profile packets are normally allowed to enter the DiffServ network, since they conform to the traffic requirements and rules.
- Out-of-profile packets can be shaped, discarded, and remarked or receive other specific actions defined in SLS.

Typically, out-of-profile packets, if allowed to enter the network, receive inferior services in relation to in-profile packets.

Packet Marking Packet marking is a functionality required to set the DSCP in classified packets. Once marked, a packet will experience a defined PHB within the DiffServ domain. In other words, once marked, a packet will belong to a specific DiffServ behavior aggregate. The DiffServ field (DS field) holds the DSCP and uses six bits of the so-called DS byte, as illustrated in Figure 3.44. The value encoded in the DS field is called DSCP. The DS byte is placed in a packet's header and corresponds to either the previous "Type of Service" octet in IPv4 or the "Traffic Class" octet in IPv6. DSCPs are used in the packet-forwarding path to differentiate the services they will receive. DSCPs are mapped to PHBs at each network node.

Traffic Conditioning Traffic conditioning is a function that verifies and eventually alters the temporal properties of a traffic stream to bring it into compliance with a specific traffic profile. In the DiffServ model, traffic conditioning is performed by combining the classifier, meter, marker, shaper, and dropper functions. In brief, traffic conditioning can mark or remark packets, shape temporal traffic properties, or drop packets in accordance with traffic conditioning rules defined in SLS/TCS.

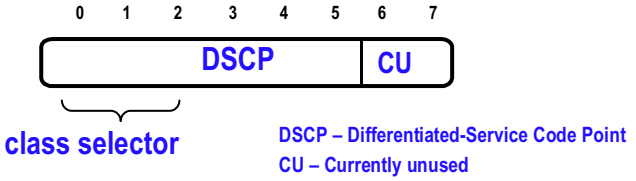

Figure 3.44 DSCP code point in DS byte.

Traffic Shaping Traffic shaping, as the name suggests, is the process of adapting a packet stream to conform to some defined traffic profile. Traffic shaping can be achieved, for instance, by delaying out-of-profile packets within a packet stream. Traffic shaping is a necessary function in the DiffServ model because traffic sources are typically unpredictable, both in short-term and long-term perspectives. As an example, a customer may negotiate an SLS with the following traffic-stream profile:

- R = rate of packets arrival (bytes/s)
- B = allowed burst size (bytes)

Eventually, the source may alter its traffic-stream properties, either in terms of rate of packets or peak bursts. The traffic shaper would then guarantee that packets out of the shaper are always in accordance with the former temporal properties independent of any unpredictable behavior experienced by the traffic generator.

Packet Dropping Packet dropping is another possible action for out-of-profile packets. In this case, packets are discarded as a result of *policing* the packet stream.

To summarize, classifiers and traffic conditioners are used in the DiffServ model to select which packets are to be added to a particular aggregate.

Behavior Classification Marked packets are grouped according to their DS byte values. This functionality corresponds to aggregating traffic flows into classes to receive predefined behaviors (PHB).

Packet Scheduling This last function schedules packets on a per-class basis.

Several scheduling mechanisms may be employed to deliver the defined PHBs. As an example, PHBs can be implemented by using PQ or WFQ, among other possibilities.

3.6.5 Differentiated Services and Per-Hop Behaviors

In the DiffServ architecture, the per-hop behavior (PHB), can be understood as the basic building block for "network services." At this point, it is important to observe that overall services are obtained as a result of multiple behaviors experienced by packets in network nodes. The term per-domain behavior (PDB), is used to express the overall behavior expected in a DiffServ domain.

Considering the packets themselves, PHBs define the specific forwarding treatment each one will receive at every node along its path within the network. In a more precise view, PHBs define how traffic belonging to a particular behavior aggregate or, simply, aggregate, is treated at individual nodes in a domain. In effect, packets from multiple sources may experience identical treatments at the same node or, in other words, may experience identical behaviors.

Another way to define PHB is presented by the IETF: PHB can be understood as the description of the externally observable forwarding behavior of a DiffServ node applied to an aggregate of packets flow.

Packets are allocated to behaviors (PHBs) in each DiffServ domain node according their marked DSCP encoding or code point.

3.6.5.1 DSCP Encoding and PHB Mapping

The DSCP encoding or code point is the value attributed to the DSCP part of the DS field (Figure 3.44) and maps to PHBs.

The code-point mapping considers the following:

- Standardized PHBs (EF, AF) have specific code points defined.
- Code point DSCP = 0 maps to a default PHB, for instance, best effort PHB.
- Experimental PHBs and local defined PHB mapping are allowed.
- Mapping between code points and PHBs is locally defined (although there is a mapping recommendation for standardized PHBs).

There are 64 possible DSCP values and there is a recommended standardized PHB mapping, as illustrated later. Also, DiffServ implementations are allowed to choose alternative mappings.

The standardized code-point encoding is the following:

- Best effort service (default behavior) → 000 000
- EF service → 101 110

The code points for the assured forwarding services are illustrated in Table 3.5.

Mapping between code points and PHBs is flexible and may use various relations such as 1-to-1 or *N*-to-1. The later relation means that multiple code points could be mapped to a single PHB, for instance, when considering experimental DSCPs or locally defined DSCPs. Also, the "class selector" bits, as illustrated in Figure 3.44, are intended to preserve IP precedence compatibility.

DiffServ defines two standardized PHBs: EF and AF. A discussion on the characteristics of these services follows.

3.6.6 Expedited Forwarding

Expedited forwarding (EF) behavior, also known as premium service, determines that the departure rate of the aggregate's packet from the DiffServ node must be equal or greater than a predefined rate.

EF service delivers minimum guaranteed bandwidth. EF guarantees at any time that the packet stream will always be allowed to use its minimum guaranteed bandwidth. From the user's point of view, the EF service has assured bandwidth with low latency, low jitter, and low loss. In other words, this service model emulates a leased-line service for its user.

Table 3.5 DSCPs for Assured Services

Dropping Precedence	Class 1	Class 2	Class 3	Class 4
Low	001 010	010 010	011 010	100 010
Medium	001 100	010 100	011 100	100 100
High	001 110	010 110	011 110	100 110

Several queue-scheduling mechanisms for delivering EF PHB are possible. These include:

- Priority queuing (PQ), with EF queue having maximum priority.
- Weighted round-robin (WRR) with bandwidth allocated to the EF queue equal to Ag_DRT_{min}.

Figure 3.45 illustrates the EF service with the following definitions:

$\Sigma_n AG_n$ = packet-arrival rate for all aggregates
L_Q = queue length
AG_N_ARt = aggregate's arrival rate
AG_N_DRt = aggregate's departure rate

Fundamentally, the queue scheduling mechanism adopted to implement the EF PHB should guarantee a minimum departure rate ($AG_N\_DRt_{min}$) for a given aggregate (AG_N) mapped to EF behavior.

EF behavior implementation, for a given aggregate, has also to consider other aspects, such as:

- The aggregate's maximum arrival rate should be less than or equal to the aggregate's minimum departure rate in order to keep queue length empty as long as possible, leading to minimum latency, low jitter, and low losses.
- The queue scheduling mechanism used for EF PHB should behave invariantly with respect to traffic stream variation for any other aggregate.

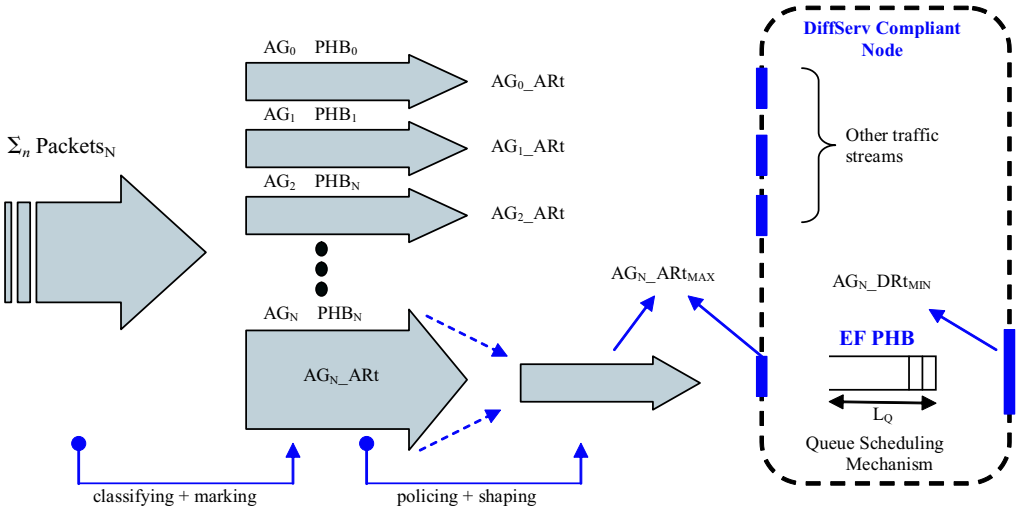

Figure 3.45 The expedited forwarding (EF) per-hop behavior (PHB).

Since various queue-scheduling alternatives (PQ, WRR, etc.) are possible to implement EF PHB, implementations should take the following into consideration:

- When an optimized use of network resources is intended, the maximum departure rate for a given aggregate should be kept as close as possible to the minimum guaranteed by the service:

$$AG_{n\_}DRt_{max} \cong AG_{n\_}DRt_{min}$$

- It is recommended that scheduling mechanisms that may cause "loss" to other aggregates be avoided.

The first consideration is related to the fairness of distribution for overprovisioned resources at each node. Network resources are scarce, and so the scheduling mechanism used should provide a way to distribute extra resources in a fair way. For instance, if bandwidth is available beyond the minimum required by all aggregates, this extra resource should be distributed in a fair and intelligent way.

3.6.7 Assured Forwarding

In general terms, assured forwarding (AF) services (AF PHB) offer a mechanism for service discrimination by creating "classes" for incoming traffic. In AF, each class corresponds to a certain level of service (greater or smaller probability of forwarding) that is scalable without the need for per-flow or signaling at every node.

In technical terms, AF PHB guarantees that packets belonging to a particular class have a defined probability of being forwarded. AF PHB services do not define any specific restriction on bandwidth or delay for packets. Instead, AF PHB distributes available resources in such a way that a relative probability exists for forwarding packets from different classes.

From the customers point of view, AF PHB services do not assure forwarding, but rather offer different levels of forwarding assurance for IP packets.

For ISPs or any service provider, offering service differentiation (services with different levels of forwarding assurance) is a flexible solution for billing and accounting. Customers can agree on the expected forwarded probability and be billed as such. Also, overbilling strategies for extra levels of guarantees are also possible to implement for customers.

AF PHB services are defined as follows:

- N = independent classes
- M = different levels of dropping precedence within each defined AF class

Each AF class is allocated a certain number of forwarding or networking resources at DiffServ nodes. All AF classes are independent, having their own forwarding resources. Within an AF class, drop precedence is mainly intended for deciding on packet priority during congestion. Once the network becomes congested, packet precedence will define which ones will be dropped first.

The standardized AF PHB is the following (IETF):

- 4 AF classes
- 3 drop precedences within each AF standardized class

DiffServ domains may define additional AF classes for local use.

The convention adopted for representing the classes and corresponding drop precedences is as follows:

$$AF_{IJ} \rightarrow \text{AF PHB class "I" with drop precedence "J."}$$

At this point, the next basic question is: How can AF classes and drop precedences be defined and implemented in DiffServ domains?

As stated earlier, first packets have to be conditioned at DiffServ network borders. Figure 3.46 represents the incoming stream of packets being conditioned for an AF PHB class "N".

The AF behavior states for the AF-defined class N that:

$$P_{F\_IN} > P_{F\_OUT}$$

$$P_{F\_OUT} \geq 0$$

where:

P_{F_IN} = forwarding probability for in-profile traffic
P_{F_OUT} = forwarding probability for out-of-profile traffic

Dropping precedence is configured by management within each AF class using microflow traffic parameters. Figure 3.47 illustrates dropping precedence configured for microflows within an AF class N.

Convention packets with smaller drop precedence values within an AF class are forwarded before packets with higher drop precedence values.

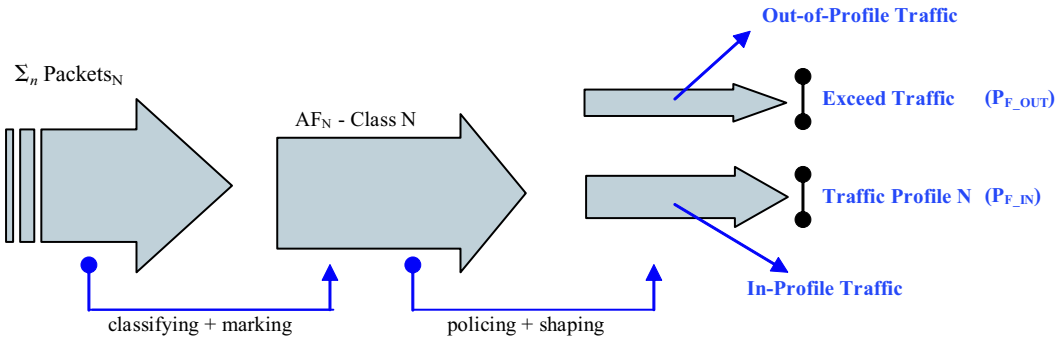

Figure 3.46 AF class conditioning.

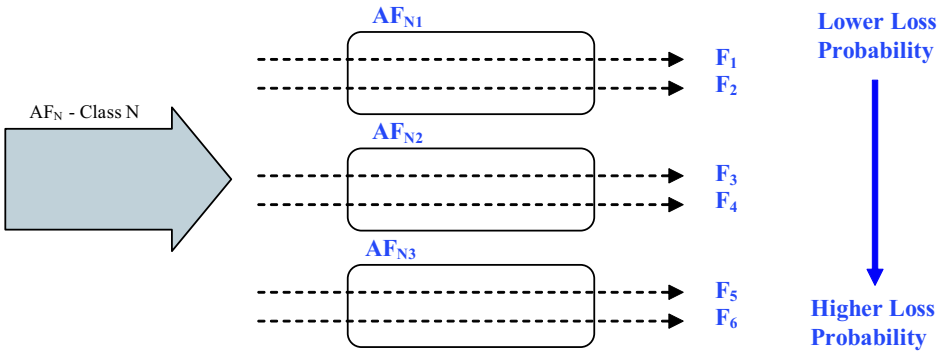

Figure 3.47 Dropping precedence configured for microflow traffic.

Figure 3.48 represents a generic view for an AF node in DiffServ domain having three classes implemented.

The number of forwarding resources allocated for each class is statically defined according to the scheduling mechanism chosen. As an example, percentage of available bandwidth or relative priority among AF classes are, among others, alternative ways for resource allocation.

Forwarding resources allocated for AF classes may not be in use and may be allocated to other AF classes. The utilization of these excess-forwarding resources results in higher efficiency for individual nodes, but has the drawback of being more complex.

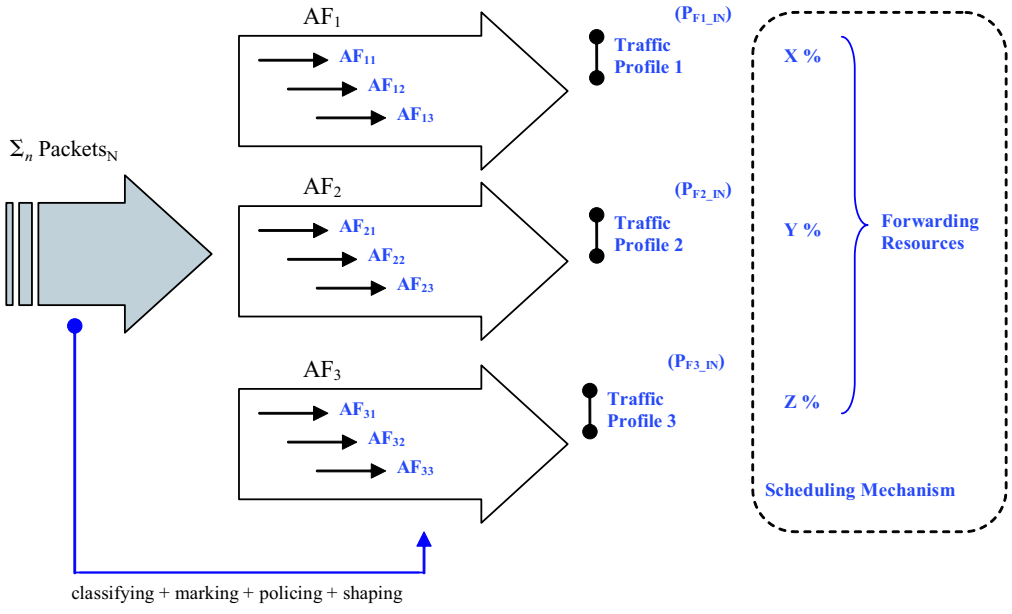

Figure 3.48 AF PHB with three classes.

3.6.8 Differentiated Service: Final Considerations

DiffServ architecture has important advantages. These include:

- Scalability
- Reduced packet-processing overhead

Scalability is one of the most important requirements for deploying any QoS strategy in complex networks like, for instance, the Internet. Scaling problems can be considered in two circumstances:

- Scaling to a large number of flows
- Scaling to high speed flows in routers and switches

Scaling to a large number of flows is a fundamental requirement of core routers in ISPs and complex networks. These routers frequently have to forward a large number of flows and, in this context, the differentiated service approach tries to minimize the amount of per-flow information required in routers by forwarding packet aggregates. The per-flow processing complexity is left to border routers, which normally handle a significantly smaller number of flows.

Scaling to high-speed flows is a requirement resulting from the evolution of technology. In effect, network interfaces are capable of very high sustained transmission rates (ATM, DWDM, SONET, etc.). In this scenario, applications will probably require high-speed flows with QoS to be maintained through the network, and simple DiffServ processing at core nodes may represent a considerable implementations advantage to guarantee these implementations.

As just mentioned, simplified processing is one major advantage for core routers in DiffServ. Since complexity is relegated to network borders, forwarding can be optimized, and this represents an implementation advantage with respect to other alternatives, in particular, with respect to IntServ.

DiffServ also has some potential problems. In particular, end-to-end behavior is not easily derived from the individual node's PHB. Also, specifying the distribution of network resources among nodes supporting various PHBs has potential difficulties. In particular, with both static and dynamic SLA requirements, there must be a global schema to allocate resources and to accept new flows or aggregates. This scheme presents challenging problems. One possible approach to deal with the allocation-of-resource problem and admission-control problem is to use bandwidth brokers.

As a final consideration, DiffServ is a network-oriented solution and is not the best solution for providing QoS requirements to microflows when they are needed for end users.

3.7 MULTIPROTOCOL LABEL SWITCHING

Multiprotocol label switching (MPLS) [9] is an initiative being proposed by IETF, which addresses many important requirements for today's network backbones.

In this section we elaborate on the importance of MPLS as a packet-switching technique for backbone networking. We will introduce its basic characteristics and operation,

focusing on the use of IP as the network protocol, although this technique can be applied to any network layer protocol (IP, IPX, among others).

3.7.1 Identifying Routing and Switching Basics and Possible Variations

Before presenting MPLS's operational principles, let us first identify the new technical aspects it introduces.

Routers and switches perform two basic functions: *forwarding* and *control*. Forwarding functions are performed for operations at level 3 (routing) or level 2 (switching) and for both connection-oriented and connectionless operation.

As an example, in routers the control function corresponds to the set of protocols and algorithms used to maintain the routing table. Route calculation and updating are performed by well-known protocols, such as OSPF, intermediate system to intermediate system (IS-IS), and BPG, that calculate paths and determine reachability. These protocols use different approaches with algorithms based on link-state or distance-vector information to define paths (routes) to destinations. Typically in routers, a user process executing on the equipment's processor performs route calculation.

The forwarding in routers is independent from route calculation. It corresponds to switching (routing) packets from input to output interfaces based on routing information available from the routing table. The forwarding function uses the routing table maintained by control protocols. The forwarding function in high-performance routers is typically executed by a piece of dedicated hardware (integrated function) as an attempt to increase the equipment's forwarding capacity.

In ATM switches the control function is also present. The private network-to-network interface (PNNI) is an example of a control function performed in ATM switches. The basic principle is about the same when compared with router operation. In effect, PNNI controls and establishes virtual connections between ATM users. Since the operation is connection-oriented, establishing virtual circuits (VCs) can take into consideration a great number of parameters and functionalities when choosing the path between ATM users. VCs in ATM switches are typically established by a user process executing on the equipment's processor.

The forwarding function in ATM switches is independent from VC control and establishment. In this function, ATM cells are switched between input and output ports based on previously assigned VC paths. This function is normally executed in hardware in order to achieve the high performance required by ATM connections.

MPLS introduces a slight variation of the technical solutions just discussed. In MPLS, the control and forward functions are performed according to the following schema:

- Routing protocols like OSPF and BGP are used in MPLS operation to calculate paths and establish reachability.
- Path control in MPLS follows a connection-oriented approach and uses a specific protocol (label distribution protocol (LDP)) to set adequately the "MPLS labels" along the path.
- As in routers and switches, MPLS forwarding is independent from control and is based on MPLS labels.

The interesting point introduced in this schema is: Since MPLS uses a specific protocol to distribute labels (LDP), its operation can be independent of the technology used for for-

warding. In other words, the control plane became independent from the type of forwarding being used. In brief, routers, ATM switches, and other technologies can be mixed in an MPLS network, providing a great degree of integration and flexibility. The next sections will explore some possible advantages and compromises resulting from this new technique for packet switching.

3.7.2 Goals and Basics

At first, it is important to observe that MPLS has been designed with its main target the large and complex networks typically found in ISPs and service providers. In other words, MPLS operation is supposed to scale and, as such, is intended to be an appropriate solution for backbones.

MPLS is a hybrid technique that allows very fast switching at the MPLS network core and conventional IP forwarding at the MPLS network borders. This hybrid solution has the global advantage of maintaining the basic IP operation encountered in many hosts, and also provides a generic way to achieve high-performance forwarding operation.

This high-performance forwarding operation can be achieved, for instance, by using ATM or frame relay switches at the network core and have their cross-connect tables "programmed" to switch according to IP routing requirements.

In brief, when considering IP routing, MPLS main advantages are:

- IP basic routing principles are preserved up to the MPLS network borders.
- Scaling and high-performance forwarding techniques are concentrated at the network core, which is suitable, for example, for ISP and network service providers.

MPLS supports many applications. These include traffic engineering (TE), virtual private networks (VPN), and tunneling applications.

7.7.3 MPLS Operation

MPLS basically assumes that the IP forwarding operation will be based on a "label." This will improve forwarding efficiency while allowing path setup to be defined in a more flexible way.

A simplified way to understand its basic operation is to describe MPLS as follows:

- Each MPLS node executes a routing protocol (OSPF, BGP, etc.) to discover paths (routes) and determine destination reachability.
- At the MPLS domain border, every incoming packet is associated with a *forward equivalent class* (FEC).
- Every defined FEC is associated with a route through the MPLS domain.
- At an MPLS domain border, packets also receive a *label* that assigns it to a specific FEC.
- Label information is distributed among the MPLS nodes before the forwarding process begins.
- Forwarding at the MPLS nodes is based on MPLS labels.
- Hop-by-hop forwarding using labels follows a label switched path (LSP).

- The forwarding process is repeated until the last MPLS node in the domain is encountered.
- When leaving the MPLS domain, packets are stripped of label information and are then forwarded by normal IP processing.

As this point, we observe that MPLS operation is based on definitions that must be examined in detail. These definitions are:

- FEC
- Labels
- LSPs
- LDP

We will discuss each one of these concepts in sequence, starting with the topology and terminology adopted by this packet-switching technique.

3.7.4 MPLS Domain

An MPLS domain is a set of nodes (routers and switches) running MPLS software for control and forwarding. MPLS domains are always within an administrative routing domain.

Figure 3.49 illustrates the typical nodes found in the MPLS topology and their terminology. Label edge routers (LERs) are devices operating at MPLS domain borders (Fig. 3.49). LERs can be further classified as "ingress" or "egress" devices. Ingress LERs are responsible for receiving and conditioning the packets entering the MPLS domain. Egress LERs are devices located at the MPLS domain exit points and are responsible for conditioning MPLS packets for normal IP processing. In brief, LERs are very important de-

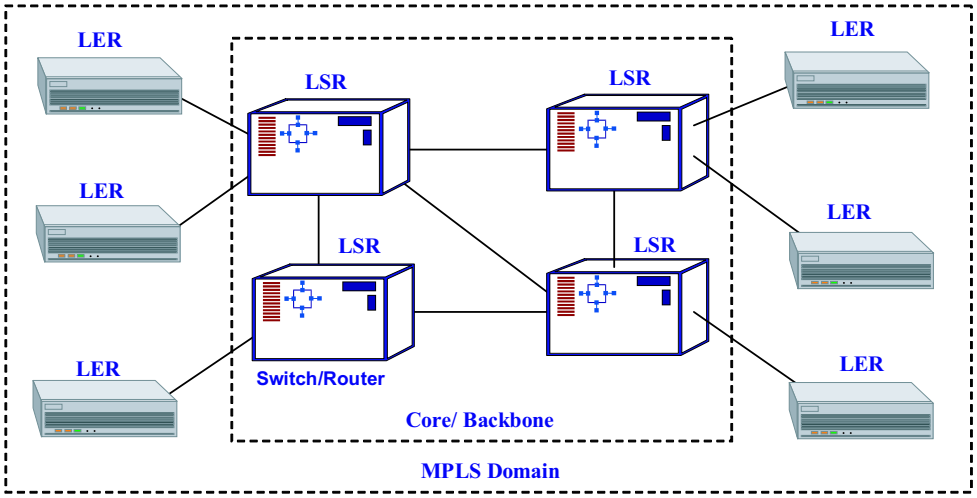

Figure 3.49 Nodes in MPLS topology.

vices that are responsible for interfacing MPLS domains with other MPLS domains or other dissimilar network.

The basic functions performed by a LER device are the following:

- To assign an FEC to packets (ingress LER).
- To assign labels to incoming packets (ingress LER).
- To participate in the establishment of "paths" (LSPs) for packets within the MPLS domain (Fig. 3.50).
- To strip labels from outgoing packets (egress LER).

Label switching routers (LSRs) are the core of the MPLS network. LSRs are typically high-performance devices (routers, ATM switches, etc.) optimized for forwarding processing.

The basic functions performed by an LSR device are the following:

- To participate in the establishment of "paths" (LSPs) for packets within the MPLS domain, in conjunction with others LSRs or LERs (Fig. 3.50).
- To "efficiently" forward labeled packets in order to achieve the best possible performance.

Figure 3.50 illustrates a path through an MPLS domain, named label switched path (LSP).

3.7.5 Forward Equivalence Class and MPLS Labels

In MPLS operation, packets are assigned to a forward equivalence class (FEC). Subsequently, the FEC is mapped to a next-hop MPLS node.

Packets assigned to the same FEC are treated indistinguishably within the MPLS do-

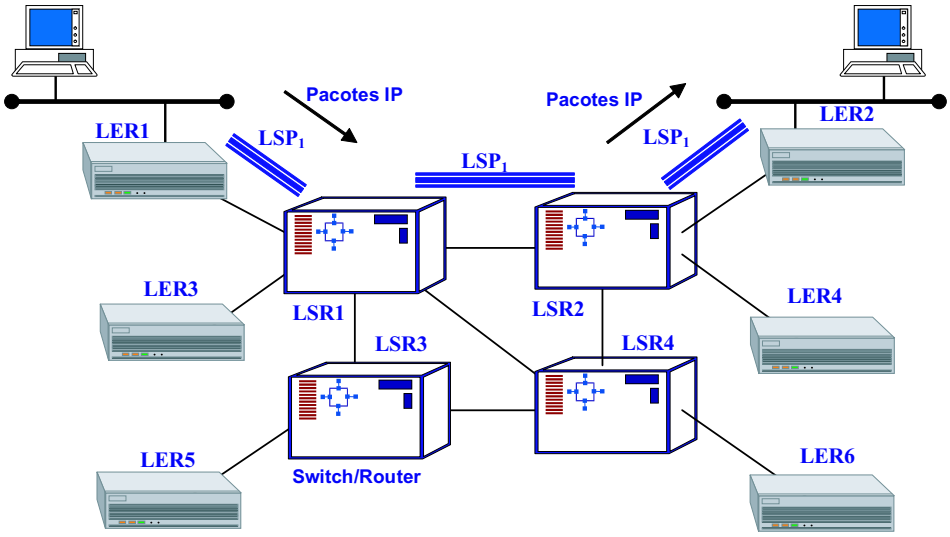

Figure 3.50 Label switched path (LSP) through a MPLS domain.

main. In other words, packets assigned to the same FEC will receive the same transport treatment at any MPLS node inside a domain independently of the packet's header contents (IP addresses, IP Precedence, etc.). This also means that, as far as their transport is concerned, packets assigned to the same FEC have, in principle, the same requirements.

Packets are assigned to an FEC only once at the MPLS domain edge by the ingress router (LER). Attaching a "label" to the packet effectively performs the packet-to-FEC mapping. Thus, by examining only the packet's label, an MPLS core node (LSR) is able to identify the FEC to which the packet is currently assigned.

Basic Principle: Packet → FEC → Label

Once packets are assigned to an FEC or, alternatively, once packets are labeled, they enter the MPLS network core and forwarding can be performed efficiently.

Each LSR device (MPLS core node) builds a table that specifies how packets are forwarded. This table is called label information base (LIB) and contains FEC-to-label bindings. Figure 3.51 illustrates LIB's contents and the next hop forwarding information used by the LSR device for a given labeled packet.

As mentioned before, the label is an identifier used to identify the packet's associated FEC. Labels have local significance. In other words, neighboring LSRs must agree on using a particular label to represent a specific FEC between them. This process is supported by the LDP, and will be discussed next. Once a label is chosen to represent an FEC between two LSRs, it has significance only to these LSR devices.

The forwarding processing is performed efficiently by using the packet's label as an index to LIB in order to identify the next hop node. No further analysis on packet's contents is executed inside the MPLS network core.

3.7.6 Label Switched Path

As illustrated in Figure 3.50, an LSP is equivalent to a path through the MPLS domain from the ingress LER to the egress LER. An LSP (route/path) has to be chosen to each FEC. In other words, associating an LSP to a particular FEC corresponds to the problem

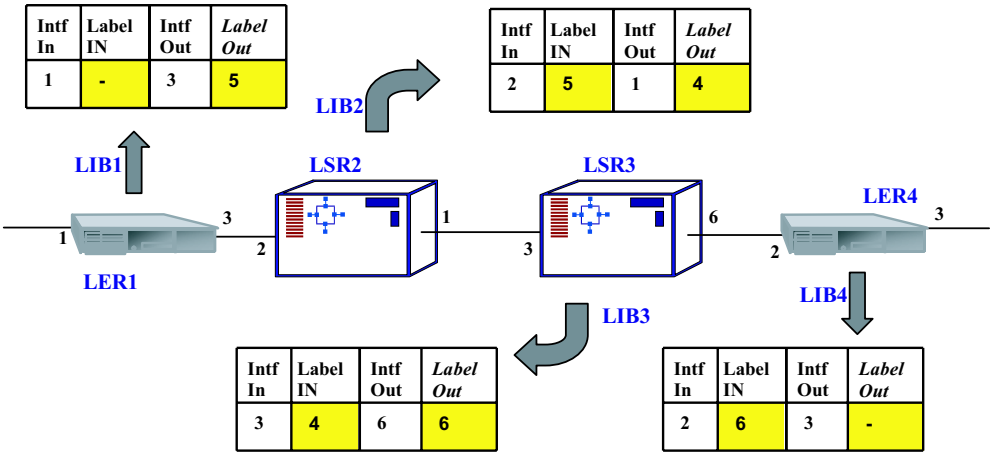

Figure 3.51 LIB and MPLS forwarding process.

of routing a set of labeled packets through the MPLS network. From an operational point of view, the LSP is set up prior to data transmission.

3.7.6.1 LSP-to-Label Assignment: Basic Principle In MPLS, the decision to bind a label "L" to a particular LSP (Figure 3.51) is made by the downstream MPLS node (Figure 3.52). Label bindings are propagated from downstream nodes to upstream nodes to establish an LSP through the MPLS domain. In other words, labels are "downstream-assigned" and their distribution (LDP operation) is executed in the upstream direction.

3.7.6.2 FEC-to-LSP Assignment Approaches There are two basic approaches to LSP setups (route selection) for a particular FEC in MPLS domains:

- Hop-by-hop routing and
- Explicit routing

In hop-by-hop routing, each LSR chooses the next hop (LSP) for a given FEC. This approach is equivalent to normal IP operation and each node chooses the next hop independently. Hop-by-hop routing uses conventional protocols such as OSPF and BGP for routing information decisions.

In explicit routing, an MPLS node (typically, the ingress router) specifies completely or partially the path followed by the packet in the explicit-routed LSP (ER LSP).

When the path is completely specified by the MPLS node, the LSP is "strictly" explicitly routed. Otherwise, the LSP is said to be "loosely" explicitly routed.

This approach is equivalent to the source routing operation previewed in IP options. In contrast to IP source routing operation, the packets are not flagged as source routed and do not carry the path information in the header. The explicit route is set up by FEC to LSP

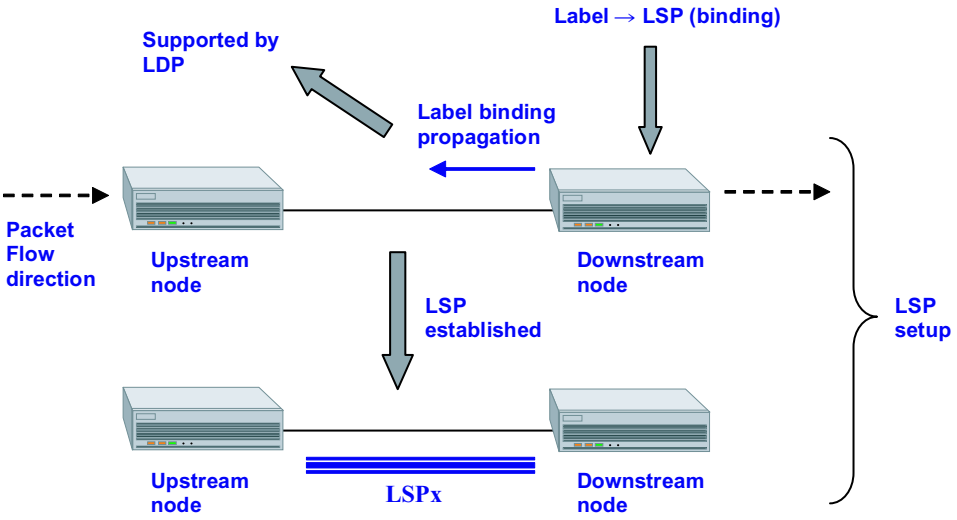

Figure 3.52 Downstream to upstream label assignment and distribution.

assignment supported by the LDP prior to packet's flow. Once established, the label assigned to the packet represents the ER LSP. This procedure allows complex computations to be realized in order to establish ER LSP. Besides this, ER LSP eliminates the actual problems found in normal IP source route operation, such as security block, header overhead, and processing overhead in high-performance routers.

Explicit routing allows the packet to follow an explicitly chosen route, which may be defined by a earlier configuration or may be defined dynamically. This approach has advantages, such as:

- The policy can be used to define routes through the network
- Traffic engineering approaches are supported

For example, nodes can make use of topological information to compute the optimal path (from ingress to egress node) through the network for quality of service or traffic engineering purposes. As another example, an explicit route can also be dynamically determined by constraint-based routing protocols such as the Constraint-based Routing LDP (CR-LDP).

3.7.7 Packet-to-FEC Assignment

A packet must be assigned to an FEC, for instance, at an MPLS network entry point (ingress LER). This assignment is realized by augmenting network layer packets with "label stacks." The packet is thereafter known as a "labeled packet."

An FEC corresponds to an aggregate of packets, all having, in principle, the same transport network requirements. In other words, once a packet is assigned to an FEC (receives a label), it will follow a specific LSP through the MPLS network.

A packet's assignment to FEC may be based on various criteria. Some of these are:

- IP addresses: complete address, prefix address, destination address, and source address
- Incoming interface
- Traffic engineering decisions
- VPN packets
- Filter based on QoS parameters (priority, ToS, IP precedence, etc.)
- Filter based on port and/or protocol, among other possible header parameters

A packet's assignment to FEC granularity may be coarse or fine grained. The level of granularity is, as far as the MPLS basic principles are concerned, an operational decision defined by operations management considering the processing resources available at routers. Packet assignment to an FEC in MPLS can be performed up to the microflow levels.

With respect to MPLS operation, a packet is initially assigned to an FEC and receives a label. In MPLS, FECs are mapped to labels, which are used at each LER/LSR to identify the packet's path (LSP) (Fig. 3.53) though the MPLS network. Thus, the packet assigned label is, in effect, that corresponding to the FEC chosen for the incoming packet.

As a final point, the FEC can also be understood as a specification indicating which packets are to be mapped to a specific LSP.

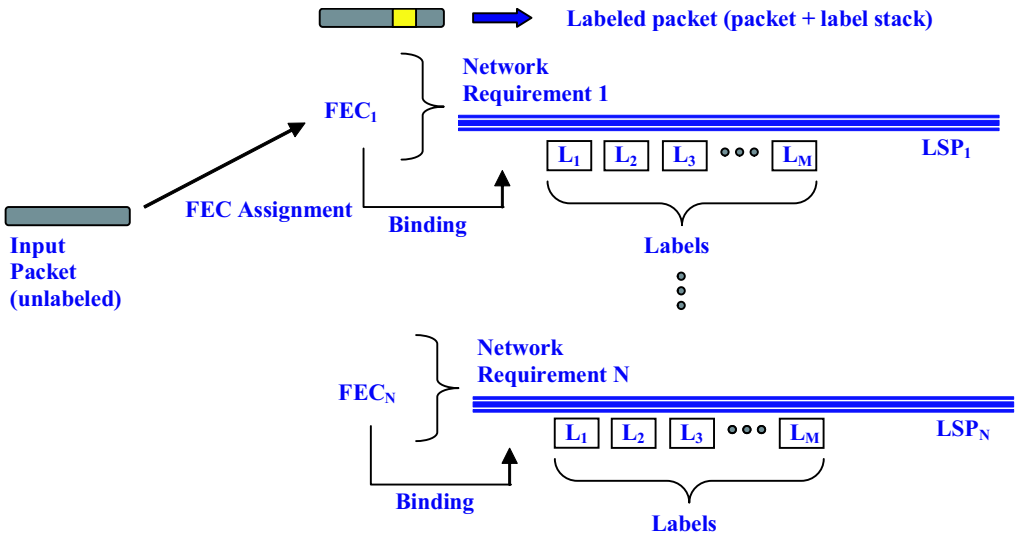

Figure 3.53 FEC and label assignments.

3.7.8 Label Distribution

MPLS architecture allows alternative methods for distributing labels among nodes and to manipulate them internally at each node. These alternative methods are used in the *hop-by-hop routing* approach, providing flexible MPLS operation and extended MPLS applicability.

First of all, there are two approaches for *FEC-to-label binding distribution.* Label bindings can be distributed for a particular FEC in two ways:

- Downstream-on-demand
- Unsolicited downstream

In downstream-on-demand, an upstream LSR explicitly requests its next hop (downstream LSR) for a particular FEC-to-label binding. In other words, the upstream LSR asks for the label, which it should use for a particular FEC (Fig. 3.54).

In unsolicited downstream (also called downstream unsolicited), a downstream LSR pushes an FEC-to-label binding to an upstream LSR (Fig. 3.54).

A second operational aspect of label distribution is when to advertise FEC-to-label binding. In MPLS architecture, two alternative methods are previewed to control label advertisement:

- *Independent* label distribution control
- *Ordered* label distribution control

In independent label distribution control, an LSR can advertise FEC-to-label mapping to its neighbors at any time (unconditional) (Fig. 3.55). In ordered label distribution control, a downstream LSR can advertise an FEC-to-label mapping only if the corresponding FEC

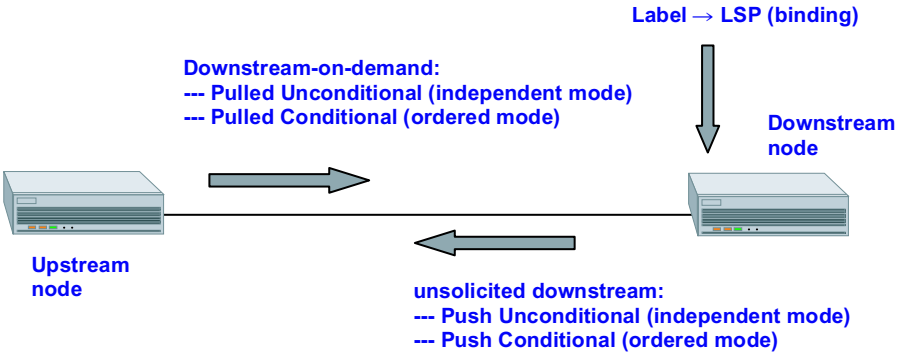

Figure 3.54 Label distribution methods: downstream-on-demand and unsolicited downstream.

already has an FEC-to-label mapping to the next hop (downstream) (Fig. 3.55). The ordered control method effectively forces the FEC-to-label biding advertisements to occur in order and sequentially, in the upstream direction, from egress node to ingress node. The independent control method, as the name suggests, allows FEC-to-label binding to be advertised in the upstream direction independently of any mapping to the same FEC in the downstream direction.

Another operational aspect concerning labels is whether each LSR should retain learned FEC-to-label binding for an LSR that is not the next hop for the FEC. In MPLS terminology, the label retention mode assumes two possibilities:

1. *Conservative* label retention mode
2. *Liberal* label retention mode

In conservative label retention, the LSR only keeps advertised label mappings, which will be used to forward packets. In this case, FEC-to-label mapping to routes, which are not used to forward packets, are discarded. The main disadvantage of conservative label-retention mode is that a new label mapping has to be obtained whenever a route change occurs. In liberal label retention mode, however, the LSR keeps all advertised la-

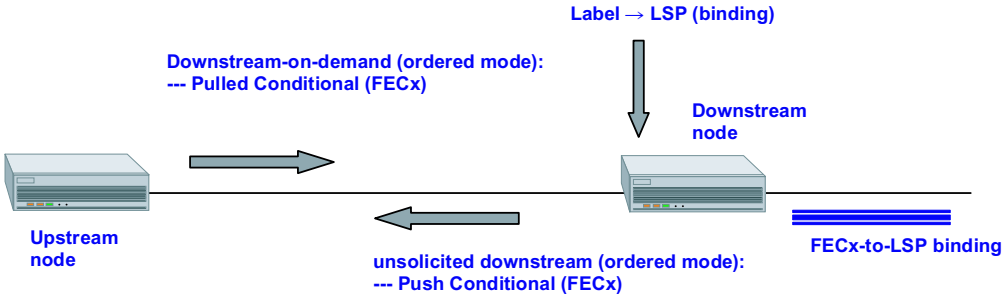

Figure 3.55 Label distribution methods: ordered control.

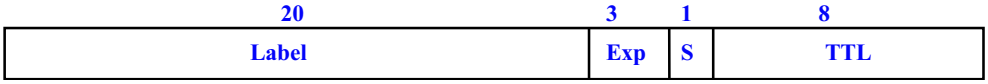

Figure 3.56 MPLS generic label encoding entry format (shim header).

bel mappings learned from its neighbors. When the LSR uses the downstream-on-demand method, the requesting LSR might choose the label mappings it requires for retention.

3.7.8.1 MPLS Label Stack In simple terms, a label identifies the path the packet should use within an MPLS domain. Packets in MPLS networks are labeled by augmenting them with a *label stack.* The label stack is a sequence of label entries whose general-purpose format is illustrated in Figure 3.56. In the figure, "Label" represents the label value that corresponds to the entry; "Exp" are bits left for experimental use; Bit "S" is used for stack operations, as described next; the TTL bits encode TTL values, which have the well-known "time-to-live" meaning, used in normal IP packet processing.

Label stacks are encoded ("shim headers") and carried with IP packets (Figure 3.57). A label stack is placed after the data link layer header and before the network layer header. The last label entry in the stack must have S bit set.

ATM, Frame Relay, LAN, and PPP Encapsulation Depending on the layer 2 technology used between MPLS devices, the label can be embedded (totally or partially) in the layer 2 headers. The basic idea is that when using technologies such as ATM or frame relay, their addresses (pair VPI/VCI and DLCI) could be inferred from the MPLS label or, alternatively, their addresses can be effectively used as MPLS labels.

For LANs and point-to-point protocol (PPP) encapsulation, there is a standardized encoding for the labeled packets (Fig. 3.58).

3.7.8.2 MPLS Label Stack Processing: Additional Considerations In the previous discussion, we have by simplification considered only single labels in MPLS labeled packets. Generically, labeled MPLS packets carry a label stack to support, among

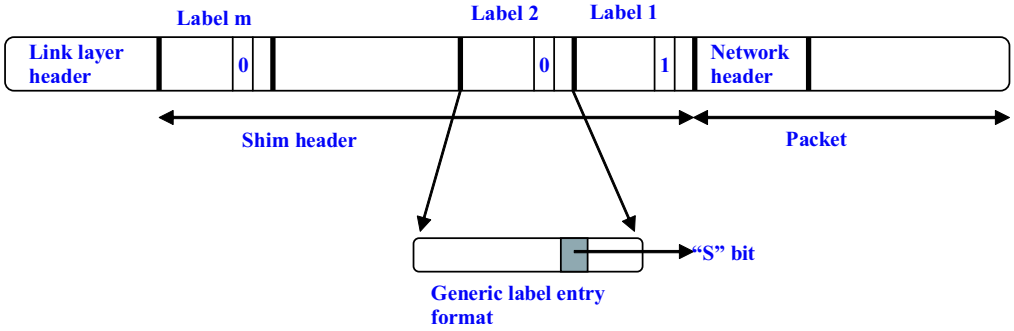

Figure 3.57 MPLS label stack carried with IP packets.

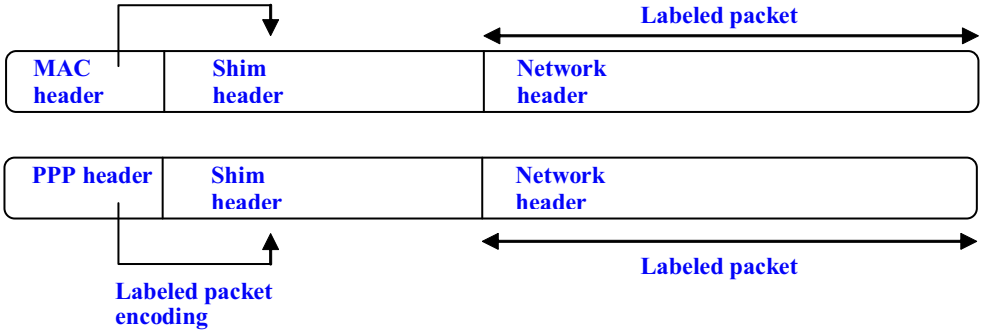

Figure 3.58 Labeled packet encapsulation.

other possibilities, LSP tunnels and MPLS hierarchy. The label stack processing follows the basic principles already defined and provides additional flexibility.

In brief, it follows the label stack processing for a labeled packet (Fig. 3.59):

- The top label is retrieved from the label stack (basic processing is always based on the top label).
- The top label is used to index the Incoming Label Map (ILM).
- The ILM points to the next hop label forwarding entry (NHLFE) entries at the LIB.
- NHLFE contains information required for label processing:
  —the stack operation to be executed (supporting hierarchy, tunnels, or conventional processing);
  —additional information, such as data-link encapsulation and label stack encoding.

Since the ILM points to multiple NHLFE entries, additional flexibility is incorporated to support, for instance, forwarding on balanced routes according to local policy management.

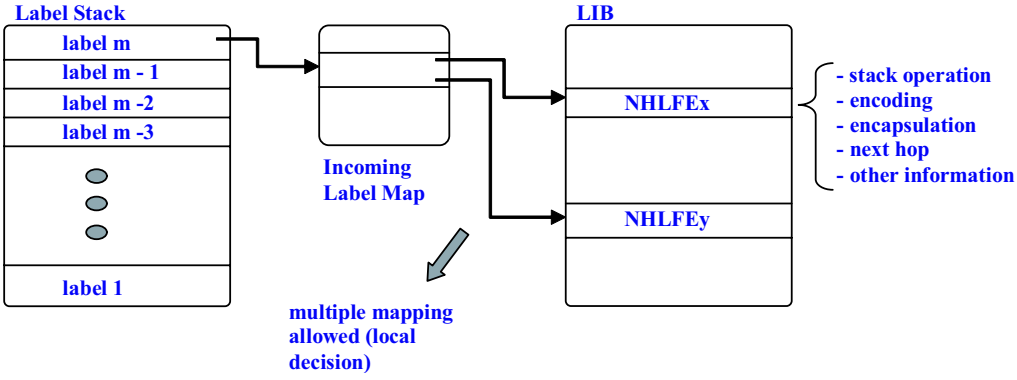

Figure 3.59 Label stack basic processing.

3.7.9 Label Distribution Protocols

Label information is distributed in MPLS domains using a protocol generically referred to as label distribution protocol (LDP). The following protocols are possible alternatives for label information distribution: LDP; RSVP; and BGP.

LDP [10] is a new standard protocol proposed by IETF for label distribution. In LDP, LSRs and LERs cooperate and exchange label information using LDP messages in a reliable LDP session. The LDP is discussed in the following section. Additionally, the constraint-based routing label distribution protocol (CR-LDP), an extension of LDP, is introduced in Section 3.7.10.

RSVP for traffic engineering (RSVP-TE) is an extension of the previously discussed resource reservation protocol, standardized by IETF for IntServ architecture, to distribute labels in MPLS domains.

With BGP, label information is exchanged, piggybacked on BGP messages.

3.7.9.1 Label Distribution Protocol As stated before, the LDP is a new protocol being defined by the IETF whose procedures and messages allow MPLS nodes to use network-layer routing information to establish LSPs. The LSPs are effectively created by *mapping* the *network-layer routing information* just mentioned to *data-link layer switched paths.*

The LDP protocol establishes reliable LDP sessions between LDP peers (MPLS nodes: LSRs and LERs) using TCP (Fig. 3.60).

In brief, LDP operation has the following phases:

- Discovery
- Session initialization
- Information exchange

In the discovery phase, "hello messages" are sent to well-known LDP port and router addresses (multicast router addresses) in order to announce neighborhood. Hello messages are periodically sent using UDP, since the basic idea is to allow the detection of potential LDP peers in the neighborhood. After the initialization phase, hello messages are also sent periodically to maintain LDP sessions as a keep-alive method.

During the session initialization phase, a TCP connection is established between LDP peers using session messages. An LDP session is then created and additional information messages could be exchanged using a reliable communication path.

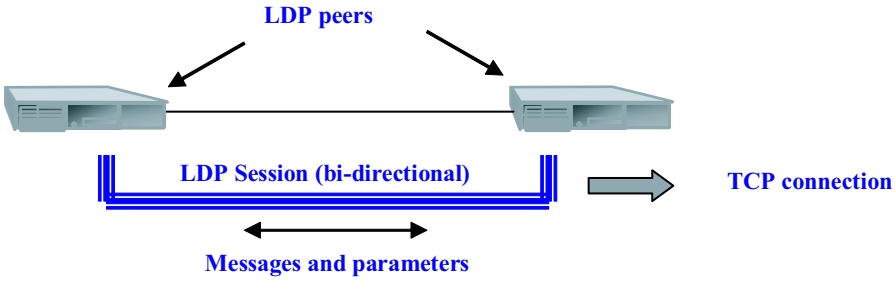

Figure 3.60 LDP basic operation.

During the information exchange phase, advertisement and notification messages are exchanged in order to create and maintain LSPs. Advertisement messages are used either to request a label or to advertise a label mapping. LDP uses these messages to create, delete, and change label mappings. As the name suggests, notification messages are used to announce additional information needed in LDP sessions or to carry error and notification information. Table 3.6 illustrates the general use of version 1 LDP protocol messages.

Table 3.6 LDP Message Types

LDP Message Type	General Use
Notification	LSR signals an event or provides information to LDP peers. Some examples are: • Fatal error (keep-alive timer expires, internal errors, etc.); and • Message formatting errors.
Hello	Supports the discovery mechanism used in LDP protocol to determine and maintain neighborhood.
Initialization	Supports session establishment between LDP peers. Examples of initialization parameters exchanged during the initialization phase are: • KeepAlive timer definition • The advertisement method used to advertise labels (unsolicited downstream or downstream on demand) • Loop detection enabling • Parameters to identify hello adjacency match • Additional parameters: ATM and frame relay session parameters (merge capability, address ranges, directionality, etc.)
KeepAlive	Supports the monitoring of LDP session integrity. KeepAlive message resets KeepAlive timer.
Address	Supports the advertisement of interface addresses. Activated interfaces should have their address advertised.
Address withdraw	Supports the deactivation of interfaces. Previously advertised address must be withdrawn whenever an LSR deactivates an interface.
Label mapping	Supports the distribution of FEC-to-label bindings to LDP peers. Label mapping messages are transmitted upon different conditions with the LSR configured as *independent distribution control* or *ordered distribution control* (Section 3.7.8).
Label request	Allows an upstream LSR to request an FEC-to-label biding (mapping). The basic parameter sent with a label request message is the FEC to which the mapping is requested. Optional parameters are a hop count (LSRs) and a path vector mainly used for loop detection.
Label abort request	This message is used to abort outstanding label request messages.
Label withdraw	Label withdraw message breaks previously defined FEC-to-label mapping. This message is used, for instance, by a downstream LSR to signal that a particular mapping should not be used.
Label release	The label release message is used by an LSR to signal to its LDP peer the release of a FEC-to-label biding. A label release message is sent, for instance, after the LSR receives a label withdraw message.

3.7.10 Constraint-Based Routing Label Distribution Protocol

Constraint-based routing (CR) is a model in which additional parameters (such as bandwidth, delay, QoS and explicit hops) are taken into consideration for packet forwarding decisions. In the CR model, the routes are typically computed at a single point, and both routing information and constraints must be distributed within the network.

The use of CR adequately supports the implementation of services and such features as:

- Load-balancing
- Traffic redirection
- VPN
- Rerouting under failure

In MPLS, the CR-LDP, an extension of the LDP protocol, supports the constraint-based routing model. CR-LDP has the same basic operation as was discussed for LDP. In addition, the CR-LDP protocol defines a mechanism, which sets up a constraint-based routing label switched path (CR-LSP) initiated at the ingress LSR and based on constraints. CR-LDP is mainly intended to support traffic engineering and, generically, to provide more flexible and robust routing decision to LDP, while the CR-LSP is an explicit routed path through the MPLS network, which is calculated at the MPLS network entry point (ingress LSR).

In a CR-LSP setup, an ER is calculated, taking not only routing information (routing tables) but also additional parameters into consideration. The idea is to consider setup characteristics and operational criteria for CR-LSP that accommodate, for instance, specific traffic and service demands.

The ER set up by CR-LDP can be strictly or loosely routed (Section 3.7.6.2). Loose ER represents flexibility for routing and rerouting. A loose ER segment allows for route definition at a higher level without specifying details. For instance, a loose ER segment could be set up automatically by LSRs based on level 3 routing information.

The LDP-CR protocol also provides information about traffic parameters that can be used for resource reservation at MPLS nodes.

3.7.10.1 CR-LDP Parameters CR-LDP basically extends LDP operation by including additional information for LSP setup. For instance, the label request in CR-LDP includes the FEC and two fundamental sets of parameters: the ER and the traffic parameters. ER parameter corresponds to the computed LSP, and the traffic parameters, as the name suggests, represent the traffic characterization. The list of parameters for label request is as follows:

- FEC (same as LDP)
- ER
- Traffic parameters
- Preemption
- Pinning option
- Resource class
- LSPID (the CR-LSP identifier)

Following is a brief discussion of their meaning and applicability.

Explicit Route The ER information specifies the path to be taken by the LSP being established. This information contains a list of nodes or group of nodes along the constraint-based path. This is so that after CR-LSP setup, it will traverse the entire group of nodes indicated, or a subset of it. This approach results in flexibility for fulfilling the request for constraint route decisions (Fig. 3.61).

Traffic Parameters and Resource Reservation The traffic parameters express the required traffic characteristics for the CR-LSP. The basic parameters are: PDR, PBS, CDR, CBS, and EBS. These parameters express traffic characteristics for the LSP. Generically, they can be used for various purposes:

- To reserve resources at MPLS nodes.
- To dynamically define routes.
- To condition traffic at network entry point.
- To signal traffic characteristics to nodes supporting, for instance, DiffServ implementations.

Path Preemption CR-LDP supports path preemption. For each hop, CR-LDP indicates the required resources. It can happen that these requirements are not fulfilled for a particular hop. CR-LDP can then reallocate paths or, in other words, to use path preemption. In path preemption, paths are reallocated. To support this facility, two priorities are defined:

- setupPriority (new CR-LSP)
- holdingPriority (existing CR-LSP)

Priorities vary from zero to seven ($0 \rightarrow 7$), and the preemption decision is based on the priority values for the existing and the new CR-LSP. This feature is very important for carriers and is a basic requirement of traffic engineering. Path preemption allows high-priority traffic LSPs to be established, even if there are not enough resources for both high- and low-priority traffic. For carriers, the preemption feature

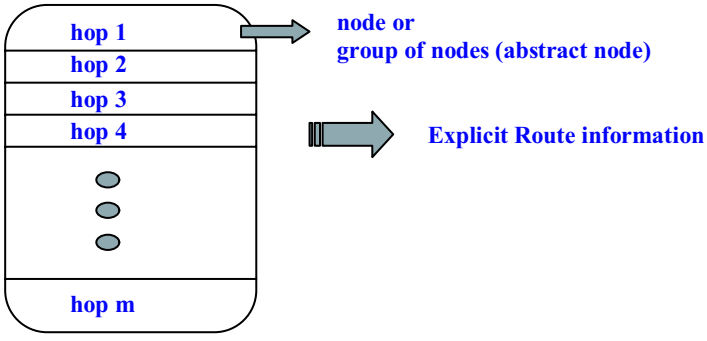

Figure 3.61 Explicit route information in CR-LDP.

is a valuable tool for network planners to assure that high-priority traffic LSPs will be set up under scarce network resource availability.

Pinning Mechanism CR-LDP also supports the route pinning feature. During CR-LSP setup the route pinning mechanism can be used to specify that the path should not be modified in case a better next hop becomes available. This prevents an LSR in the CR-LSP path from modifying its next hop, for instance, in loose routed paths. The pinning mechanism provides a method for preventing, for instance, route oscillations.

Additional Parameters CR-LDP also includes negotiation flags for each traffic parameter ("negotiable" and "not negotiable") and information concerning the service granularity ("frequency") over different time intervals. Another parameter ("weight"), weights the relative share of excess bandwidth above the committed rate. Another parameter, resource class, is a facility used to specify which links can be used by the CR-LSP. It is used to prune network topology.

3.7.10.2 CR-LDP and Traffic Engineering

Traffic engineering is one possible application of MPLS. Traffic engineering targets the efficient mapping of traffic onto a network topology. Efficient mapping means the optimization of the available network resources with respect to various criteria.

Traffic engineering has specific requirements that must be supported by the network in order to facilitate its implementation. These include:

- Adaptability to changes in network (topology, traffic load, and failure)
- Capability to support administrative policies
- Reliability
- Scalability

CR-LDP addresses these requirements, and therefore might be considered a protocol to support traffic engineering. In brief, the capabilities provided by CR-LDP are:

- Strict and loose path setup, which provides, among other possibilities, the means to support administrative policies, alternative paths, and network adaptation to changes.
- Path preemption that supports, among other possibilities, policy and network recovery procedures.
- Robust signaling by using TCP transport.
- A scalable and simple protocol with few control messages to establish, maintain, and release LSPs.

3.7.11 MPLS and ATM

MPLS can be "integrated" with ATM technology using three different approaches:

- Label-controlled ATM switching
- ATM tunneling
- Dual MPLS/ATM operation

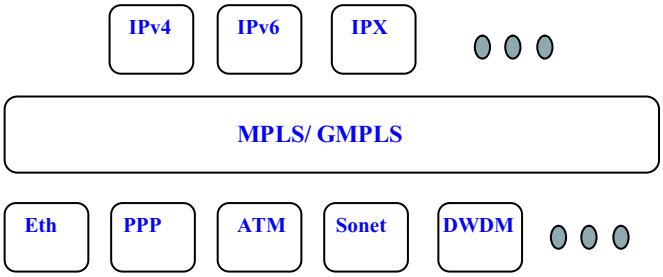

Figure 3.62 MPLS and GMPLS.

In the label-controlled ATM switching approach, ATM switches are used as LSRs. When using this approach, the ATM control software (UNI, PNNI, etc.) used by normal ATM switches to control its hardware is replaced by an IP/MPLS control software. ATM-LSRs can run network layer routing protocols and their data forwarding is based on layer 3 routes computed by these protocols. This approach therefore combines layer 3 routing with layer 2 ATM switching.

In the ATM tunneling approach, LSR devices use ATM switches as the interconnection technology between them.

In the dual MPLS/ATM operation, the ATM switch has simultaneous and independent control software for native ATM operation and IP/MPLS operation. This approach adequately supports the migration process to MPLS implementation.

3.7.12 MPLS and GMPLS: Final Considerations

As mentioned earlier, the MPLS technique can be applied to any network layer protocol (IP, IPX, and others). Besides this, MPLS can also be used with different layer 2 technologies like Ethernet, PPP, ATM, and optical network infrastructures based on SONET/SDH and DWDM (Figure 3.62).

In effect, one of the current motivations to adopt MPLS is its ability to control optical or transmission devices whose operation can be based on label swapping or label switching. The possible alternatives for applying this generalized MPLS (GMPLS) solution are numerous.

In these approaches, GMPLS and traffic engineering can be combined to control, among other devices, fiber switches, SONET and SDH switches and optical cross-connects. The IP packet forwarding processing optimization and the high throughput provided by these devices are combined to result in a robust QoS-ready multiservice network.

3.8 SUMMARY

Quality of service (QoS) approaches require both a basic background of associated principles and a general view of the alternatives available in order to adequately address the problem. In this chapter, QoS basics were initially approached by presenting the main QoS parameters and, also, by discussing the new router functionalities and how they inter-

relate in routers. Functions such as shaping, marking, token-bucket operation, and queuing are the basic blocks necessary for building any general QoS solution.

Based on the knowledge about basic router operation and building blocks, the standardized alternatives including IntServ, DiffServ, and MPLS, were presented. Here the main result obtained was an understanding of the architecture approach, the services previewed, as well as how router functionalities can be assembled to construct a more general solution.

The chapter then addressed an introductory approach to understanding QoS by selecting the relevant basic topics and interrelating them in order to provide a more general understanding of the subject.

REFERENCES

1. James D. McCabe, *Practical Computer Network Analysis and Design,* Morgan Kaufmann Publishers, Inc., Series in Networking, 1998
2. S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," *IEEE/ACM Transactions on Networking,* vol. 1, no. 4, pp. 397–413, August 1993.
3. W. Stevens, "TCP Slow Start, Congestion Avoidance, Fast Retransmit and Fast Recovery Algorithms," *RFC 2001,* January 1997.
4. M. Christiansen *et al.,* "Tuning RED for Web Traffic," *IEEE/ACM Transactions on Networking,* vol. 9, no. 3, pp. 249–264, June 2001.
5. B. Braden *et al.,* "Recommendations on Queue Management and Congestion Avoidance in the Internet," *RFC 2309,* April 1998.
6. R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: An Overview," June 1994.
7. R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource Reservation Protocol (RSVP)—Version 1—Functional Specification," *RFC 2205,* September 1997.
8. S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An Architecture for Differentiated Services," *RFC 2475,* December 1998.
9. E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture," *RFC 3031,* January 2001.
10. L. Anderson *et al.,* "LDP Specification, *RFC 3036,* January 2001.