1

Motivation and Basics

Torsten Braun and Thomas Staub

Summary. This chapter provides an introduction to the topic of Quality of Service and motivates the rest of the chapters in the book. The performance of network applications (e.g., video on demand, collaboration tools, voice over IP, Internet TV, video conferencing) depends on the quality of network connections. Parameters such as packet loss, delay, delay variation, and out-of-order delivery are important to describe network performance. Since applications differ in their Quality-of-Service (QoS) needs, this chapter provides a classification of some typical applications and sheds light on their requirements. It further illustrates the implementation and performance of QoS-aware applications, as well as the benefits of such QoS-aware applications, and concludes with a short overview of the following chapters.

1.1 Quality of Service and its Parameters

Quality of Service (QoS) is a measure of the ability of network and computing systems to provide different levels of services to selected applications and associated network flows. Since Internet Protocol (IP) based networks are expected to form the basis for all kinds of future communication services such as data transfer, telephony, television, etc., and users expect at least the same quality for those services such as when delivered over dedicated networks, QoS support for IP networks is urgently required. Currently, however, IP networks are Best-Effort networks. As the name suggests, packet forwarding is performed with the best effort, but without guaranteeing bandwidth, delay bounds etc.

Before going into more detail, the term QoS must be defined more accurately. [1] distinguishes perceptual, application, system, network and device QoS [2] for multimedia systems. The network QoS parameters are most important for this book. They may be specified in terms of the network load including packet interarrival times, burstiness and packet sizes, as well as the network performance describing network service guarantees. Network performance can be described in more detail by several parameters such as delay, delay variation, bandwidth, and packet loss rate. These are the basic QoS parameters discussed hereafter. QoS supporting systems try

to guarantee QoS by not exceeding QoS parameter limits. QoS parameters are also called QoS metrics in relation to measurement, see Chap. 2.

1.1.1 Delay and Delay Variations in End-to-End Packet Delivery

As described more formally in Chap. 2, the one-way delay (only called delay in this chapter) normally describes the average delay that packets experience over a specific connection. Packet delays can be split into four components:

- *Processing delay* is the time needed by network elements such as routers or end systems to process a packet. It depends on the processing speed of the network element hardware and the complexity of the functions to perform. These range from simple packet classification for forwarding and fire-walling, to complex payload modifications for encryption and content adaptation.
- Network components normally have input and/or output queues. The time a packet resides in these queues is called *queuing delay*. Queues become larger when the network becomes congested, which results in a longer queuing delay.
- *Transmission delay* is the time needed to transmit a packet at a specific bit rate. It can be calculated as

$$\text{transmission delay} = \frac{\text{number of bits to transmit}}{\text{transmission rate}}.$$

- The *propagation delay* describes the time needed by the signals to travel (propagate) through the medium. It can be calculated as

$$\text{propagation delay} = \frac{\text{physical distance}}{\text{propagation velocity}}.$$

Propagation and queuing delay are the key contributors to delay as long as no heavy processing like encryption or packetisation by applications (cf. Sect. 1.2.2.1) is needed.

In real-world networks, packets experience a delay on their path from the sender to the receiver, which is not constant but rather varying over time, because conditions on a route and the involved systems change. This is a result of the fluctuation of Internet traffic and resulting queue sizes. The delay is bounded by a minimum and maximum delay. The difference between these bounds is called delay variation. In the remainder of the section we use the same definition for the delay variation as the ITU-T for the IPDV as outlined in Chap. 2. The typical behaviour for the delay of packets of a single flow in the Internet is depicted in Fig. 1.1. Since processing, transmission, and propagation delay normally do not change for a given route, the delay variation has its source in the varying queuing delay.

The delay variation can be compensated by buffering packets, either within the network elements (routers) or the receiving end systems. Since end-system memory is much cheaper than router memory, buffering in the end system is usually preferred. Figure 1.2 shows the concept of play-out buffering. In this example, we assume that a continuous stream of packets is sent with a difference of 160 ms between each packet. Each packet has a timestamp indicating its transmission time. The delay of

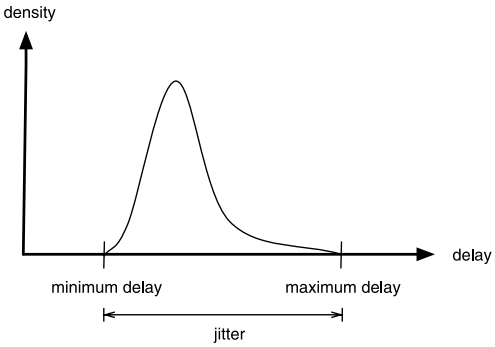

Fig. 1.1. Delay variation

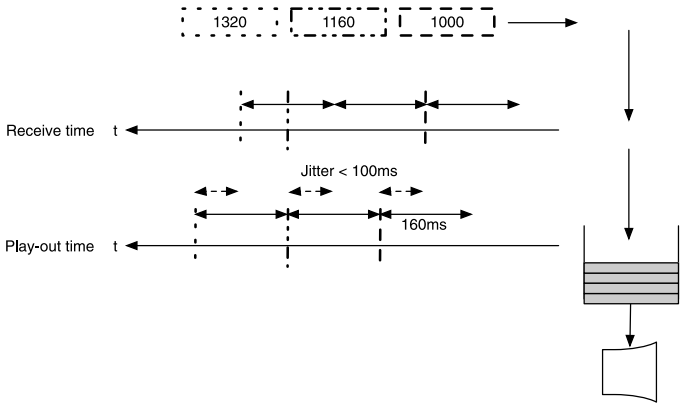

Fig. 1.2. Play-out buffer

the first and third packet is rather low (minimum delay), while the second packet experienced maximum delay. Assuming that the difference between the minimum and the maximum delay (delay variation) is not more than 100 ms, it is sufficient to delay each packet by 100 ms at the receiver. In this case, the first packet is delayed by additional 100 ms after reception and the second one arrives just in time so that the two packets can be played out with the original difference of 160 ms. The example shows that play-out buffering only works if the delay variation is bounded.

1.1.2 Bandwidth and Packet Loss Ratio

The bandwidth describes the capacity of a link or end-to-end path. It is measured in bits per second. The packet loss rate indicates the number of packets that do not reach the destination in relation to all sent packets. Packet loss has mainly two causes—packet errors, e.g. due to bad link quality (especially on wireless links) and packet drops, e.g. due to congestion. It can be calculated as

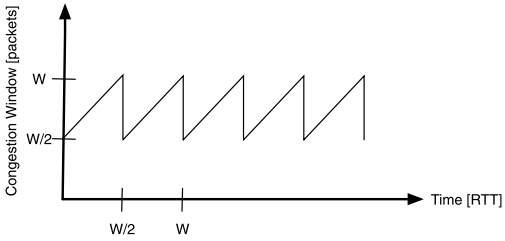

Fig. 1.3. Congestion window in TCP

$$\text{Packet loss ratio} = \frac{\text{packets sent} - \text{packets received}}{\text{packets sent}}.$$

Considering IP packets, there is no direct relation between bandwidth and packet loss ratio. However, in case of TCP connections, the achievable bandwidth depends on the round-trip time (RTT) and the packet error rate: $BW < \frac{MSS}{RTT} \times \frac{1}{\sqrt{p}}$ [3, 4] where BW is bandwidth; MSS is maximum segment size; RTT is round-trip time; and p is packet error rate. The achievable bandwidth can be calculated as follows. It is assumed that the delivery of $\frac{1}{p}$ packets is followed by a single packet loss, e.g. due to congestion. If the receiver acknowledges each packet, the window opens by 1 per round trip. With the maximum congestion window size W, and $\frac{W}{2}$ as the minimum congestion window in equilibrium, we get the behavior depicted in Fig. 1.3.

In each cycle ($= RTT \times \frac{W}{2}$) one packet is lost and the number of packets delivered is $(\frac{W}{2})^2 + \frac{1}{2}(\frac{W}{2})^2 = \frac{3}{8}W^2 = \frac{1}{p}$. The bandwidth BW is calculated as

$$\begin{aligned} BW &= \frac{\text{data per cycle}}{\text{time per cycle}} \\ &= \frac{MSS \times \frac{3}{8}W^2}{RTT \times \frac{W}{2}} \\ &= \frac{\frac{MSS}{p}}{RTT \times \sqrt{\frac{2}{3}p}} \\ &= \frac{MSS \times C}{RTT \times \sqrt{p}}, \quad C = \sqrt{1.5}. \end{aligned}$$

We conclude that the achievable bandwidth for a TCP connection depends on the round-trip time, as well as on the error rate.

1.2 Applications' QoS Requirements

Many network applications work fine with Best-Effort services, while others have strong QoS requirements and only work with guaranteed QoS, or at least benefit significantly if QoS guarantees are possible. After describing two classification schemes

of network applications, we give an overview of application requirements for audio, video and data applications.

1.2.1 Types of Network Applications

Each application probably has individual QoS requirements. However, they can be classified using different classification schemes. In the following we discuss two classification schemes for network applications, namely (in)elastic applications as well as (non-)interactive applications.

1.2.1.1 Elastic and Inelastic Applications

First, we distinguish between elastic and inelastic applications. An elastic application is able to adapt to changing QoS parameters and does not fail in that case. Elastic applications are also called Best-Effort applications. File transfer and e-mail are examples of elastic applications, because they do not require guaranteed bandwidth or delay bounds. In case of low bandwidth, the file or e-mail transfer just takes somewhat longer.

In contrast to elastic applications, inelastic applications need strict QoS guarantees. Real-time applications by nature are mostly inelastic, but may have some ability to adapt to certain QoS parameter changes. For example, audio/video conferencing can adapt to less bandwidth by using more efficient video codecs. A codec (derived from "coder/decoder") is a software or hardware device able to encode and decode a digital data stream. However, a minimum bandwidth is required to run the most efficient codec. Moreover, the delay requirements are quite stringent in such cases.

1.2.1.2 Interactive and Noninteractive Applications

Another classification scheme distinguishes interactive and noninteractive applications. Interactive applications include human interaction. Typically, a human user interacts remotely with another end system and expects a quick reaction to the performed action. The reaction should normally be as quick as possible with hard bounds of low delay. Due to the strict delay bounds, the error rate is quite important, because retransmissions are not possible without exceeding the tolerable delay if round-trip times are rather high. Forward error correction is a means to reduce delays in such a case, but additional processing is required by the end systems. Figure 1.4 shows the delay and error rate requirements for real-time voice transmissions. Interactive applications can also be categorised as real-time applications. In most cases, they are inelastic too.

Examples of interactive applications are voice over IP (VoIP), audio/video conferencing, collaborative online applications, and online games. Examples of noninteractive applications are Web browsing, file transfer, chats and multimedia streaming. For multimedia streaming a server begins to send a continuous stream of multimedia data to be played out at the receiver. In theory, insufficient bandwidth and

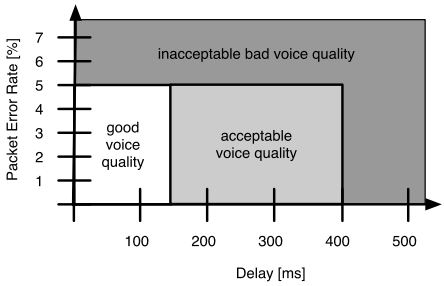

Fig. 1.4. Delay and error rate requirements for voice transmissions according to one-way values of [5]

delay variation can be compensated by buffering. If the bandwidth requirements are not met, the stream can only be played out after a significant delay introduced by buffering a huge amount of data, which can easily exceed available buffer space. As discussed in Sect. 1.1.1, large delay variation also has an impact on the required buffer size.

1.2.2 QoS Requirements of Applications

1.2.2.1 Audio Applications

Audio transmissions normally have widely varying bandwidth requirements, depending on whether telephony or high-fidelity music is being transferred. In addition to the encoded audio data, protocol overhead by IP, User Data Datagram (UDP), and Real-Time Transport Protocol (RTP) headers must be considered. Larger packets can reduce this overhead. In this case, several audio samples are collected and put together into a single packet, but longer packetisation intervals increase the delay. Moreover, sensitivity to packet loss is increased, since a single packet includes a rather long sequence of consecutive audio samples.

In particular, in case of interactive audio such as telephony, strong delay requirements exist. The recommended maximum tolerable delay for telephony is 150 ms [6]. This includes the four delay components described in Sect. 1.1.1 as well as packetisation delay. The delay variation should be limited too, since high delay variation increases the required size of play-out buffers at the receiver. Moreover, interactive audio applications are quite sensitive to packet loss. Thus, they require (very) low loss rates. The concrete numbers depend on the type of the audio application. Telephony using mother language requires less stringent error bounds than telephony using a foreign language or even high-fidelity music. Moreover, less efficient encodings have some degree of redundancy and can therefore tolerate higher packet loss.

In the case of streamed audio where a single user receives and listens to an audio stream without having interaction, the delay, bandwidth and error rate requirements can be relaxed. However, due to buffer limitations, guaranteeing QoS parameters might be desirable as well.

1.2.2.2 Video Applications

Many statements for audio apply to video transmissions as well. Audio and video transmissions have much in common. If interactive, both are sensitive to delay, delay variation and packet loss. However, there are several differences between audio and video. Most importantly, the required bandwidth for video is much higher, which depends highly on the quality level desired by the user or supported by the video equipment. While PC- or mobile hand-held-based video conferencing systems work with even a few tens of kbps, high-definition television demands several Mbps. The bandwidth requirements depend on several system parameters such as colour depth, screen size and resolution, frame rate and acceptable quality degradation by compression.

Another observation is that video traffic is usually burstier than audio traffic due to the used encoding schemes. Schemes like MPEG periodically send a so-called intracoded frame, which does not have any reference to other preceding or succeeding frames. Frames that are sent between these intracoded frames, so-called intercoded frames, can have a reference to other frames and only encode the difference (in terms of movement vectors, colour differences etc.) compared to the referred frames. This results in so-called weakly regular traffic for video, while audio often is strongly regular as depicted in Fig. 1.5. Weakly regular traffic creates some short-term bursts. The first option to handle such bursts is to provide sufficient resources (in particular bandwidth and buffer memory) in the network elements. The other option is to smooth out the traffic at the sender's end in order to produce rather constant traffic. However, this can again can lead to additional delays.

As for audio, the required video quality heavily depends on the type of the application scenario. A simple video conference with some known colleagues might have less stringent quality requirements than a movie or telemedicine applications. There is also a difference between stored and real-time video. If the video has been recorded in advance, video encoding can be more efficient resulting in higher burstiness, while live video compression may be less bursty due to the lack of processing time required for highly efficient interframe coding. The same as discussed for streamed audio applies to streamed video, but again, due to the higher bandwidth, buffer size requirements are much higher.

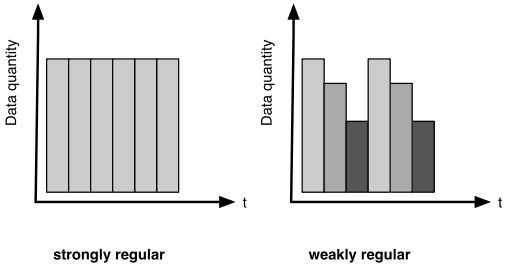

Fig. 1.5. Strongly and weakly regular traffic according to [1]

1.2.2.3 Data Traffic

Although nonaudio and nonvideo traffic is summarised as data traffic it must be kept in mind that there are many different data traffic classes. File transfers of multi-megabyte files as well as interactive console traffic is part of the data traffic category. The term transactional traffic is often used for interactive data traffic. Since there are thousands of data applications, it is impossible to discuss them all. Every application potentially has a unique traffic pattern and network requirements. Even different versions of the same application may result in very different traffic patterns [7].

1.3 Packet Scheduling in Network Elements

Sharing of network resources automatically introduces the problem of contention. Applications need QoS guarantees and congestion in networks is still common. Thus, scheduling disciplines implemented in network elements such as routers and switches are important so that network resources are shared fairly and performance guarantees for performance-critical applications. A scheduling discipline has to achieve two main tasks, as indicated by Fig. 1.6.

- First, it has to decide the order in which requests are serviced (packet selection).
- Second, it has to manage the service queue of requests awaiting service (packet dropping).

This section describes the concepts and requirements of a scheduling discipline and introduces some of the most important scheduling algorithms.

1.3.1 (Non)Work-Conserving Scheduling Disciplines

The most simple scheduling discipline is First Come First Serve (FCFS), also known as First In First Out. In this case, all arriving packets are served according to their arrival sequence. A scheduling discipline like FCFS is idle only if its queue is empty. Such schedulers are also called work-conserving [8]. On the other hand, a nonwork-conserving scheduling discipline may be idle even if it has packets to serve. A packet is sent only if it is eligible, otherwise it is delayed until it becomes eligible. However, why are nonwork-conserving scheduling disciplines useful? The reason is that downstream traffic can be made more predictable, and thus the buffer sizes and delay variation can be reduced in downstream routers and receiving systems. Bursts are eliminated and traffic becomes smoother. Of course, the mean queuing delay of a nonwork-conserving scheduling discipline is larger than with FCFS.

Fig. 1.6. Scheduling

The Conservation Law states that a work-conserving scheduling discipline can reduce a connection's mean delay, compared with FCFS, only at the expense of another connection. Thus, if a particular connection receives a lower delay than with FCFS, at least one other connection gets a higher delay. For that reason, the sum of delays with FCFS is a tight lower bound for the sum of delays for every scheduling discipline, whether it is work-conserving or not. The Conservation Law is as follows: $\sum \rho_i q_i = const$, with ρ_i as the mean utilisation of a link due to connection i and q_i as connection i's mean waiting time at the scheduler.

1.3.2 Fairness

Another important issue is fairness. A scheduling discipline should divide resources fairly among a set of users. A problem to be solved is when some users demand fewer resources than others do. So how can the resources left by these users be divided? The Max-Min Fair Share approach solves this problem by maximising the minimum share of a source whose demand is not fully satisfied. In this case, resources are allocated in the order of increasing demands. No user gets a resource share larger than its demand and sources with unsatisfied demand get an equal share of the resource. The resource allocation algorithm is as follows:

1. Divide capacity C by n: $\frac{C}{n}$ resources for each connection.
2. Connection 1 needs x_1 resources ($x_1 < \frac{C}{n}$).
3. Distribute exceeding resources $\frac{C}{n} - x_1$ equally among other connections, so that each connection gets $\frac{C}{n} + \frac{\frac{C}{n} - x_1}{(n-1)}$ resources allocated.
4. Continue the process if resource allocation is larger than x_2.

If we want to give a bigger share to some user than to others, we can use weights that reflect their relative resource share. The demands of the users are then normalised by the weight and sources with an unsatisfied demand get resource shares in proportion to their weights.

1.3.2.1 Requirements for Scheduling Disciplines

There are four (sometimes contradictory) requirements that a scheduling discipline must satisfy.

- Ease of implementation: In a high-speed network, once every few microseconds a server has to decide on which packet to pick for transmission. Therefore, a scheduling discipline should require only a few simple operations. Preferably, they should be implementable inexpensively in terms of hardware. Furthermore, the number of operations should be as independent of the number of scheduled connections as possible.
- Fairness and protection: An allocation at a network element can be considered as fair if it satisfies the max-min fair share criterion. For Best-Effort connections fairness is an intuitively desirable property. However, for guaranteed-service connections it is not a concern. A scheduling discipline provides protection if the

misbehaviour of one connection does not affect the performance of other connections.
- Performance bounds: A scheduling discipline should allow arbitrary connection performance bounds, limited by the conservation law only. Performance bounds can be defined in a deterministic or statistical way.
- Ease of efficiency and admission control: A scheduling discipline should be able to decide whether it is possible to meet the performance bounds of new connections or not. This decision is also called admission control. The decision should lead neither to network underutilisation, nor to jeopardising the performance of existing connections.

1.3.3 Scheduling Disciplines

In addition to FCFS, a variety of scheduling disciplines exist. An ideal and work-conserving scheduling discipline that provides a max-min fair allocation is Generalised Processor Sharing (GPS). Unfortunately, GPS cannot be implemented. GPS serves packets as if they are in separate logical queues. Each nonempty queue is visited in turn and an infinitesimally small amount of data in each queue is served. Thus, the scheduler can visit each queue at least once in any finite time interval. We assume N connections with equal weights send data to the scheduler infinitely fast. The GPS scheduler serves an infinitesimally small amount from each connection in turn. Therefore, each connection gets a share of $\frac{1}{N}$ of the bandwidth. If a connection sends less data than this share, the queue of this scheduler will occasionally be empty. Thus, the GPS scheduler skips empty queues, and because of its round-robin service the time saved is equally distributed to the other connections resulting in a new service rate. If another connection has a rate larger than $\frac{1}{N}$, but smaller than the new service rate, its queue will occasionally be empty too. Again, the remaining connections will receive a slightly larger share. Obviously, each connection with a rate smaller than its fair share gets its demand allocated, while each connection with a larger demand gets an equal share. Thus, we see that GPS service achieves the max-min fair share defined above.

A priority scheduler knows different priority levels, which have their own queue. Every incoming packet will be assigned to a priority level, depending on protocol type, application, IP addresses, etc. The packet with the highest priority will be processed. Packets with lower priority are selected only if there are no packets with higher priority available.

A (Weighted) Round-Robin scheduler has a queue for every service class and serves the packets from each nonempty buffer in turn. To obtain a service differentiation, service classes can have different weights so that the buffers will be served in proportion to their weight.

Weighted Fair Queuing is an emulation of GPS scheduling. For each incoming packet a finish number is calculated. The theoretical meaning of this finish number is the time the last bit of the packet should be transmitted if the GPS scheduler would be used. In practice, the finish number is only a service tag and does not stand for the actual time at which a packet is served. Packets are ordered by their finish number

and serviced in that order. The finish number of a packet arriving at an inactive connection is the sum of the current round number and the packet size in bits. If the packet is arriving at an active connection, the finish number is the sum of the largest finish number in the queue, or of the packet last served and the packet size in bits. The finish number is calculated as [8]:

$$F(i,k,t) = \max\{F(i,k-1,t), R(t)\} + P(i,k,t)/\phi(i), \quad \text{with}$$

$F(i,k,t)$: finish number of the kth packet of connection i with arriving time t
$P(i,k,t)$: packet length
$R(t)$: number of the round at time t
$\phi(i)$: weight of connection i.

The round number increases inversely proportional to the number of active connections. It indicates the number of rounds a bit-by-bit round robin scheduler has completed at a given time. A connection is active if the largest finish number in the connections queue or of the packet last served is bigger than the current round number.

Rate-controlled scheduling consists of two components: a regulator and a scheduler. The regulator determines the packets eligibility time and forwards only eligible packets to the scheduler. The scheduler uses an arbitrary algorithm (FIFO, Priority, Round Robin etc.) to schedule the packets.

1.3.4 Packet Dropping

The limited length of the various queues in a scheduler requires dropping packets in overload conditions. In order to avoid that important packets are dropped while less-important packets are not dropped, packets can be marked by applications or routers with a packet-drop priority. Packets with high drop priorities are dropped first. One approach could be to assign a low dropping priority to packets that have been travelling for a very long time. This avoids dropping packets that have already consumed a large amount of network resources.

Another issue is whether a scheduler drops packets when there is absolutely no space in the queue, or somewhat earlier to always have some space for important packets to serve. Alternatives are early dropping schemes such as Random Early Detection (RED). In this case, packets can be dropped even if the queue is not full. This always keeps some space for important packets arriving later. Those packets would otherwise have to be dropped.

When packets must be dropped, we can drop them from the tail of the queue. This is easy to implement, but may be unfair if packets of well-behaving connections have just arrived. An alternative is random dropping, where packets to be dropped are selected randomly. Even packets at the head of the queue can be dropped. This scheme has the advantage that the receiver will notify a packet loss earlier than for dropping packets at the tail. In this case, the congestion control mechanisms as implemented in TCP will react earlier.

1.4 Quality-of-Service Architectures

It is not sufficient to deploy routers with advanced packet schedulers to guarantee QoS between two computers in the Internet. End-to-end quality of service requires a system that is able to coordinate the routers in the network according to the users' demands. Several architectures have been proposed in the past to solve this problem. The following sections present two important cases, the Integrated Services and Differentiated Services architectures. We further discuss pure end-to-end approaches.

1.4.1 Integrated Services

The IETF's Integrated Services (IntServ) architecture (RFC 1633 [9]) was designed to overcome the inability of the Internet to provide guaranteed end-to-end QoS. It is based on the reservation of network and system resources (bandwidth, buffer, CPU time) depending on application requirements. In addition to the traditional Best-Effort service, IntServ provides the two new service classes: Controlled Load Service and Guaranteed Service. In order to use these services, the affected network elements must be configured by means of a signalling protocol. The most popular IntServ signalling protocol is the Resource Reservation Setup Protocol (RSVP).

Integrated Services relies on flow descriptors to characterise network traffic and reservation requests. Flow descriptors consist of a filter specification (FilterSpec) for identifying source(s) and destination(s) of the flow as well as a flow specification (FlowSpec) with information about the flow's traffic characteristics and the requested reservation. The abstraction of token buckets (RFC 2215 [10]) is used to describe the traffic characteristics of a flow. Token buckets are defined by the token rate r (in bytes/second) and the bucket size b (in bytes). See Fig. 1.7 for illustration. Packets will be sent (or forwarded) only if there are enough tokens in the bucket, which is refilled with r tokens per second. Flows that follow a token bucket model are predictable enough to allow for traffic guarantees. At the same time, they are flexible enough to cover most types of Internet flows. Optionally, three additional parameters may be used to make a flow even more predictable: the peak rate p (in bytes/second), the minimum and maximum packet sizes, m and M. These five parameters (r, b, p, m and M) make up an IntServ Traffic Specification (TSpec).

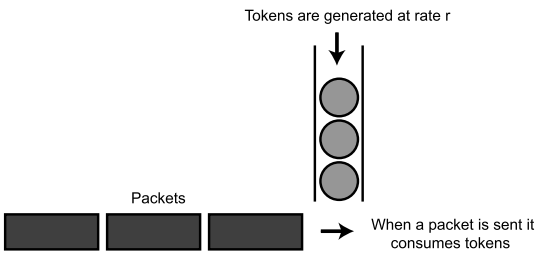

Fig. 1.7. The token bucket model

Most network applications can adapt to the relatively small variations in end-to-end latency and the occasional packet loss that occur when the network is not overloaded. However, when the network is congested their performance will degrade significantly. The Controlled Load service class (RFC 2211 [11]) aims to create such a congestion-less state. A flow can only enter the network if every network element on the flow's path has sufficient resources to satisfy the flow's traffic specification. Otherwise, the flow must be rejected. When a flow has been admitted, the network elements ensure that the flow complies with its traffic specification (this is called policing). The Controlled Load service only requires the basic token bucket parameters in traffic specifications. The Guaranteed Service class (RFC 2212 [12]) is aimed at applications with hard real-time requirements. It can provide a guaranteed upper bound for end-to-end latency, but it is also more expensive to implement. Each flow has to be handled separately and must be reshaped to maintain its characteristics.

The Resource Reservation Setup Protocol (RSVP) is the reservation protocol (RFC 2005 [13]) for IntServ architectures. It supports IP multicast and can be used without any changes to TCP and UDP. The two most important signalling messages in RSVP are PATH and RESV to establish sessions. The sender of a flow sends PATH messages along with the regular packets. This causes each router on the path to remember its previous hop, which allows any RESV messages travelling in the opposite direction to follow the same path. The RESV messages carry the actual reservation request (see Fig. 1.8). This approach has two advantages. First, each receiver can decide about the quality of service it wants to request. Second, reservations flowing upstream allow for elegant integration of multicast.

Another interesting aspect of RSVP is its use of soft states. Reservations must be refreshed in regular intervals to keep them active. This is done by resending the PATH and RESV messages, and reservation changes can be requested in the same manner. However, if a router does not receive any messages for a certain amount of time, the reservation will expire. This prevents "stale" entries in the routers' tables when end systems do not cancel their reservations. Nevertheless, reservations are made normally by the sender or by the receiver, using PATH_TEAR and RESV_TEAR messages, respectively.

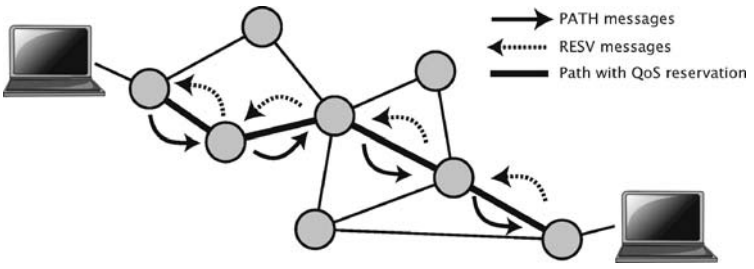

Fig. 1.8. Setting up a reservation with PATH and RESV messages

While RSVP has proved to be viable in small- to medium-sized networks, it has shown severe scalability problems in large-scale networks. The size of the flow state tables in RSVP routers tend to grow beyond manageable levels when faced with millions of users, each of which may have several active flows. Moreover, the overhead for a reservation may be too high for short-lived flows.

1.4.2 Differentiated Services

The Differentiated Services (DiffServ) approach aims to provide QoS support in large scale networks while avoiding the scalability problems of the Integrated Services (IntServ) concept. An Integrated Services router has to be able to distinguish between a potentially very large number of flows, resulting in very large flow state tables. The DiffServ architecture reduces this to a small number of aggregate flows. Furthermore, admission control or policing are no longer the responsibility of core routers. These changes make DiffServ much more scalable, but also less dynamic than IntServ.

Differentiated Services are built upon the concept of network domains. Usually, network domains correspond to a particular ISP network. Network operators negotiate contracts between each other, called Service Level Agreements (SLAs). SLAs contain Service Level Specifications (SLSs), which are the technical part of an SLA and define issues such as QoS parameters, encryption services, routing constraints etc. Several types of routers can be distinguished within a network:

- Core routers are inside a domain. They only forward packets according to the PHB (discussed later) are indicated in the packet headers.
- The responsibility of edge routers residing at the border to other domains is to police the incoming traffic (to control its conformance with the existing SLSs), and to make sure that the traffic leaving the domain does not violate the SLSs with the next ISP on the path. These routers are named ingress and egress router, respectively.
- First-hop routers are a special case of ingress routers. They are directly connected to an end user and mark the user's packets with codepoints according to the negotiated rules.

The level of QoS support that a flow receives depends on a six-bit identifier in the IP header; the so-called Differentiated Services codepoint (DSCP). When a core router receives a packet, it inspects the packet's codepoint to decide how the packet should be processed. This is referred to as Per Hop Behaviour (PHB) (RFC 2475 [14], RFC 2474 [15]). A codepoint of zero results in Best-Effort treatment, while other codepoints may be freely assigned by the network operator. Nevertheless, several default mappings have been defined by the IETF, namely for the Expedited and Assured Forwarding PHBs. This restriction to a small number of predefined PHBs frees core routers from the task of maintaining a (very large) table of flows. Instead, they only distinguish between a small number of macro-flows (or Behaviour Aggregates, BAs), i.e. the set of all flows that carry the same codepoint.

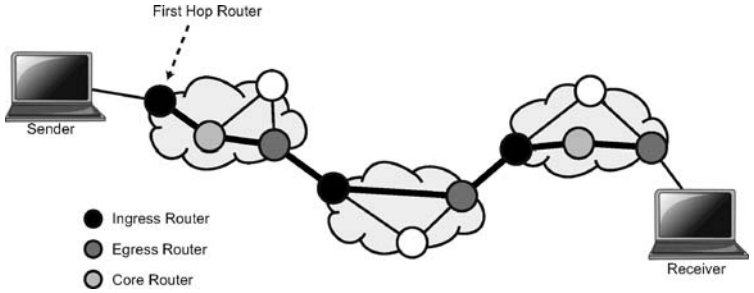

Fig. 1.9. The DiffServ view of the network

A customer would probably be interested in the end-to-end treatment of the packets, rather than in their hop-by-hop treatment. Per-domain behaviour (PDB) (RFC 3086 [16]) is the result of classification and traffic conditioning in edge routers, the PHB at interior routers, the traffic load and the network topology. A well-known example of a PDB is the "Virtual Wire" PDB [17], which is based on the "Expedited Forwarding" PHB (RFC 3246 [18]) and was intended as a circuit replacement over IP networks. PDBs can be used to implement different service classes. Creating such a service class (or Class of Service) requires conditioning the traffic aggregate so that its arrival rate at any node is always less than that node's configured minimum departure rate R. This is achieved by policing at the ingress and shaping at the egress router. The Expedited Forwarding PHB in interior routers must ensure that the traffic aggregate has a well-defined minimum departure rate R independent of the intensity of other traffic at the node. The Expedited Forwarding PHB can be implemented by priority or weighted round robin scheduling. The PDB for a given (aggregated) flow is defined in Service Level Specifications (SLSs).

In addition to the Expedited Forwarding PHB, the Assured Forwarding (AF) PHB group has been standardised (RFC2597 [19]). It provides delivery of IP packets based on four independent AF classes. Within each AF class, an IP packet can be assigned one of three different drop precedence levels. An IP packet belonging to AF class i with drop precedence j is marked with the AF codepoint AF_{ij}. A DiffServ node must allocate a minimum amount of forwarding resources to each AF class. Packets in one AF class must be forwarded independently from packets in another AF class. A DiffServ node does not reorder IP packets of the same flow if they belong to the same AF class.

PDBs and PHBs can be used to implement different service classes. A service class represents a set of traffic that requires specific delay, loss and jitter characteristics from the network (RFC 4594 [20]). A service class belongs to applications with similar characteristics and performance requirements, such as a non real-time service class for applications like file transfer, or a real-time service class for voice and other telephony services. Such a service class may be defined locally in a Differentiated Services domain, or across multiple DiffServ domains, possibly extending end-to-end.

1.4.3 End-to-End QoS Mechanisms

Both Integrated and Differentiated Services require special functionality in the network beyond simple IP forwarding. This is often problematic for several reasons. Replacing existing routers is expensive and, furthermore, Internet Service Providers have proved to be reluctant to accept global changes to the IP protocol since this would require a global agreement between ISPs, which is hard to achieve.

There are several architectures to achieve end-to-end quality of service without any—or very little—support from the network. They commonly rely on implicit or explicit signalling between end systems and require end systems to back off if there is any risk of congestion in the network. Unfortunately, this usually cannot be enforced, which makes these approaches viable only in controlled environments.

1.4.3.1 Endpoint Admission Control

Many of the end-to-end QoS architectures belong to the group of Endpoint Admission Control schemes. The basic idea of these schemes is simple: Before a new flow can enter the network, it must send a stream of probing packets to the flow's destination. The rate of the stream should be equal to the peak rate of the flow. If the number of successfully received probes is sufficiently high (e.g., $\geq$99%), the flow may enter the network—otherwise, it must back off. The streams probe packets thus serve as an admission control mechanism that is able to work without any support from the underlying network. This approach results in a service similar to the Controlled Load service from the Integrated Services architecture. The system cannot guarantee that a specific packet reaches its destination. However, the admission control mechanism ensures that virtually all packets of accepted flows do reach their destination.

Several variants of this scheme have been proposed. They can be categorised by two criteria [21]:

First, the admission control procedure can either simply rely on the probing stream's loss ratio, or it can require a marking mechanism in the network (e.g., based on the explicit congestion notification (ECN) bit in the IP header (RFC 2481 [22]). Marking approaches are generally more accurate.

Second, probing traffic can be either transmitted with the same priority or service class as regular data traffic (this is called in-band probing), or it can be transmitted with a different one (out-of-band probing). For example, if the network supports two levels of service, already accepted flows can be sent with high priority in order to protect them from low-priority probing traffic.

These criteria sort into four categories: in-band dropping, out-of-band dropping, in-band marking, and out-of-band marking (RFC 2481 [22]). In all these variants, the user data transmission is admitted only if the fraction of lost/marked packets stays below a certain threshold.

Unfortunately, all these approaches share a common problem: The measurement phase before a flow can enter the network may take several seconds, which is clearly too long for most applications.

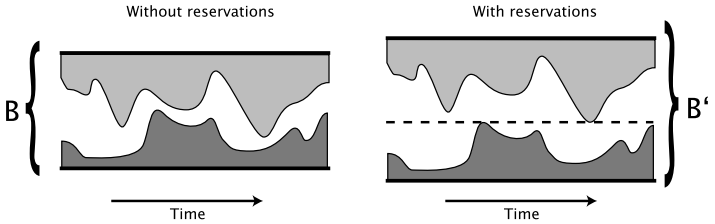

Fig. 1.10. The effect of statistical multiplexing: B is the link capacity, B’ is the link capacity necessary with peak rate reservations

1.4.3.2 Statistical Multiplexing and Egress Admission Control

A major advantage of measurement-based end-to-end schemes, as compared to reservation-based systems like Integrated Services, is an effect called statistical multiplexing. Many flows have varying bandwidth usage over time. For example, the bandwidth used by video transmissions often depends on the amount of movement in the encoded video. As a result, two flows may use the same link without packet loss even if the sum of their peak rates is larger than the available bandwidth on the link, and if their bursts (short intervals at which a flow sends at its peak rate) do not occur at the same time (see Fig. 1.10). Accordingly, the QoS experienced by a number of flows sharing the same link may be very good even if the sum of their peak rates far exceeds the link’s capacity. In contrast, reservation-based architectures usually allocate bandwidth at the peak bandwidth for every flow, which can result in very poor utilisation of the available link capacity.

1.4.3.3 Other End-to-End Approaches

A related approach without the need for active measurements is Egress Admission Control [23] that requires the edge routers of the network to perform admission control. They keep track of the amount of traffic leaving the network through them, and before a flow can enter the network, its traffic specification must be sent to its egress router. Based on this specification and the current traffic, the egress router will then make the admission control decision. In contrast to the endpoint admission control approaches, egress admission control requires ISPs to deploy new hardware in their network. Nevertheless, the necessary changes are kept to a minimum.

1.5 Implementation and Performance of QoS-aware Applications

1.5.1 Prerequisites for Successful QoS Applications

To support end-to-end QoS for network applications, the complete chain of involved systems and networks have to support it. Assuming that appropriate mechanisms exist on the network level, applications and the end system have to support QoS as

well. Ideally, applications make use of end-to-end signalling in order to signal QoS requirements and agreed QoS between each other. Signalling information should include flow and traffic descriptions, describing which packets shall receive what kind of treatment by the network elements. Applications also may explicitly mark packets to receive QoS support accordingly. Appropriate scheduling mechanisms in operating systems and end systems should be ideally available as well. If such signalling or marking cannot be implemented in the applications and on the end systems, corresponding policies might be defined and used by network elements to enforce proper packet treatment.

1.5.2 Media Scaling

An important concept usually implemented in multimedia applications is media scaling. This is particularly valuable when the underlying network cannot deliver the full load generated by the application. Media scaling can be realised in both transparent and nontransparent ways.

1.5.2.1 Transparent Scaling

In case of transparent media scaling, network elements drop packets from the media stream and deliver only those packets for which there is sufficient capacity in the network. Excess traffic will be dropped. Preferably, less-important packets are dropped and the most-important packets are delivered to achieve good perception quality at the user. Ideally, those important packets are marked by the application to allow the network to select those packets. Packet selection and dropping is performed in a transparent way for the application. A sophisticated example for media scaling is Receiver-Driven Layered Multicast (RLM) [24], where different layers of a media stream are split over different multicast groups. Receivers join the multicast groups as long as packet loss is below a certain threshold. If only the base layer is received, the media stream can be decoded with a base quality. Any other received layer improves the perception quality accordingly.

1.5.2.2 Nontransparent Scaling

Nontransparent scaling is not transparent to the application. Receivers or network elements return feedback about packet loss and delivery. The sender reacts to high packet loss by choosing more efficient media encodings, e.g. by using higher compression and/or lower media quality. In case packet loss is low, the media quality can be increased again. For example, for still image transfer a JPEG picture can be encoded with high-quality (1.638 bit per pixel, 96 KB, Fig. 1.11(a)) and significant low-quality settings (0.137 bit per pixel, 8 KB, Fig. 1.11(b)) for high and low network capacity, respectively.

(a) high quality (b) low quality

Fig. 1.11. JPEG pictures with different quality settings

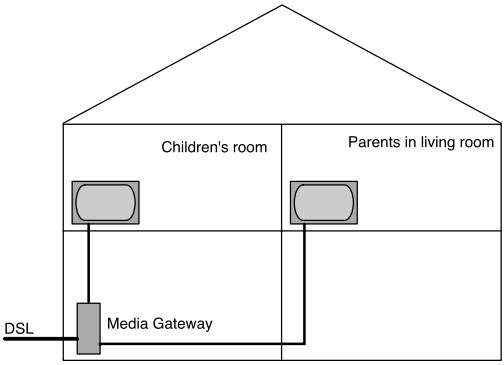

Fig. 1.12. QoS scenario

1.5.3 Applications' Performance Gain Due to QoS

The need for QoS in a scenario that is quite relevant for today is depicted in Fig. 1.12. Several persons living in a household share a single IP connection to an Internet Service Provider (ISP). The parents have subscribed to a high-definition TV service with additional monthly fees, while the children are enjoying Web 2.0 or Peer-to-Peer applications and are freely receiving streamed videos from some servers or peers. Unfortunately, the capacity of the access link is not sufficient for the simultaneous transfer of both video streams, see Fig. 1.13. However, if the video application displaying the high-definition TV program is linked with an end-to-end QoS system,

(a) High-definition stream (b) Freely received stream

Fig. 1.13. Two simultaneously received video streams without QoS support. (Elephants Dream, (c) copyright 2006, Blender Foundation/Netherlands Media Art Institute/www.elephantsdream.org, CC Attribution 2.5 and shuttle lift-off, NASA/nasa.gov)

(a) High-definition stream with QoS support (b) Freely received stream

Fig. 1.14. Video streams with and without QoS support. (Elephants Dream, (c) copyright 2006, Blender Foundation/Netherlands Media Art Institute / www.elephantsdream.org, CC Attribution 2.5 and shuttle lift-off, NASA/nasa.gov)

the situation might change as depicted in Fig. 1.14. In this case, the application requirements of the high-definition TV screen have been signalled to the QoS support system and network elements have been configured to give priority to this video stream. Consequently, the quality improves and packet loss is limited only to the freely received video stream, while the paying customers enjoy the QoS they expect for a pay service.

1.5.4 Summary

Despite high bandwidth availability in today's IP networks and techniques like adaptive applications, there is a strong demand for QoS support in IP-based communication systems. This demand is mainly coming from the fact that more and more applications have been moved from dedicated networks to IP based networks, in particular telephony and television. Important QoS parameters to be considered are bandwidth, delay, delay variation and error rates. Communication systems and networks must be designed to support QoS for certain network applications, in particular those where users are used to high quality and/or paying fees.

1.6 Structure of the Book

The following chapters introduce and explain mechanisms needed to design and implement a comprehensive QoS architecture for network applications running over IP networks. Chapter 2 presents methods for monitoring and measurement in IP-based networks. They are required for correct configuration of the network components and for traffic engineering (TE), which is the topic of Chapter 3. TE represents an important tool for providers to keep pace with the increasing traffic volume. Different TE techniques as well as the combination of TE and QoS are discussed. Chapter 4 covers signalling. The support of QoS over heterogeneous networks usually requires the establishment of sessions for the applications, e.g. video conferencing or video on demand. The sessions are set up using signalling and resource management. This book describes the signalling protocols Session Initiation Protocol (SIP), Next Steps In Signalling (NSIS) and Common Open Policy Service (COPS). They are used for application and network signalling. Chapter 5 is devoted to enhancements of transport protocols. Existing transport protocols offer only a limited set of services for the support of QoS. Enhanced transport protocols provide an adequate solution for providing soft QoS guarantees. The chapter shows the different mechanisms and their possibilities. Chapter 6 on the EuQoS system provides a case study based on the developments in the EuQoS project (End-to-End Quality of Service over Heterogeneous Networks) from the European Commission's Sixth Framework Programme. It shows the EuQoS architecture as one solution for providing end-to-end QoS. In the appendix, network simulation and emulation are introduced as two important techniques for network development and research.