CHAPTER 6

Traffic Handling and Network Management

This chapter discusses the constituent parts of Differentiated Services. In *Philosophical Investigations*, Ludwig Wittgenstein asked, "But what are the simple constituent parts of a chair?" Then he commented, "It makes no sense at all to speak absolutely of the 'simple parts of a chair'" (Wittgenstein 1953).

The definition of the leg, as a simple part of a chair, is better expressed by defining the leg's purpose rather than by describing the forms or characteristics of a leg. A leg of a chair can take many forms: It can be straight or crooked, wooden or metallic, red or blue, and so on. The purpose of a leg is to support something against gravity. The system of Differentiated Services is a logical structure—in the sense that the inherent logic of the system and the relationships of the various parts make the service differentiation, not the characteristics of the individual parts.

Nevertheless, the parts must be designed as well. As with a chair, after the manufacturer has clarified the purpose of the chair, appropriate form and material for the legs and back must be chosen. The main technical parts of the network service are as follows:

- Tools for traffic handling in boundary nodes
- Tools for traffic handling in interior nodes
- Network operation and management systems

This chapter discusses all these aspects.

6.1 Traffic Handling in Boundary Nodes

The role of boundary nodes as they relate to traffic management was discussed in the section "Traffic Classification and Conditioning" in Chapter 3, "Differentiated Services Working Group." This section now takes a closer look at the mechanisms needed for proper traffic control. There are four basic phases of traffic handling:

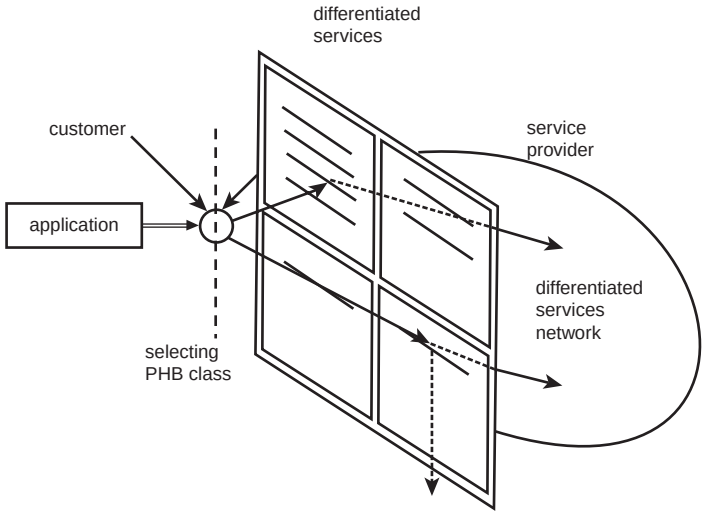
1. Setting the target, or goal
2. Collecting information
3. Making the decision
4. Executing the decision

Defining the target could be the most difficult task. You could say something general, such as that the result of traffic handling should be fair, efficient, and robust; or something specific, such as that each flow should obtain predefined quality requirements. Many important questions are situated in the middle ground between these extremes, actually they consist as both general and specific aspects. This chapter focuses on the aspects of the middle ground that may associate the general targets to specific mechanisms of Differentiated Services.

It is useful, once again, to recall that several service models with different requirements exist. Many, although not necessarily all, models can be mapped in the structure shown in Figure 6.1. A customer, or actually an application, sends a flow of packets to the network. Customers and applications do not, however, observe the network directly; instead, they observe a service structure specified by the service provider.

In a genuine Differentiated Services network (in the sense that not only the traffic control inside the network but also the customer service is based on the Differentiated Services model), service structure may be composed of a couple of service classes. The fundamental characteristic of a class is that it offers *reasonable* service for a packet flow, where reasonable means that someone has made a reasoned selection for each flow. But who makes that selection? The possibilities range from a pure application model in which the application automatically selects a preferable service class to a customer model in which the network selects an appropriate service class based on the customer contract regardless of the application.

Figure 6.1 Differentiated Services from a customer's perspective.

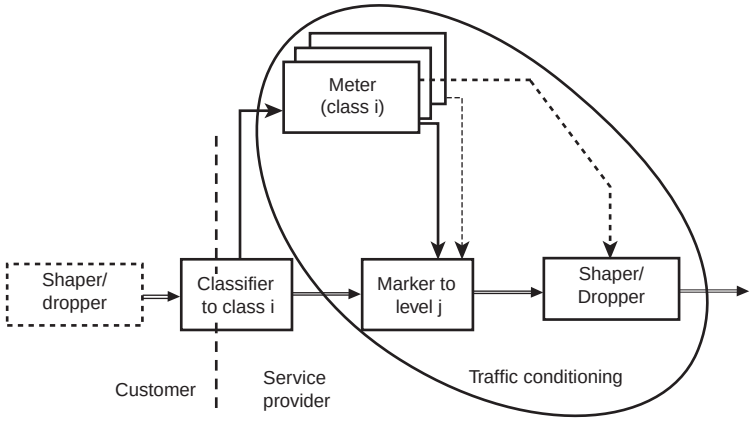


The overall system should allow simultaneous use of different models. Figure 6.1 presents an approach in which the PHB class is selected based primarily on the requirements of the application. Each class consists of several importance levels, emphasizing the need to have one unambiguous importance dimension inside the network, as described in section 4.5.2, "Importance," of Chapter 4, "General Framework for Differentiated Services."

Although the selection of the PHB class is based mainly on the requirements of the application, the selection of the importance level has more to do with the customer contract. The customer contract, or SLA (see the section "Architecture Model" in Chapter 3), defines the relationship between the incoming traffic stream and the importance level for every service class: The higher momentary load, the lower importance. Furthermore, packets belonging to the lowest importance level might be discarded immediately. Although this structure does not cover all possible service models, it forms a good basis for evaluating traffic-control mechanisms. It should also be noted that because the selection process can be dynamic, as described in the section "Service Models" in Chapter 3, this model is actually quite comprehensive.

The implementation of the system illustrated in the figure can be based on the model presented in the architecture document titled "An Architecture for Differentiated Services" (Black, Blake, Carlson, Davies, Wang, Weiss 1998). You can use Figure 3.8 in Chapter 3 with slight modifications. In Figure 6.2, *classification* means the selection of PHB class, and *marking* means the selection of the importance level. Note that here importance refers to the situation inside the network, and does not usually directly relate to the relative importance of different packets of one flow.

Figure 6.2 The main building blocks of traffic handling.



Moreover, based on the classification and marking result, the network can either delay some packets (that is, shape the traffic flow) or even drop packets. Nevertheless, it often seems more reasonable, for reasons discussed later in this chapter in section 6.1.4, "Traffic Shaping," to shape the traffic flow before metering and marking, perhaps in the customer's network.

In summary, the main elements of traffic handling in boundary nodes are classifier, meter, marker, shaper, and dropper. The main purpose of the whole system is to support the service model applied by the service provider, and therefore, the characteristics of the mechanisms cannot be assessed without defining the service model to some extent.

6.1.1 *Classifiers*

A *classifier* is a mechanism used to select the PHB class for a traffic flow. Although this is quite a straightforward task technically, some scalability problems could result if a large number of issues are taken into account. Because this question depends on the service model adopted by the service provider, it is necessary to refine the issue. The following models, differing in the classification method, can be identified:

1. The user selects a definite service class from the available classes.
2. The application automatically selects a preferable service class for each flow or packet.
3. The network selects an appropriate service class based on information about the application.

4. The network selects an appropriate service class based on the customer contract regardless of the application.
5. A combination of the first four approaches.

Note, in particular, that the classification should not depend on the result of traffic metering, because all packets belonging to a flow should be classified in the same class.

The main requirement of the first approach is that there should be a method to transmit classification information between customer and network. If the customer is not satisfied with one service class for all flows, a sufficient dynamics of signaling is required. This method, without additional mechanisms, does not allow the simultaneous use of several classes—for instance, one class for an IP telephony application and another one for data transfer.

Therefore, the second approach in which the application informs the network seems to be more practical. In a Differentiated Services framework, a sensible scheme is to use the same DS codepoints between the customer and network as inside the differentiated network. Then the classification problem can be solved in such a way that the application selects a DSCP codepoint that it expects to be the most appropriate for the application. A videoconference application, for example, may select the highest importance level of the real-time PHB class for all packets. Thus, if the customer is not allowed to use the selected class, the network operator can use the default service for those packets.

One problem with the second approach is that it requires the capability to set an appropriate DSCP codepoint in the customer's equipment. If that kind of mechanism is not available, the only reasonable choice is to classify the packets in the boundary node (third approach). It is likely that a standard mapping is available for all widespread applications.

It is also possible that each customer has access only to predefined PHB classes. In the extreme case (the fourth approach), each customer can use only one PHB class. Then the classification problem is trivial: All packets of one customer are classified in one PHB class.

Finally, it is possible to combine different approaches. For instance, the boundary node usually selects the PHB class based on the identified application, but if the application has already marked packets with a predefined DS codepoint, the boundary node does not make any change of codepoint.

6.1.2 Meters

The main purpose of traffic metering, with regard to Differentiated Services, is to make it possible to sort the classified packets into the right importance level. The primary approach is that the ensuing decisions (marking, shaping, and dropping) are based on the measuring

result of the class to which the packet belongs. It also is possible that the packet marking takes into account several measuring results. If there are two classes—real-time and data, for example—the user can either divide his capacity (permanently) between the classes or use the whole capacity continuously for both classes. In the latter case, the marking of each packet depends on the measuring result of both classes. Therefore, it is usually necessary to measure every PHB class separately. The rest of this section addresses the issue of traffic metering of one PHB class.

The Target of Metering

The combination of traffic metering and traffic marking is a very critical issue for the general characteristics of the network service. Therefore, it might be useful to recall the four aspects defined in Chapter 1, "The Target of Differentiated Services," used to assess the target of traffic metering:

- Versatility
- Cost-efficiency
- Fairness
- Robustness

Whatever method the service provider decides to use for measuring traffic, it has to work properly in all situations even with a mixture of strange traffic patterns. This does not mean, however, that the method should work optimally in every imaginable case.

Cost-efficiency dictates that the method should be as simple as possible, but with the appropriate level of flexibility. Even though someone may invent a theoretical model that promises a perfect result, it is useless if the model is too complicated for any practical implementation. For instance, a model with 10 different mechanisms for 10 time scales could be, in theory, much better than a scheme with only one or two mechanisms with four parameters. The simple model could be much more practical, however, if it provides a satisfactory result. Consequently, the primary goal is a simple, but versatile enough model.

The other two aspects, fairness and robustness, should be used to further assess the measuring methods. Again, it should be stressed that the overall target is the key. Simply put, the target is to provide a fair share of network resources in a robust manner.

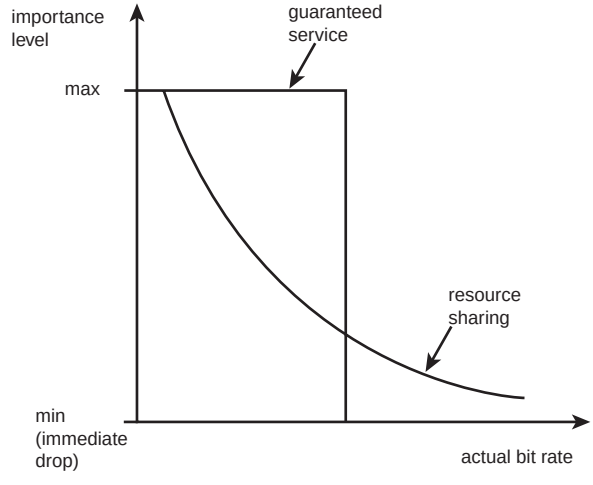
One interpretation of *fair share* is that for a given quadruple of {PHB class, importance level, point of time, link}, the price of a packet is approximately the same (here the term price should be understood very generally). Unfortunately, because of the inherent complexities of pricing issues, this interpretation is not such a clear statement as is it may appear. It might be clearer, therefore, to express the target in another way:

A fair share means that for a given quadruple of {PHB class, importance level, point of time, link}, the price of a packet reflects appropriately the service model applied by the service provider. (This statement ignores the fundamental question of what is a fair service model.)

You may even be able to make some inferences from this statement. Figure 6.3 shows two possible service models with essentially different targets. In the guaranteed-service model, the packets under a predefined bit rate obtain the best possible treatment (related to delay and loss probability); all excessive packets, on the other hand, are either marked with lowest importance or dropped immediately. In the resource-sharing model, a (virtually) continuous importance function is needed to inform the interior nodes about the relative amount of resources used by customers. Apparently, different characteristics are needed for these two service models.

In the guaranteed-service model (as well with several other service models), it is only necessary to know whether a limit is exceeded. In the case of the resource-sharing model, the metering gives information about the instantaneous bit rate, or more generally, about the amount of resources needed to transmit the packet flow through the network.

Figure 6.3 Desired measuring principle for guaranteed service and resource-sharing service.



Note also that not only bit rate, but also packet rate can sometimes be an important factor. For a given link speed, the number of packets that the router has to handle each second depends on the packet size: The smaller the packets, the harder the processing requirement. Therefore, it can sometimes be reasonable to give more weight to small packets by using an extra term (B_0), as shown in Formula 6.1.

Formula 6.1

$$B_j = c_M * B_{j,real} + (1 - c_M) * B_0$$

The component c_M is a constant between 0 and 1, $B_{j,real}$ is the real size of the IP packet j and B_0 is the "standard" packet size. If c_M is 1, the result is the real amount of bytes, and $c_M = 0$ means that only the number of packets is taken into account, not the packet sizes. Note that this is only one example of many possible schemes to take this requirement into account.

Measuring Principles

A long tradition of traffic measurements (well, at least *long* in a network-specific sense) means that there are many ways to manage traffic. This brief introduction to this complex issue concentrates on only a few typical models. As to the first goal (to evaluate whether a limit has been exceeded), the prevalent standard is to apply a token bucket scheme. See, for example, "General Characterization Parameters for Integrated Service Network Elements" (RFC 2215) and "An Architecture for Differentiated Services" (RFC 2475).

The Token Bucket Principle The basic principle of *token bucket* is that a bucket of capacity *b* is emptied by a constant rate of *r*. (Note that *leaky bucket* is often used as a synonym for token bucket.) The result of this formula measures whether the bucket is empty (or more accurately, whether there are enough tokens available for the incoming packet).

Token bucket is a reasonable and prevalent way to classify packets into two classes that can be called *in profile* and *out of profile*, where the profile is defined by the two parameters, *r* and *b*. Consider a simple example with an on / off source. This kind of source can be defined by three parameters:

- Average bit rate (A)
- Peak bit rate (P)
- Burst size (B)

For a given pair of *b* and *r*, each source defined by a triple (A,P,B) can be classified either as conforming or nonconforming flows (that is, some packets do not get a token). For instance, if

P = 10*A

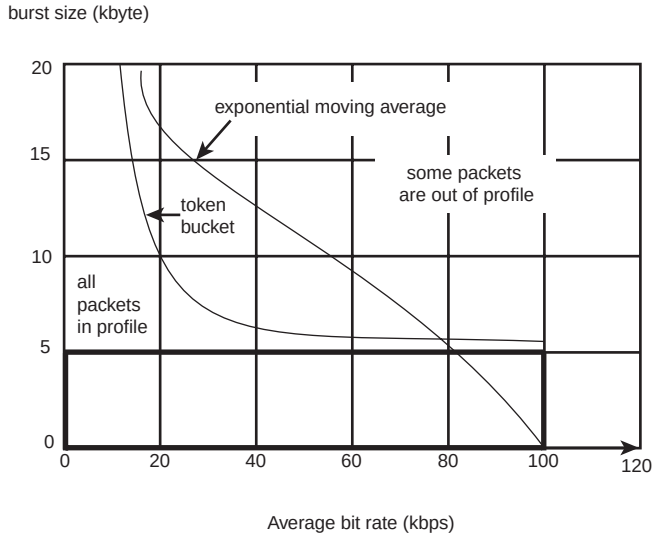
b = 5KB

and r = 100kbps

the classification shown in Figure 6.4 results. The clarity of the token bucket principle stems from the fact that for every source with A ≤ r and B ≤ b, the packets of the flow are

classified as conforming. Furthermore, if $P \leq r$, all packets are conforming regardless of burst size.

Figure 6.4 Conforming and nonconforming flows measured by token bucket and by exponential moving average.



Measuring Bit Rates Another simple alternative is to measure the momentary bit rate at the arrival time of packet j, $M(t_j)$, by using the exponential moving average, as shown in Formula 6.2.

Formula 6.2

$$M(t_j) = M(t_j\text{-}1)*\exp\{[t_j\text{-}t_j\text{-}1]/?\} + B_j/\tau$$

In Formula 6.2, B_j is the size of the packet size in bytes, and defines the measurement period. An alternative simple method is based on windows (see Formula 6.3).

Formula 6.3

$$M(t_j) = \text{sum}(B_j \mid t \geq t_i > t_j - \tau)/\tau$$

This bit rate is the average bit rate over a period τ. It should be noted that the window method requires more memory because all the arrival times and packet sizes during the measuring period should be kept in the memory.

Both formulas produce a minimum bit rate for each value of packet size: $B_j/$. If the packet size is 500 bytes and the measuring period is 10 milliseconds, for example, the smallest

measuring result is 500*8 / 0.01 = 400kbps. This is a quite high value for an individual flow and, therefore, the measuring period should be longer if possible.

It is not clear why the measuring period should be short in general. What would happen if you applied an essentially longer period (for example, 10 seconds)? As to the fairness, the question can be asked in the following way: If each customer (i) is promised to obtain bit rate R_i, or a share S_i of network resources, what is the period on which the share or bandwidth should be determined? You may have different answers, ranging from one millisecond to one month. If your answer is relatively long (say, one hour), totally different traffic processes are considered equal.

A flow (F1) with constant bit rate of 10kbps is considered equal to another flow (F2) with one 10-second burst once an hour with a bit rate of 3.6Mbps, for example. This result seems to be inappropriate. If the period is 10 seconds, flow f2 is considered equal to a flow F3 with a permanent constant bit rate of 3.6Mbps. This result also seems to be inappropriate.

One solution to this dilemma is that instead of one measuring period, there are two (or even more). The total measuring result can be of the form shown in Formula 6.4.

Formula 6.4

$$M(t_j) = c_1 * \{M(t_j\text{-}1) * \exp\{[t_j - t_j\text{-}1]/t_1\} + B_j/t_1\} + (1\text{-}c_1) * \{M(t_j\text{-}1) * \exp\{[t_j - t_j\text{-}1]/t_2\} + B_j/t_2\}$$

Whether this additional complexity provides enough advantages is hard to assess without practical experience. With flat-rate pricing, the answer can be positive; with time pricing, however, the answer is most likely negative.

6.1.3 *Packet Marking*

Packet marking has two basic targets, as described in the preceding section. The first is to mark packets below and above a given threshold for bit rate. This can be generalized to a system with several thresholds. The second target is to somehow reflect the smoother function in Figure 6.3—that is, how much below or above the measuring result is compared to a standard value. From the perspective of Differentiated Services, the main objective in both cases is to map packets into one of the available importance levels of the PHB class used by the flow. When the measuring results are available, the actual marking is a mere technical issue.

There is a degree of freedom to build different models if the system provides measuring results on different levels of aggregation. The actual marking may depend on the measuring result of the particular PHB, of total traffic sent by the customer, and of the total traffic load of the end user's organization. Very complicated models are not recommended because of the additional management burden and higher probability of erroneous configurations.

Furthermore, an additional issue may have a significant effect on the robustness of the system. There are two marking principles:

- When a packet exceeds a threshold, it is marked as low importance, but it is not used to determine the load level of the following packets. Effectively, the allowed bit rate of the higher importance level is totally independent of the load of lower importance level. This principle is usually used with the token bucket system.
- When the momentary load level exceeds a threshold, every packet is marked with lower importance. This principle seems to be more reasonable with several importance levels, in particular, if the intention is to offer different availability levels.

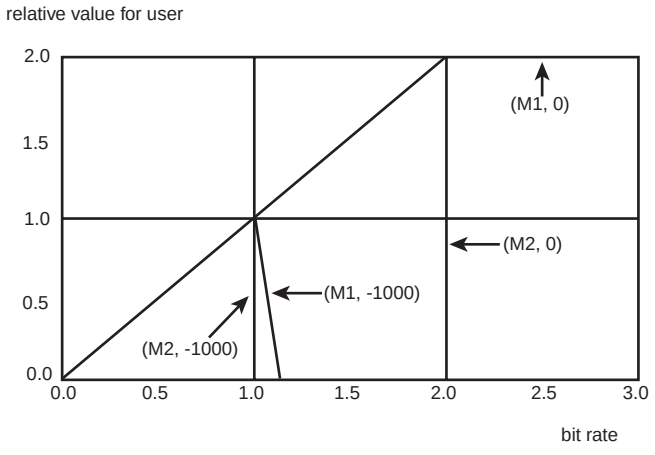
The following example illustrates the fundamental difference between these principles. Suppose, for example, that there are three importance levels with packet-loss ratios 0, 0.01, and 1, and thresholds between the levels are 1 and 2. In marking method M1, the system marks as many packets as possible with the highest importance level, or with the intermediate importance levels if there are packets that cannot be put into the highest level.

In method M2, all packets are marked with the same importance level based on the bit rate of the flow: If the threshold is below 1, all packets attain highest importance; if the threshold is above 2, all packets get the lowest importance.

Now the final effect of packet losses depends crucially on the application that needs the network service. With certain applications, such as videoconferencing, the value of the service does not only depend on the successfully transmitted packets, but the negative value of a lost packet can be remarkable, because even one lost packet can generate significant disturbance in the application. Suppose that the value of a successfully transmitted packet is 1, and the value of a lost packet is –1000. (That means that a packet loss ratio of 0.001 or higher yields a worthless service.) In this case, the result from the user's perspective is quite similar with both marking principles, marked by (M1, –1000) and (M2, –1000) in Figure 6.5.

Alternatively, if the application is not at all sensitive to packet losses, there is no negative value for the lost packets. That is, in principle, the case if the application can retransmit lost packets, and only the total amount of successfully transmitted packets is an important factor. In this case, there is a significant difference between the two marking methods. After the bit rate exceeds the second threshold (2), the value for the customer basically remains constant with marking method M1, whereas marking method M2 results immediately in zero value for the customer. Therefore, the method M2 induces much stronger incentive for the customer to control its bit rate and not send excessive packets.

Figure 6.5 Two marking principles (M1 and M2) with two values for lost traffic (0, –1000).



6.1.4 *Traffic Shaping*

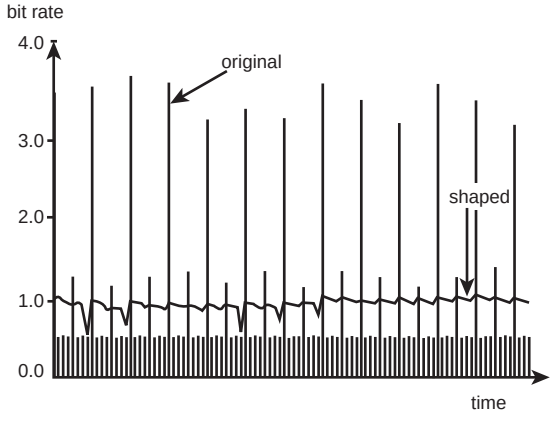
The basic idea of shaping is that if a packet should be re-marked to a lower importance level, an alternative could be to shape the traffic process in such a way that re-marking (or dropping) is not necessary. Then you would obviously need another metering to determine whether the result is acceptable, however. In the case of several importance levels, it can be quite hard to decide when shaping is reasonable and when it is not, because this issue depends also on the delay constraints.

The user is, of course, allowed to shape the traffic before it is sent to the network. In general, that is probably a more reasonable scenario than shaping inside the network. If the only alternative for shaping is immediate dropping, the shaping option inside the public network might be reasonable. It is fair to conclude that shaping is usually better to do before metering and marking, or alternatively, in an integrated unit that could be able to make an optimal decision.

Therefore, you may consider shaping as something that is done for a given traffic process to reduce traffic variations, and by that means to improve the treatment inside the network. This is relatively simple action if there is a known target bit rate (or packet rate) and a known limit for additional delay. In contrast, it is not at all clear whether it is practical to delay packets to smooth a traffic process for achieving some benefit in some other part of the network. The two main points seems to be that shaping at the edge may decrease the delay variations inside the network for some other flows, and statistical multiplexing can be improved inside the network.

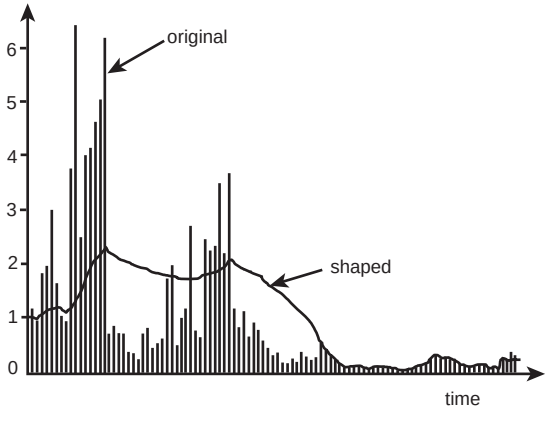
If the traffic process includes frequent high peaks, the multiplexing target could be realistic. An MPEG type of video coding could generate the traffic in Figure 6.6. In this case, it surely is reasonable to smooth the regular high peaks if the additional delay is not too long for the actual purposes of the application.

Figure 6.6 Traffic shaping for an MPEG-coded video stream.



In that case, Figure 6.7 shows a case in which the traffic process is more irregular and has significant variations on different time scales. Although the traffic process could somewhat be smoothed, it is not feasible to even variations on time scales longer than some tens of milliseconds. If middle- or long-term variations are dominant, the real effect of traffic shaping on statistical multiplexing is likely insignificant. It is actually better to drop some packets rather than delay them excessively, because that could be the only way to reduce the load level.

Figure 6.7 Traffic shaping for traffic process with variation on all time scales.



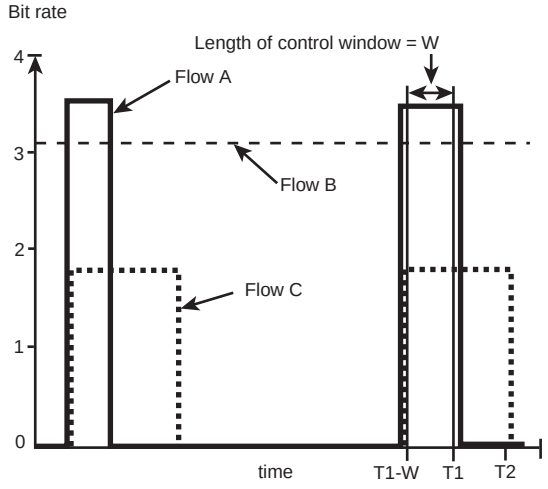
As to delay target, the main conclusion could be that if there are different flows with differing delay requirements multiplexed in the same aggregate stream, it could be advantageous to shape streams with loose delay requirements at the boundary nodes as much as allowed. The main point is that traffic control inside the network is in practice easier with a smooth traffic process than with high variable traffic, even though in theory the drop and delay characteristics could be quite similar with and without shaping. A more recommended solution is always to use different delay classes on the one hand, for flows requiring low delay, and on the other hand, for flows with high bit-rate variations.

In particular, if and when traffic control is based on momentary load rather than on long-term average load, some packets may get essentially better treatment after shaping. In Figure 6.7, for instance, the boundary node may mark the packets belonging to the highest peaks by low importance if it interprets the situation in such a way that the traffic load is high during the peaks—that is, it does not take into account that the peak rate is used very briefly.

Figure 6.8 further illustrates the situation. For instance, the decision of packet dropping may depend on the numbers of packets sent during a control window in a relatively short period (W). Suppose, for this example, that there are two flows on a link, an on / off flow (A) and a constant bit-rate flow (B).

If congestion emerges at time T1, the control system notices that there have recently been more packets in flow A than in flow B. In consequence, the node discards packets from flow A rather than from flow B. Because the average bit rate of flow B is much higher than that of flow A, it is more likely that flow B is more responsible for the congestion than flow A. Depending on several issues, however—such as other traffic flows and buffering capacity—the situation could be considered the other way around as well.

Figure 6.8 Effect of shaping on the metering result.



Then if you replace flow A by a smoother flow C with the same average bit rate, the result could be that the point of congestion is moved from T1 to T2 and the same amount of packets are discarded as in the original case. The difference is that, in this case, the control system expects that flow B is more responsible for the congestion than flow C, and as a result, discards packets from flow B. In this case, the discarding decision seems to be the right one. In summary, the effect of shaping can be quite dramatic for the application, although it is possible that the real effect from the network operator point of view is insignificant (for instance, the total amount of discarded packets remains the same).

6.1.5 Packet Dropping at Boundary Nodes

The last option is to drop packets before they enter the network. Certain services, particularly those using guaranteed connections or leased-line services presented in Figure 5.1 in Chapter 5, "Differentiation of Customer Service," may require that nonconforming packets be discarded immediately. Clear and unambiguous rules and algorithms should be used with these services, because erroneous packet drops violates the service model.

Although immediate packet dropping based merely on the traffic process sent by the customer is somewhat opposite to the fundamental principles of the resource-sharing model, an additional threshold for every user may restrict the total traffic sent by the customer. The problem of a straightforward system that controls only the total traffic is that it does not take into account the differences in packet importance. Customers would surely accept that some of the least important packets are dropped at the network boundary; whereas they probably would not approve if any of the most important packets were dropped immediately. A more complicated dropping system might solve this problem.

6.2 Traffic-Handling Functions in Interior Nodes

Basically the same tools are available both in boundary and interior nodes. There are, nevertheless, certain differences with regard to the main tasks. Boundary nodes are mainly responsible for packet marking and classification, but the main tasks of internal nodes are buffering and discarding. In addition, interior nodes may give information about the load or congestion to traffic sources.

In principle, the same classification as in the boundary nodes can apply to interior nodes as well. According to the main principles of Differentiated Services, however, interior nodes classify packets based on the DSCP field of the packet. In that sense, the situation is very clear. You may still ask whether some other information can be used as well (for example, the information provided by MPLS). This is surely a possible scenario, but it is not discussed further here because it is somewhat beyond of scope of Differentiated Services.

6.2.1 Buffering

A lot has been written about different queuing systems. For an in-depth analysis of the different systems, refer to *Queueing Systems, Volume 1, Theory* (Kleinrock 1975), *Queueing Systems, Volume 2, Computer Applications* (Kleinrock 1975), and *Broadband Network Teletraffic* (Roberts 1996). This section does not try to explain every queuing system in exhaustive detail. The main goal is to give an overview of the most reasonable queuing system, particularly from a Differentiated Services perspective.

First In, First Out (FIFO)

The primary model of queuing is *first in, first out* (*FIFO*). Because the simplicity is one of the key goals of Differentiated Services, it can be always considered as the default choice. A more complicated queuing system should be applied only if a FIFO system cannot offer appropriate characteristics.

In FIFO the packets that want to use an output link are placed into a queue in the order in which they were received. In the basic form of FIFO, every packet accepted into the queue is also transmitted forward, and moreover, packets are discarded only if the queue is so full that there is not enough space for the incoming packet.

As to the four attributes used for assessing Differentiated Services in this book (fairness, cost-efficiency, versatility, and robustness), the FIFO system provides certain clear advantages.

Cost-Efficiency FIFO can offer high cost-efficiency because the output link and buffer capacity are utilized very efficiently (although not necessarily in the absolutely optimal manner) with a simple mechanism. In that respect, there is not much to be gained with a more complicated mechanism. On the other hand, there are some significant problems with the other three attributes.

Fairness As long as all users behave in the same way—that is, they send packets according to an identical traffic process—FIFO appears to be quite fair. The delay and packet-loss properties are similar for every flow. That is, however, not a sufficient criteria for fairness, because a versatile system must also be able to offer fairness among customers with different behavior patterns. One of the fundamental problems of FIFO is that the system provides the same treatment for flows with a very high bit rate and a very low bit rate. That is, of course, not a problem if the customer is charged according to the bit rate; if the customer is charged a flat rate, however, this can be a significant problem.

Robustness Although a part of the fairness problem can be solved merely by providing appropriate traffic control in boundary nodes, even in the case of pure FIFO buffering

inside the network, it seems that a high level of fairness requires a more complicated queuing. The same statement is largely valid for robustness as well. A basic FIFO relies completely on the traffic-control functions in boundary nodes, which may make the overall system too vulnerable to misbehaving users.

Versatility Moreover, the most critical shortcoming of FIFO from a Differentiated Services viewpoint is that it definitely does not support quality differentiation, because every flow going through a FIFO queue recognizes the same delay and packet-loss characteristics. Particularly, delay differentiation is practically impossible to achieve without a more advanced queuing system in interior nodes.

The main approach applied in this chapter and following chapters is to solve the versatility, fairness, and robustness problems of FIFO queuing. The following list summarizes the goals:

- It must provide reasonable tools for differentiation related to delay and loss characteristics.
- It must provide reasonable fairness even in cases where customers have totally different behavioral properties.
- It must work appropriately even in cases where the traffic control at boundary nodes does not work perfectly.

Priority Queuing

The first consideration is the issue of delay differentiation. This target can be met by using one queue for every delay class. In the simplest case, there are two queues with a strict priority discipline; that is, the lower-priority packet is served only if the higher-priority queue is empty (when the packet is starting the service).

A basic performance evaluation for the higher class is quite straightforward. If the link speed is C, the size of the higher-priority buffer is S_H, and the maximum size of packet is B~max, the maximum delay for a higher-class packet is $(S\sim H,+ B_{max}) / C$. For instance, if link speed is 100Mbps, buffer capacity is 10KB, and maximum packet size is 1KB, the maximum queuing delay is approximately 0.9 milliseconds.

On the contrary, it is difficult to declare much about the delay of the lower-priority class, because that issue depends crucially on the traffic load and variations of the higher class. Yet, if the network operator can guarantee that the load level of the higher class remains always below a certain value, an evaluation of maximum delay for the lower class can be made. If the load level of the higher class measured over period *T* is always less than ρ_H, the maximum delay for lower class is bounded by the limit shown in Formula 6.5.

Formula 6.5

$$D_{max,L} < r_H * T + (S_L + S_H)/[(1-r_H)*C]$$

In Formula 6.5, S_L is the buffer size of the lower class and S_H is the buffer size of the higher class. This approximation is based on the assumption that when a lower-class packet arrives, the queue of the higher class is full, and in the lower queue there is space exactly for the incoming packet. To formulate an approximation, you must keep in mind two factors:

- The time that the higher queue can remain full because of the higher-class traffic
- The time needed to empty both buffers if you suppose that the load of the higher class remains on the level of $\rho_H * C$

If S_L = 10KB, S_H = 1000KB, ρ_H = 0.5 and T = 0.1 seconds, for instance, the maximum delay for the lower class is less than 212 milliseconds.

The practical problem is how the operator can effectively control the load level of a service class in interior nodes. It seems apparent that there is not any strict method because the load on every interior link is not controllable in a pure Differentiated Services networks. Nevertheless, something can be done for better control of delays as shown in the next sections. It should be stressed that in all practical implementations, the load level of the low-delay class should be somehow controlled to keep the packet-loss ratio at an acceptable level. This is particularly important because small buffers mean that the nodes cannot absorb a large burst of packets in the same way as larger non-real-time buffers.

Weights for Controlling Delays

The fact that traffic loads cannot be tightly controlled inside a Differentiated Services network makes the delay control problematic. Although the incoming packet stream in an interior node is largely uncontrollable, the outgoing stream of each service class can be rigorously controlled. Suppose, for example, that a PHB class *i* gets a proportion w_i of the whole link capacity when every queue is non-empty. Moreover, if one or several queues are empty, the link capacity is divided in proportion to the shares. If you ignore the problems to realize perfect weights with variable size packets, the maximum delay for class *i* is that expressed in Formula 6.6.

Formula 6.6

$$D_{max,i} = S_i / (w_i * C)$$

The main disadvantage of this approach is that although the weights seem to solve the delay-control problem, the target is attained at the expense of increased problems related to packet losses.

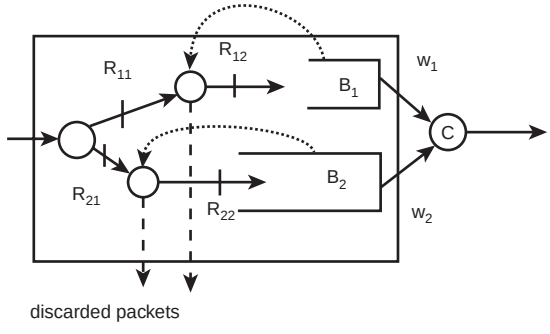
You may ask whether there is any fundamental difference between the approach based on weights and the approach that seeks to directly control the incoming load of each service class. In a certain sense, there is not much difference, because the weights together with finite buffer sizes actually control the incoming bit rates after the packet discarding (that is, R12 and R22 in Figure 6.9). For instance, the maximum average rate at point R12 over a period of length t is that expressed in Formula 6.7 if the buffer S_2 is not empty during the period t.

Formula 6.7

$$R_{12},max(t) = w_1*C + S_1/t$$

Therefore, if the load of Class 2 is high, the incoming flow to the Class 1 is controlled basically in the same way as in the delay-control system based on direct control of incoming traffic presented in the preceding chapter.

Figure 6.9 Bit rates before (R_{11} and R_{21}) and after (R_{12} and R_{22}) packet discarding.



Weights are, however, better in the sense that they allow the use of total link capacity for one class when there is no traffic demand in the other class(es). Two key things make the weight approach better. First, weights yield a control loop from the buffer to the packet-discarding mechanism. The second key thing is the controlled sharing of the network capacity. Even the straightforward method to discard packets only when the buffer is full is definitely better than to discard packets merely based on the incoming bit rates of each class (R_{11} and R_{12} in Figure 6.9) regardless of the load in the buffers. This system, by itself, cannot manage packet-loss ratios if the incoming bit rates (R_{11} and R_{21}) are not regulated. Therefore, both fairness and robustness of this system depend crucially on traffic control at the boundary nodes.

Table 6.1 shows some numeric values that illustrate the delay characteristics of a queuing system connected to a 100Mbps link. The buffer size of the real-time class is so small compared to the typical packet size in IP networks that load level has to be low to obtain a

small packet-loss ratio. Other tools to control packet-loss ratio in case of small buffers are to limit the maximum packet size and to limit traffic variations. This is the main reason why the metering function at boundary nodes should be different for real-time classes (as discussed in Chapter 5 in section 5.3.9 “Variable Bit Rate”).

The maximum delay for the other two classes calculated by Formula 6.6 can be quite conservative, because the formula supposes that Class 1 utilizes its whole share all the time. If and when the average load level of Class 1 remains low—for instance, below 0.3—the real bandwidth share of the other classes is significantly higher than the theoretical minimum share. For instance, the capacity used by classes 1 and 2 during a 100-milliseconds period very seldom exceeds 50%. It should be stressed the figures in the rightmost column are valid only if the network-management system and the traffic control related to classes 1 and 2 can appropriately limit the load levels of classes 1 and 2.

Table 6.1 Maximum Delay for Three Traffic Classes

Delay Class	Share	Buffer Size Kb	Theoretical Maximum Delay (ms)	Realistic Maximum Delay (ms)
Class 1	0.75	10	1.07	0.5
Class 2	0.20	50	20	10
Class 3	0.05	500	800	100

Equal Queuing

As discussed in the preceding section, the basic mechanisms that can be used to control delays are often inappropriate for controlling bit rates or packet-loss ratios. This section offers an overview of a simple system that can be used to improve the fairness and robustness of FIFO queuing. This discussion uses the general term *equal queuing* for those systems that try to divide the link capacity evenly among some entities. The entity can be an active flow or an aggregate traffic stream—it is up to the service provider to specify the entity and to use an appropriate system to classify packets according to the specification.

This discussion avoids using the term *fair queuing*, because equal share is not necessarily synonymous with fair share. (Note that according to RFC 1254, “Gateway Congestion Control Survey,” fair queuing is the policy of maintaining separate gateway output queues for individual end systems by a source-destination pair [Mankin, Ramakrishnan 1991].)

From the Differentiated Services viewpoint, the fundamental difficulty is that interior nodes are not supposed to keep any per-flow records. Further, even though an equal share

for every individual user could be considered fair, the same approach is not usually applicable to aggregate streams. What then would be the use of equal queuing in a Differentiated Services network? One tentative answer is robustness. Although equal share is not necessary an optimal result, and it is not easy to classify packets accurately into meaningful groups inside the network, even an inaccurate classification may significantly improve the robustness of the total service system.

Several proposed queuing systems can be classified into this group of queuing systems. A brief outline of two of them is offered here: Stochastic Fairness Queueing and Fair Buffer Allocation.

Stochastic Fairness Queueing (SFQ) McKenney suggested Stochastic Fairness Queueing (SFQ) as a technique to solve the implementation problems of more complicated fair queuing systems (see *Stochastic Fairness Queueing* [McKenney 1990] and RFC 1254, "Gateway Congestion Control Survey" [Mankin, Ramakrishnan 1998]). SFQ classifies packets according to the source-destination address pair in an incoming packet. SFQ uses a simple hash function to map the packet with an address pair to one of the available queues. The hash function means that the assignment of an address pair to a queue is probabilistic. Therefore, it is possible that packets belonging to different flows are classified into the same queue. To improve the long-term fairness, SFQ periodically changes the hash function in a way that the same address pairs do not collide continually.

According to McKenney, you may need to have up to 5 to 10 times more queues than the number of active source-destination pairs. Note that the activity has in this case a very limited sense: Only those flows that have at least one packet in the buffer are active. It is possible to assess the effect of stochastic sharing by a simple mathematical model. The probabilities that a flow has its own queue or that it has to share the queue with one or several flows can be calculated from binomial distribution. Table 6.2 shows the result in a case with 1,000 queues and either 100 or 200 active flows. You can further calculate a theoretical value for the available bit rate, if you suppose that SFQ can divide the bandwidth exactly among all active queues and that the bandwidth of a queue is divided exactly evenly among flows directed to it.

Table 6.2 SFQ Queuing System Sharing a 10Mbps Link

	100 Active Flows		200 Active Flows	
Sharing of Queue	Probability	Bit Rate (kbps)	Probability	Bit Rate (kbps)
Alone	0.9057	105.0	0.8195	55.1
With 1 flow	0.0898	52.5	0.1632	27.6
With 2 flows	0.0044	35.0	0.0162	18.4
With 3 flows	0.0001	26.3	0.0011	13.8
With 4 flows	0.0000	21.0	0.0001	11.0

It should be noted that although SFQ clearly increases the bit-rate variation compared to a perfect system (that is, equal sharing of bandwidth), the number of active flows is in practice a variable as well. The user cannot, therefore, expect that the available bit rate for a flow remains constant in any resource-sharing system. Therefore, the essential issue is whether the increase of bandwidth variation is acceptable or even noticeable. The final assessment of SFQ depends on the service model adopted by the service provider: The more guarantees promised, the more queues needed. Further, because of the stochastic nature of SFQ, it seems very difficult to use different weights for different flows.

Fair Buffer Allocation In the Fair Buffer Allocation, the interior node keeps track of the number and / or size of packets in each buffer. When the occupancy level of the buffer exceeds a certain limit, the queuing system starts to discard incoming packets belonging to those streams that have the most packets in the buffer. The decision formula proposed in "A Fair Buffer Allocation Scheme" was originally intended for ATM networks with constant size packets (Heinanen, Kilkki 1995). If you modify it to IP networks with variable-sized packet, the rule shown in Formula 6.8 results.

Formula 6.8

Discard an incoming packet belonging to stream i

if $x > c_2$ and

$S(i) / E\{S(i)\} > c_1 * [1 + (1 - x) / (x - c_2)]$

for which

$S(i)$ = buffer space used by stream i

$E\{S(i)\}$ = average buffer space used by all active streams

x = occupancy level of the buffer—that is, the amount of used buffer space divided by the total buffer size

c_1 should be smaller than 1 to keep the probability of a totally full buffer small

c_2 defines the lowest occupancy level on which the FBA is active

An obvious result of this system is that the packet-loss ratio of streams with a high bit rate (or burstiness) is higher than that of smooth streams with a low bit rate. With TCP it tends to equalize the bit rates of different flows.

Class-Based Queuing (CBQ)

According to Ferguson and Huston, in the book *Quality of Service, Delivering QoS on the Internet and in Corporate Networks*, the fundamental goal of CBQ is to prevent complete

resource denial to any particular class of service (Ferguson, Huston 1998). (See also "Link-Sharing and Resource Management Models for Packet Networks" [Floyd and Jacobson 1995].) Hence, the objective of CBQ is to solve the starvation problem of strict priority queuing. The basic assumption of CBQ is that traffic is classified into relatively large aggregates according to a principle that depends on the service model. (See section 4.2.2, "Levels of Aggregation," in Chapter 4.) The three main principles are as follows:

- Requirements of applications
- Price paid by customers
- Organization of the user

Requirements of Applications As to the CBQ, the first option means that flows are grouped according to the application, and one class can be used simultaneously by a large number of users. What is the fairness (or robustness) problem that CBQ tries to solve in this case? Apparently users are grouped according to the application they are using. If a group is too active or too large compared to other groups, it is considered fair to limit the available bandwidth for those users. If a user then changes the group—for instance, from IP telephony to a data application—he may obtain better service because the latter group happens to be smaller at that moment.

So the fundamental idea is that a group of users should not utilize the whole capacity even though the application they are using may need higher quality than another application. What does this statement actually imply? One interpretation is that the importance of an individual packet depends on the aggregate load level of the class. The more users there are in the same service class, the less important individual packets are, and vice versa.

Because the relative importance of packets from different classes depends both on the load levels and on the weights of each class, it is practically impossible to make any useful predictions for the quality of individual flows. It is not possible to state that the packet-loss ratio in a class is always lower in Class 1 than in Class 2, for instance. In particular, without any additional traffic-control mechanisms, this system is vulnerable to misbehaving users. This application model could, however, be reasonable in certain cases—for instance, in private networks.

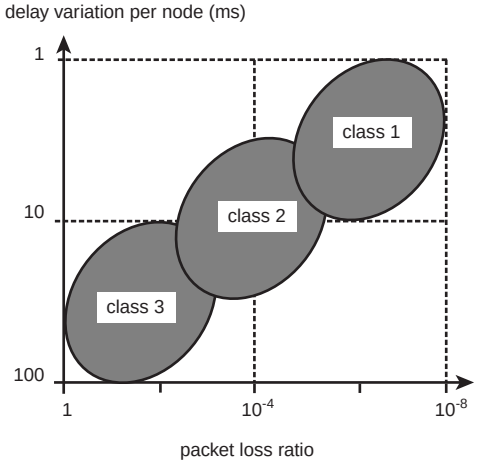
The following evaluation of other approaches is based mainly on the pricing model—that is, on the assumption that the main classification criteria is related to pricing.

Higher Price, Better Quality A common claim related to CBQ is that lower relative load level somehow automatically means better service quality. You can attempt to attain different relative load levels either by regulating incoming traffic or by changing the weights dynamically. Static weights without traffic control, on the other hand, seems to be an unrealistic approach.

A simple approach is to just give different prices for different classes (see, for example, the Paris Metro pricing discussion in "A Modest Proposal for Preventing Internet Congestion" [Odlyzko 1997]). Another approach could be that each user is allowed to send more traffic if he selects a lower class. (Note that there is actually not much difference between these two approaches.) Regardless of the mechanism used to induce load differences, one question remains: In what way is the service of the higher class better than the service of the lower class? The answer might be more bandwidth or smaller packet-loss ratio or lower delay, or possibly even a combination of these.

It seems apparent that the loss ratio of every flow within a class is approximately the same if there is only one importance level in every class. If the load level is the only significant factor, it seems obvious that both delay and loss characteristics are better in Class 1 than in Class 2 or 3, if the relative load level is highest in Class 3 and lowest in Class 1. Figure 6.10 shows a possible target of this system.

Figure 6.10 A target for quality differentiation of a CBQ system.



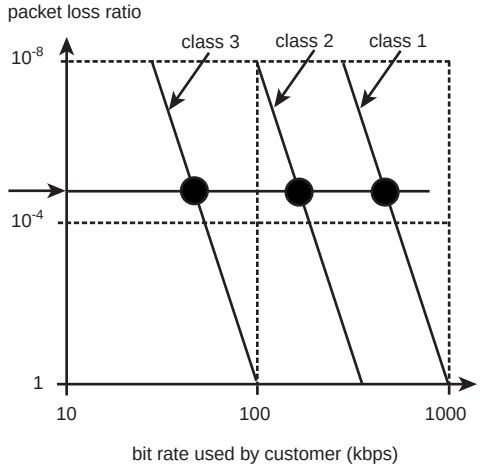
If the average load level is the only controlled factor, however, the relationship between delay and loss is not as clear as can be expected. Both low delay and low loss ratio can be achieved only if the traffic load is somehow controlled within a relatively short period, as described in the section "Priority Queuing" earlier in this chapter.

A short control period actually means that traffic must be smooth (or the load level must be extremely low). Unfortunately, high price does not, by itself, encourage smooth traffic. On the contrary, the first assumption of some customers may be that high price means freedom to send whatever they want. It is not recommended, in general, to mix delay-sensitive and bursty data traffic in the same class, even in the case of a low load level. An

additional mechanism is needed to prevent users from sending highly variable traffic to a low delay class.

Higher Price, More Bandwidth The general problem with trying to associate high price with high quality is that the system should also define how the quality depends on the traffic sent by the customer. It does not appear reasonable just to have three classes without defining the relationship between sent traffic and quality. A CBQ system could have a more advanced goal, such as within each class, higher bit rate means higher packet-loss probability. The difference between the classes could then be in the available bit rate. Figure 6.11 depicts this target system.

Figure 6.11 Target "higher price, more bandwidth" for a service model using CBQ.



Although this target, or goal, could be achieved by means of CBQ with some additional mechanisms (discussed later in section 6.2.2, "Discarding," it is important to ask whether CBQ is the best tool for trying to attain this goal. Suppose, for example, that flow 1 with a bit rate of 500kbps uses Class 1, flow 2 with a bit rate 150kbps uses Class 2, and flow 3 with a bit rate 50kbps uses Class 3. According to Figure 6.11, the packet-loss target for every flow is approximately the same, 10^-5. In this case, the fundamental issue is how the treatment of these three flows will differ inside the network. The importance of the packets belonging to each flow is the same, and the bit rate should be controlled in the boundary node.

Delay is the remaining basic service characteristic. It is not, however, clear why the operator should associate low delay with high bit rate. Once again, a feasible low delay service cannot be realized without a mechanism that controls the traffic burstiness.

Summary of CBQ Although Class-Based Queuing is technically a reasonable approach, it is somewhat difficult to define a feasible purpose for its application. The three targets identified here are the requirements of the applications; higher price and better quality; and higher price and more bandwidth. Each of these could be met with CBQ, and CBQ might even be practical in some cases. Nevertheless, the fairness criterion of the Internet is that customers get service proportionate to the price they pay. CBQ doesn't seem to meet this expectation; if it did, CBQ is difficult to apply efficiently.

Weighted Fair Queuing (WFQ)

You can find overviews of Weighted Fair Queuing (WFQ) in *An Engineering Approach to Computer Networking: ATM Networks, the Internet, and the Telephone Network* (Keshav 1997) and *Broadband Network Teletraffic* (Roberts 1996). According to Keshav, the intuition behind WFQ is to compute the time a packet would complete a service had a General Process Sharing (GPS) server been used to service the packets and then to serve packets to these finishing times. GPS is a theoretical scheduling discipline that shares the bandwidth exactly in proportion to the weight of the connections.

Three facts make WFQ itself not an attractive approach in Differentiated Services networks. First, WFQ requires per-flow queuing. Second, WFQ systems need to know the weight of each flow, and if the requirement changes, a signaling system is needed to transmit relevant information. Third, WFQ requires quite a hard computational effort. All these three factors—per-flow queuing, signaling, and complex computation—are avoided in Differentiated Services networks.

Several variations of WFQ may reduce the computational complexity. These variations include Self-Clocked Fair Queuing (SCFQ) and Start-Time Fair Queuing (SFQ). (Note that the abbreviation SFQ is applied in this book usually to the Stochastic Fairness Queuing). Because all these disciplines require per-flow computations inside the network, they are not discussed further in this book.

If the principle of WFQ is used to offer fair service between aggregates, there remains the fairness problem between flows inside the aggregate. Therefore, it is not necessarily useful to implement complex systems to guarantee optimal fairness between aggregate streams if there is not any mechanism to provide reasonable fairness within the aggregate. This issue depends, however, on the service model applied by the service provider. If the service provider wants to separate the traffic streams among organizations, a variant of WFQ can be a reasonable solution.

6.2.2 Discarding

Various methods have been developed to discard packets better than the basic FIFO queue. In this instance, basic FIFO refers to a queuing system in which packets are discarded only if the queue is so full that there is not enough space for the incoming packet. Usually the decision to discard is based on the current load level: The higher the load level, the higher the probability of discarding a packet. There are various ways to implement this simple idea:

- The load can be measured in several ways. The main approaches are based on the buffer occupancy level and on the bit rate. Although it is also possible to take into account the trend (rate of load change), more complicated algorithms may increase the risk of unstable behavior. Therefore, the starting point should be a simple system based on an average load level.
- Different time scales can be used from momentary occupancy level to a bit rate measurement over a long period.
- The load can be related merely to the PHB class used by the incoming packet, or it could be related to the total load.
- Either hard thresholds or smooth probability functions can be applied to make the discarding decision.

The following sections provide an overview of some discarding principles. The target is not to explain all possible principles, but to emphasize realistic applications for Differentiated Services networks.

Hard Thresholds

The simplest step from a basic FIFO queue is to classify packets into two importance groups. Packets with high importance are accepted whenever it is possible, and packets with low importance are accepted only when the load level is below a certain threshold. Obviously, this kind of system provides a simple tool to realize two importance levels within a class (where all packets of a particular class use the same queue).

Although this is a well-known and elaborately studied system, it is difficult to give any general rule for dimensioning the system—that is, to determine the proper buffer size and the proper threshold for given quality requirements. Nevertheless, you can take the higher importance level as a starting point and suppose that it needs a buffer of size B_H to guarantee sufficient quality. In this case, the worst case scenario is that the traffic load on the lower level is not controlled at all and, therefore, keeps the buffer occupancy level steadily near the threshold. Therefore, the total buffer capacity would be $B_H + B_L$, where B_L is the

buffer space available for packets with low importance. The only significant effect of low importance packets to the high importance packets is the increased delay component.

Although this system can be easily generalized to several levels of threshold, an accurate analysis of the system becomes very complex. In particular, one fundamental question has to be answered before any real implementation with several importance levels can be made. Although a two-level system may work even without any significant traffic control for the lower-importance level, a network with several importance levels certainly requires a systematic control mechanism that makes the use of the system logical.

In principle, the system can rely either on the application model or on the customer model. (See section 4.2.2, "Levels of Aggregation," in Chapter 4.) One plausible reason for the opposition of several importance levels in the Internet is that more than two levels are hardly justified if the service model is based on the different requirements of applications. Therefore, the purpose of several importance levels has to be sought from the customer model instead. And even in that case, the target is not to provide a large number of distinguishable levels of packet-loss ratio, but rather levels of availability (see section 4.2.1, "Availability of Quality," in Chapter 4).

Therefore, the system should support the provision of availability levels in a consistent fashion. One possible target could be the realization of function presented in Figure 5.10 in Chapter 5 (see section 5.3.3, "Price of Availability," in Chapter 5). There could be, for instance, three levels of availability with different prices aimed to cover idle, normal, and busy hours. In addition, the service provider should take into account the effect of different destinations, because the available resources may greatly vary in different parts of the global Internet. These two dimensions, time and destination, together may yield a need for several levels of availability, even though at times two levels of importance could be enough on a certain link for all practical purposes.

Random Early Detection (RED)

Although the previous discussion emphasizes to the need to support customer service with differentiated pricing, there is one clearly identified need to improve the performance in case of a pure application model. The problem relates to the interaction between the congestion control of TCP and the "tail drop" principle, where the tail drop means that packets are dropped only when the queue is full.

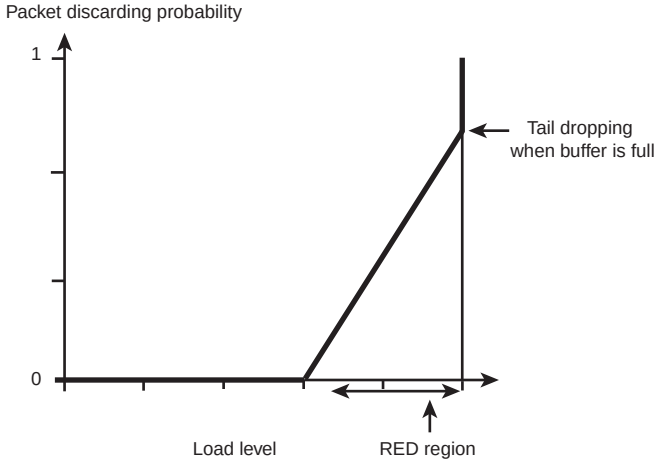
The core of the problem is that although certain properties of a TCP control mechanism are definitely necessary, they result in problems with a pure FIFO queuing. TCP effectively increases the bit rate of a flow until a packet is dropped. Because the control mechanism allows the sender to transmit several packets in a burst, a full buffer may generate a situation in which numerous flows encounter several packet drops. As a result, all these flows

decrease their bit rate sharply, and then start again to increase their bit rate. In the worst case, this may result in a global synchronization where the overall load oscillates between overload and low load levels. For a more detail account of this issue, refer to RFC 2309, "Recommendations on Queue Management and Congestion Avoidance in the Internet" (Braden *et al.* 1998).

Large buffers in IP networks are primarily needed to handle bursts of packets appropriately, not to increase utilization. Note that, in general, the link utilization depends only on the probability that a queue is non-empty, not on the average number of packets in the buffer. Therefore, it can be reasonable to drop some packets even before the buffer becomes full to control the incoming bit rates. In addition, it is desirable that only one packet per flow is dropped and that drops do not occur exactly at the same time. This kind of system, Random Early Detection (RED), was first proposed by Floyd and Jacobson in an article titled "Random Early Detection Gateways for Congestion Avoidance" KC (Floyd, Van Jacobson 1993, 397–413).

Figure 6.12 illustrates the principle behind RED. Under a certain load level, all packets are accepted. In such as case, on the middle region the packet-loss probability depends on the load level. Finally, if the queue, after all, becomes full, all packets are discarded.

Figure 6.12 The principle behind RED.



RED can be evaluated by using the four attributes guiding this discussion: fairness, cost-efficiency, robustness and versatility. RED solves certain problems very effectively due to its simplicity. The increase in complexity compared to FIFO is so small that it does not seem to be an obstacle even in very high-speed routers.

Although RED attempts to solve only specific problems, if (and when) it is used in real networks, network operators and service operators have to be sure that it works appropriately with all applications and services that use queues with RED. In case of large buffers, an appropriately configured RED probably does not have any significant harmful effect. On the contrary, it may both reduce average delay and provide a more even packet-loss ratio among flows than a pure FIFO queue.

In a sense, RED is very fair: The packet-loss ratio is similar with all flows, which means that flows with a higher bit rate encounter a larger number of losses. But then, once again, the real fairness depends crucially on the target, or on the service model. If the target is to share the capacity equally and efficiently, RED serves that purpose well. If the target is more complicated, as it likely is in Differentiated Services networks, a mere RED could be a useful, but not a sufficient, tool.

Because RED seems to decrease the probability of some harmful effects, such as global synchronization, it clearly is more robust than a FIFO system. RED can work effectively only if the majority of flows apply an end-to-end congestion control, however, such as TCP. Therefore, it does not solve the fairness and robustness issues related to nonadaptive flows.

Besides, any system with several free parameters requires additional management effort. To be really efficient, RED parameters should be properly adjusted to each situation with a specific traffic process, link capacity, and buffering scheme. Without a profound understanding of these issues, the parameters could be incorrectly selected; if that happens, system performance could deteriorate.

RED with Several Levels of Importance

The previous chapters separately discussed the need of several importance levels with hard thresholds, and the need of one soft threshold. If and when both needs are justified in real networks, you should be able to combine these two into the same system. Technically, this is a simple issue: Every hard threshold can be replaced by a soft, probabilistic discarding function. However, a couple of practical issues are not totally clear. Should the RED regions overlap each other, and should you use RED for all thresholds?

It seems that the primary situation is that RED regions of different importance levels should not overlap with each other, as shown in Formula 6.9.

Formula 6.9

$$\text{If } 0 < Pr_{drop}(x,p) < 1,$$
$$\text{then } Pr_{drop}(x,p-1) = 1 \text{ and } Pr_{drop}(x,p+1) = 0$$

The component *x* is load level, and *p* is an importance level, and higher numeric value of *p* means higher importance, and $Pr_{drop}(x,p)$ is the packet-discarding probability. In addition, the probability function should be a nondecreasing function, such as this:

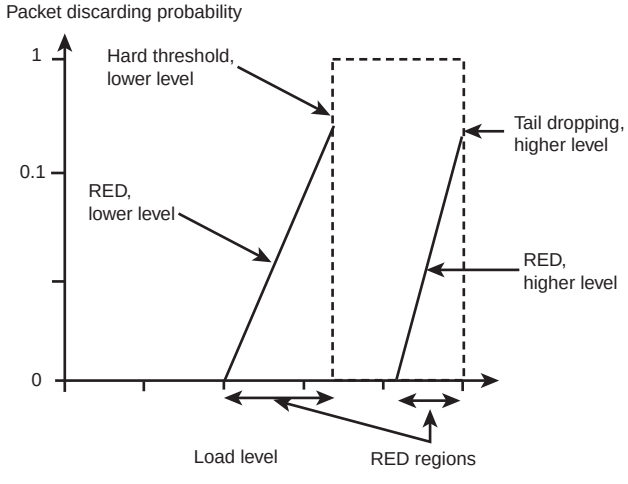
$Pr_{drop}(x,p) \geq Pr_{drop}(y,p)$ if $x > y$

$Pr_{drop}(0,p) = 0$ (for every p, unless the service provider wants to discard all packets of lowest importance levels)

$Pr_{drop}(1,p) = 1$ (because 1 means full system)

If the service provider, after all, decides that RED regions should overlap, that means a weaker separation between adjacent importance levels. Because this is not usually a desirable effect, it could be better to separate the RED regions, and even to leave an additional region between the areas as shown in Figure 6.13.

Figure 6.13 Random Early Detection with two importance levels.



Moreover, it is not clear whether a RED mechanism is useful with the highest levels of importance. With high service levels, the expectation is that packet losses are very rare, and consequently, it is not reasonable to suppose that the bit rates are adjusted based on packet losses. If TCP or another control mechanism is not used to adjust the bit rate, there is not necessarily any significant difference between hard threshold and RED. Yet, there could be some advantages to using RED also with higher importance levels; for instance, it may distribute the packet drops more evenly among active flows, and it may give a warning signal about imminent quality degradation to the users.

Finally, it is possible that in some cases TCP flows that normally use lower importance levels may exploit higher importance levels by decreasing their bit rate below a certain threshold. Therefore, even though under normal circumstances higher importance levels are used to transmit something other than TCP flows, under abnormal conditions a significant portion of the traffic could be TCP flows.

In summary, RED is certainly recommended for the dominant service class of the Internet—that is, for best-effort service with a majority of flows using TCP congestion control. On the other hand, it is not evident whether RED is advantageous with nonadaptive services, although it doesn't seem harmful.

6.2.3 *Feedback Information*

The Differentiated Services model is based primarily on the assumption that each user sends packets into the network according to a traffic process that is essentially independent of the network service. Packets may request a PHB class based on the requirements of the application, and the importance level of the packet is set mainly by the boundary node according to the service level agreement. However, the network does not require that a PHB class be used only if the traffic process complies with certain rules. The definition of PHB may, of course, include that when the incoming bit rate exceeds a certain limit, some packets are discarded immediately. Thus the PHB rules and incoming traffic process together determine how the packets are treated inside the network—but basically the network service does not dictate the incoming traffic process.

This primary principle appears logical as long as the service is based on the customer model. (See section 4.2.2, "Levels of Aggregation," in Chapter 4.) If the service model is based on the requirements of applications, however, with the assumption that users behave appropriately, the network operators may use a different approach in which they try to directly control the incoming traffic process based on the load inside the network.

TCP flow control is one example of this kind of approach (see section 2.3.2 "Basic Best-Effort Service Based on TCP," in Chapter 2, "Traffic Management Before Differentiated Services," and the previous section in this chapter related to the RED mechanism). Although TCP relies on the fact that some packets must be dropped during (severe) overload, some other schemes are based on softer congestion indication. Further, it is possible that the network tries to inform the source about the exact available bit rate (see the section "Available Bit Rate, (ABR)" in Chapter 2). Because that kind of service model requires per-flow calculation inside the network, it is chiefly beyond the scope of Differentiated Services and this book.

Although the current Differentiated Services framework does not include the application of congestion indication, it may be used later with some PHB classes as proposed by

Kalyanaraman, *et al.*, in "A One-Bit Feedback Enhanced Differentiated Services Architecture" (1998). Therefore, this discussion provides a brief overview of two possible approaches: DECbit and congestion avoidance in Frame Relay.

DECbit

The fundamental idea behind the DECbit approach is that the packet header carries a bit that can be set by a network node when the node is experiencing congestion, as discussed in "A Binary Feedback Scheme for Congestion Avoidance in Computer Networks with Connectionless Network Layer" (Jain, Ramakrishnan 1988). The receiver copies the bit to the acknowledgment packet, and sends it back to the source. Finally, the source is supposed to reduce the sending bit rate when more than 50% of the packets have the congestion bit set; if less than 50% of the bits are set, however, the sender is allowed to increase the bit rate.

In the basic DECbit scheme, the network nodes make the decision of setting the congestion bit depending on the bit rate of each connection. The nodes set bits first on those connections that are using more capacity than their fair share. This system apparently does not comply with the fundamental philosophy of Differentiated Services because it requires per-flow calculations inside the network. In a simpler scheme, the network node measures only the aggregate queue length and sets the congestion bit when the average queue length exceeds a threshold.

The main flaw of the DECbit approach is that it requires that both senders and receivers cooperate and be able to adjust their bit rate. If one sender decides to ignore the congestion bit, for example, it may have a harmful effect on all other flows. In such a case, if the operator tries to control the behavior of all customers, complicated mechanisms are needed at boundary nodes. This hard problem has to be solved before a DECbit or another similar scheme can be applied to the global Internet.

Congestion Avoidance in Frame Relay

The congestion avoidance scheme of Frame Relay is similar to that of DECbit (see, for example, the FRForum Web site at `www.frforum.com`). The main difference is that in a Frame Relay network, nodes may inform both sender (Backward Explicit Congestion Notification, [BECN]) and receiver (Forward Explicit Congestion Notification [FECN]) about upcoming congestion. In the Internet environment, backward congestion indication appears impractical, if not impossible, because there is no guarantee that packets in the reverse direction use the same path.

Although there is no specification as to how the user should react when receiving congestion notification, the basic idea is that end users alleviate the congestion by reducing the

bit rate somehow. From a fairness point of view, this is obviously a problematic situation because when some users react and other do not, the final result is an uneven share of resources.

Moreover, Frame Relay specification also introduces a *discard eligible* bit. Essentially, this means that frames can be classified into two importance groups as described earlier in the section "Hard Thresholds". In summary, a Frame Relay node may have four states, depending on the load:

- Low load, where all frames are accepted and no congestion indication bit is set.
- Moderate load, where all frames are accepted but congestion indication bits are set.
- High load, where frames with a discard eligible bit are discarded, and congestion indication bits are set.
- Full buffer means that all incoming frames have to be discarded.

A similar system is surely possible in Differentiated Services networks—actually, the discard eligible bit is a natural part of Differentiated Services. On the other hand, the role of congestion indication is not very clear: The network may, of course, set the bit, but what is the customer supposed to do when he receives a congestion notification?

One approach is to make the congestion notification just extra information provided by the network; users can use the information or ignore it totally. It seems that to be useful in Differentiated Services networks, the congestion notification should somehow indicate on which service (or importance) level the congestion notification is valid. Then, for instance, if the network informs that some packets on the lowest importance level of a PHB class must be discarded, the user can reduce the bit rate in such a way that all packets attain better importance level. Some users, on the other hand, may prefer to use a high bit rate even at the risk of losing some packets. Both behaviors should be equally acceptable.

6.3 Functions Related to a Network Domain

Most of the standardization effort of Differentiated Services relates to the function inside network nodes. In addition to these important aspects, a successful service provision requires a systematic way to manage the whole effort. This section briefly introduces three key issues related to the operation of a network domain: routing, resource reservations, and network dimensioning.

6.3.1 Routing

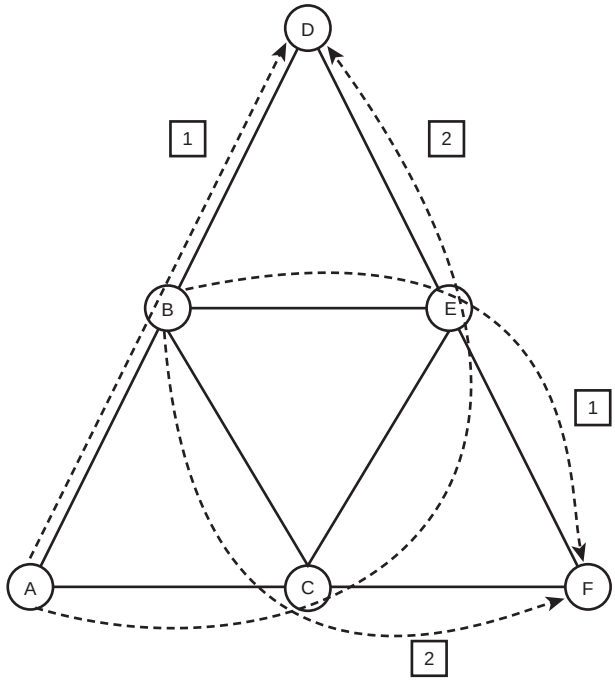
Routing is an integral part of IP networks. Although it is totally clear that routing a system is necessary, the issues related to the dynamics and QoS aspects of routing are less clear.

According to the fundamental rules of both IP networks, in general, and Differentiated Services, in particular, it is not reasonable to apply adaptive routing to individual packets based on the load in the network (that is, within the packet-forwarding function).

In contrast, it could be reasonable to balance the load of different links on longer time scales. Consider, for example, the simple network with six nodes shown in Figure 6.14. Usually the primary route is that with the shortest path. If you calculate the length between each node pair in the network, you get a result of 1.4 hops. In most cases, the secondary route is longer than the primary route. The primary route from A to D consists of two hops, for example, although any other route consists of at least three hops. Although in some cases the number of hops could be the same, such as between nodes B and F, the average length of a secondary route is significantly longer, 2.2 hops, than the average length of the primary route.

As a result, if the load level when using only a primary router is 0.7, and half of the traffic is shifted (randomly) to the secondary route, the average load level is increased to 0.9. Of course, random decision is not rational, and the situation could be much better if the decision is based on some level of optimization. The point is that optimization, or at least feasible reasoning, is necessary when making the decision to use secondary routes.

Figure 6.14 Primary and secondary routes.



The main reason to use alternative routes is *load balancing*, which is a part of the larger problem of long-term network planning, in which the operator tries to optimize the use of network resources. Load balancing can be defined as the capability to decrease the maximum load by changing the traffic from links with high load to links with lower load. Remember, however, that the average load level is increased.

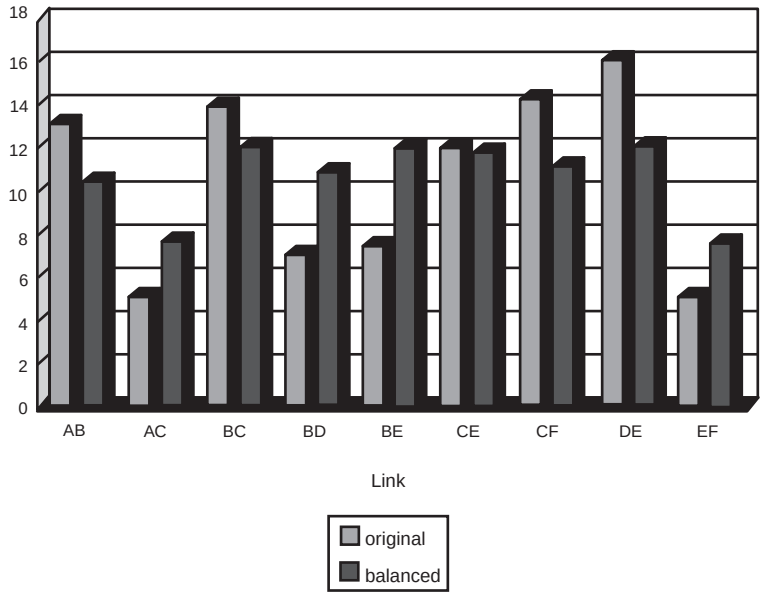
It is possible to decrease the load on links with the highest traffic, for example, but at the expense of a higher average load in the network. Based on a constricted evaluation of the network presented in Figure 6.14, it seems that the maximum load among the links can be decreased by 25% at the expense of a somewhat higher average load level (for instance, 5%). Figure 6.15 shows one example in which the load level of the most loaded link (D-E) can be reduced from 16 down to 12 by using secondary routes. At the same time, the load level of most other links is increased.

This useful, albeit moderate, result is more a theoretical possibility than an achievable proposition in real networks, however, because of the following reasons:

- It is supposed that the average load levels were exactly known. That is, of course, possible with past traffic processes, but that does not mean that the future traffic levels are known.
- It is not necessarily possible to divide traffic streams arbitrarily among different routes; for instance, in the case of AF service, different routes for PHBs within one PHB class cannot be used.
- There is no obvious and simple common goal for optimization; in the example, the goal was just to minimize the maximum load among all links. In Differentiated Services networks, that is not necessarily the right objective because it does not take into account the relative importance of packets.
- In a large network, it is very hard to solve the complex optimization problem (particularly, if the effects of the first three items are taken into account).

To summarize, even though alternative routes can be useful, especially during extraordinary situations (for instance, if a cable is broken), they probably do not essentially improve the network performance during normal situations. Load balancing is reasonable, but only with other appropriate planning and management tools. One possibility in Differentiated Services networks is to use load balancing mainly for the highest importance levels and to ignore the effect of the lowest importance levels.

Figure 6.15 Load balancing using secondary routes.



6.3.2 *Resource Reservation*

The general argument of this book relating to resource reservation is that it should be used only when really necessary, not as the only standardized solution to all quality problems.

Note

The basic characteristics of resource reservation are discussed throughout this book. For instance, see section 2.4, "Integrated Service Model," in Chapter 2; the section "Resource Reservation" in Chapter 4; and the section "Service-Level Agreement" in Chapter 5.

Considering resource reservation from the perspective of Differentiated Services, the most practical level of resource reservation is the PHB class. In addition, it could sometimes be necessary or reasonable to make capacity reservations for large organizations. In both cases, the rationality of the reservation should be assessed carefully, because reservations tend to complicate networks and their management and deteriorate statistical multiplexing.

The technical part of resource reservation is mostly beyond the scope of this book. It can be done either by using an IP protocol, such as RSVP or MPLS, or by using some other technology, such as virtual paths and virtual connections in ATM networks.

One way to implement resource reservation is with a bandwidth broker, as discussed in "A Two-Bit Differentiated Services Architecture for the Internet" (Nichols, Van Jacobson, Zhang 1997). The idea of a bandwidth broker is, in itself, clear. Rather than using a signaling system to separately ask every node through the path whether there is enough available resources to support a new flow, the request is handled primarily by one or a couple of bandwidth brokers. Consequently, an efficient bandwidth broker must have real-time information about the load within a network domain, and about the prices of all types of flows.

Although this approach is clearly unsuitable for all short-duration flows in the Internet, it may offer a reasonable solution for those (rare) cases in which a significant amount of resources are required through the network and where additional pricing is needed to control the use of such resources. A bandwidth broker can also be used solely to inform users about the characteristics and prices of different service classes.

6.3.3 *Network Dimensioning*

Dimensioning a network that is used to transmit Differentiated Services is an intricate task. The traffic processes and quality requirements vary almost infinitely—how is it possible at all to determine appropriate capacities in this kind of situation? It is fair to say that dimensioning is an almost impossible, albeit inevitable, task for any network operator.

This chapter briefly outlines two starting points for dimensioning. The first one is based on the methods and models used in connection-oriented network. The second one is mainly based on the idea that practical experience is your only guide when dimensioning packet networks with variable traffic processes and quality requirements. Certain other possibilities exist as well, but they are left to a more specific book.

Connection-Oriented Networks

Quite a lot of theoretical research has been conducted in the area of dimensioning connection-oriented networks; the history of this research can be traced back to the studies of A. K. Erlang. What Erlang achieved was a simple, but efficient formula for call-blocking probability under certain statistical assumptions. If the inter-arrival time between successive call attempts is an exponentially distributed random variable with mean λ, the average holding time is h, and the number of channels is S, the call-blocking probability is that shown in Formula 6.10.

Formula 6.10

$$Pr(\text{blocking}) = \frac{(A^S)/S!}{\sum_{i=0}^{S} (A^i)/i!}$$

A = average offered load = λ / h. If the network operator has a rational approximation for average load level (A) and a target for blocking probability, he can just use the Erlang's formula to define the required number of channels.

Although all the theoretical assumptions are not valid in reality, telephone networks have largely been planned using this formula. Why can't this be applied to the dimensioning of packet networks as well? Actually it is possible, supposing that dimensioning is based on the requirements of reserved connections. Nevertheless, some modifications are needed because the assumption of equal connections is apparently not valid in multiple-service networks. Several approaches can be followed—some approaches provide either an accurate result with complex calculations; other approaches provide a less accurate result with simpler calculations. (See Chapter 18 in *Broadband Network Teletraffic* [Roberts *et al.* 1996].)

Dimensioning of Differentiated Services Networks

The expectation in Differentiated Services networks is, however, that the great majority of traffic is not using reserved connections (instead, that they are using best-effort kind of service). Evidently, the whole problem of network dimensioning is then completely different. There are no calls, no real holding times, and no call-blocking probabilities; instead, there are just variable streams of packets. Therefore, the traffic models must rely on a totally different approach that somehow takes into account the peculiar characteristics of data traffic.

Dimensioning of a packet network is such a complex task that it cannot be addressed extensively in this book. Yet, it is one of the primary tasks of any network operator. To get some level of insight regarding this issue, consider (as a starting point) the evaluation in section 4.4.3 "A Model for Evaluating Statistical Multiplexing", in Chapter 4. In that framework, the main target of efficient network dimensioning is to reduce the parameter ?. So what is the essence of parameter ? in Formula 6.11?

Formula 6.11

$$C(T) \mid M(0) = M(0)*\{1 + \gamma*[\varepsilon*(1-\alpha_b*\alpha_d)/(\alpha_b*N)]^{0.5}\}$$

Recall the meaning of the other parameters:

M(0) defines the known, current load level

Parameters α_b, α_d, and N are used to describe the characteristics of traffic process on the link.

γ defines the desired quality level

The laborious task of parameter ε is to take care of all difficulties and inaccuracies related to the real task of making real predictions. In particular, even though M(0) can be, in a certain sense, measured accurately, a lot of practical difficulties exist with the other parameters and the whole model. First of all, it should be stressed that α_b, α_d, and N are fictional parameters without any right values in real situations. Basically the same concern is valid with parameter γ; it is only a tool that can be used to satisfy a certain quality target without any guarantee that a certain value of γ leads to the same quality level in practice. Therefore, parameter γ tries to give a realistic estimation of effects related to all these factors.

In general, the better you understand the real process, the smaller ε is, and consequently, the better prediction you can make. In theory that is clear, but what can be done in reality to decrease ε Suppose, for example, that you have 10 links to be dimensioned with an original load of 50Mbps. Without an advanced information system, you must rely on your earlier experience with link dimensioning and traffic growth. You may suppose that the expected load of every link at the end of a half-year period is 100 higher than at the beginning of the period.

Further, based on previous experience, you may assume that the typical variance of the prediction ("error") is proportional to the mean value, V(t) = [M(t) / 2]^2. Moreover, you know that you need a safety factor ε = 4 to keep your customers satisfied. A simple calculation shows that you must reserve a capacity of 300Mbps for every link.

Note that you did not make any assumption about specific traffic parameters in this simple, but seemingly practical method. Now, you can just change the relation between V(t) and M(t) if the result is not satisfactory—that is, you reserved systematically too much or not enough resources. But that information, however useful, cannot essentially improve your ability to make traffic predictions.

You can, on the contrary, improve your prediction by acquiring more information about the load on individual links, because some part of the inaccuracy of prediction is probably caused by the differences in different links. You can classify the links into two groups according to the expected growth rate. If the expected load of the first group at the end of the prediction period is 70Mbps, and the expected load of the other group is 130Mbps, the variance related to this difference is (30Mbps)^2. The remaining part of the variance, (40Mbps)^2 (note that, 40 = (50^2 – 30^2)^0.5), can then be supposed to be a result of statistical traffic variations. In this case, the required capacities are as follows:

Group 1: C(1) = 70 + 4*40 = 230Mbps

Group 2: C(2) = 130 + 4*40 = 290Mbps

This result is certainly promising, but also somewhat surprising because even the links in Group 2 require less capacity than what was originally estimated. The explanation is that

additional information about the two groups increases significantly your knowledge of the situation.

The improvement can be achieved, however, only if the link classification is reasonable in the sense that the variation inside each group is actually smaller than that of the original group of all links. On the contrary, if the classification is a random process, the outcome is probably worse than the original situation without any grouping.

This phenomenon is actually easier to be thought of in the reverse order: First you have the situation with two groups, and then you lose information about the classification links according to the growth rate. Then if you know that there are these groups, but you do not know which group each link belongs to, you can use the value of 290Mbps. Finally, without any information about the groups, you need 300Mbps capacity for every link.

This example supposed that you could improve the knowledge about growth rates on individual links. Similar evaluation is valid with other parameters, α_b, α_d, N, or whatever parameters your model includes. The values of the parameters may vary from link to link, and if the differences are significant, traffic prediction could be improved by evaluating individual links or specific groups of links in specific ways rather than evaluating all links in the same way.

6.3.4 *Boundary Nodes Between Network Domains*

Differentiated Services provide different quality levels for packets, flows, or aggregate streams, and these levels may significantly affect the customers' perception of quality level. The basic assumption of Differentiated Services is that the use of the quality levels is controlled at the boundary node between customer and service provider. Yet the real situation is that a packet may traverse through network domains managed by different service providers.

Should the end users have an SLA with every service provider that ever conveys packets sent by the end user? No, that is not a realistic approach in the global Internet. The primary rule has to be that each user has a contract with one service provider (or perhaps with a couple of them), and that the service provider makes contracts with other service providers and network operators.

The primary issue is the form of the service contract between different service providers. Because this is relatively new territory—that is, not many "real" service providers are offering Differentiated Services yet, and therefore the future business models are still largely unknown—it is prudent at this point in time to just briefly consider some realistic scenarios.

There seems to be three main options for the contract between two service providers:

- *Pricing*: A device measures the traffic of each PHB in each direction. Based on the information gathered, either of the providers may pay the other provider compensation.
- *Re-marking*: An automatic system re-marks packets based on the load level on of all PHB aggregates.
- *Per-flow marking*: A similar marking as at the first boundary node between customer and service provider is also marked at the boundary nodes between network domains.

The last option could be feasible in some special situations, but it cannot be a common solution among all service providers. In addition, the approach needs a reliable signaling system to transmit relevant information among all boundary nodes. Therefore, it is not discussed further in this chapter

Pricing Approach

In essence, this approach means that the aggregate traffic of each PHB is measured in both directions. Let's denote the amount of bytes transmitted in PHB Class i on importance level j by X(i,j) in one direction and by Y(i,j) in the other direction. The compensation (Z) can be calculated from these factors—for instance, by a linear model such as that shown in Formula 6.12.

Formula 6.12

$$Z = \sum_{i,j} [c1(i,j) * X(i,j)] - \sum_{i,j} [c2(i,j) * Y(i,j)]$$

In case of two backbone operators, it is possible that $c1(i,j) = c2(i,j)$ for all i's and j's. Moreover, the operators may agree that no compensation is made if the traffic loads are approximately in balance.

On the contrary, when a small Internet service provider (ISP) is connected to a large backbone operator the situation could be totally different. The pricing could be based only on the traffic sent from the small ISP to the backbone network, for example. It should be stressed that all these issues relate to business models of service providers and network operators; they are beyond the scope of any standardization process.

Re-Marking Approach

As with the pricing model, the traffic load of each PHB should be measured individually. Instead of pricing, the network may re-mark the packets if certain thresholds are exceeded. Although there are numerous alternatives, some rules seems to be relevant across the board:

- Under normal situations, re-marking should be avoided, because it may negatively affect the overall service.
- Re-marking should be done, if possible, only within a PHB class.
- If a PHB class has only one importance level, and re-marking is needed, the default service (used to transmit best-effort traffic) is the secondary choice.
- If there are enough importance levels, it is better to re-mark all packets systematically (one step up or down) than to re-mark packets here and there; the systematic approach helps keep the network service as consistent as possible.
- Although re-marking a packet can be based on the load level of the PHB class used by the packet, it is sometimes more reasonable to use the total load level with appropriate weights for each PHB.

It should be stressed that these rules are tentative, and without practical experience it is impossible to state what is the most practical approach.

You can also combine the pricing and re-marking approaches in several ways. If the pricing model is your primary approach, for instance, you could also apply load limits to the highest PHB classes to limit the possible problems related to PHB marking within a network domain.

Summary

This chapter discussed several integral issues related to Differentiated Services. The implementation of traffic classification, metering, marking, shaping, and dropping essentially determine the service structure. Correspondingly, the requirement of consistent, fair, flexible, and efficient service makes it necessary to carefully design traffic-handling functions.

Section 6.2, "Traffic Handling Functions in Interior Nodes," dealt mainly with the different buffering systems, including FIFO, priority queuing, SFQ, and CBQ. In Differentiated Services networks, the main constraint for the application of queuing systems is that individual flows are not supposed to be discernible; instead, all the mechanisms should use the DS codepoint in the packet. For this reason, the discussion focused mainly on the application of simple queuing systems in a Differentiated Services environment.

FIFO is the basic choice because of the plain implementation; priority queuing and Class-Based Queuing (CBQ) provide quite straightforward means to solve some of the fundamental problems of FIFO systems. This book recommends always avoiding complex systems with a large number of adjustable parameters, however, because although they can be realized, they tend to require additional management and are prone to configuration errors.

The main dropping disciplines are to drop packets just when the buffer is full, have several thresholds for different importance levels, and RED. Different combinations were also discussed.

The first, simple dropping, is not a sufficient mechanism for implementing Differentiated Services. A threshold for every importance level is an apparent and feasible approach to provide quality differentiation. Moreover, RED seems to be the recommendable mechanism to further improve the service performance when the TCP protocol is used in customer equipment. The usefulness of RED in a queuing system with several importance levels is not totally clear, however.

Finally, some elementary functions related to traffic management were discussed in Section 6.3, “Functions Related to a Network Domain.” These discussed functions included network dimensioning and contracts between service providers.