Traffic Prioritization Subject to Cost

Magnus Ekstrand

M. Sc. Thesis

Master's Thesis Project Report by Magnus Ekstrand, department of Teleinformatics, Royal Institute of Technology (KTH), Stockholm as an exchange student with the Swiss Federal Institute of Technology, Lausanne (EPFL).

March 29, 1999

Tutors: **Gerald Maguire**, Professor of Computer Communication, KTH, Stockholm
Jean-Pierre Hubaux, Professor, EPFL, Lausanne
Jean-Philippe Martin-Flatin, Research Assistant, EPFL, Lausanne
Georges Biro, Industrial Advisor, Adventis Communications Engineering SA

Executive Summary

In order to carry real-time traffic like video, IP networks—especially intranets—recently began to deal with different levels of priority for communication flows. Traffic prioritization is the key to meeting the demands of real-time traffic, which are much more stringent than mere data traffic, such as file transfers. Now that the question: "How should we specify different levels of priority?" is reasonably understood, the challenge is to answer this new one: "How should we charge for these different levels of priority?"

This document discusses the introduction of Quality of Service (QoS) in the corporate network of a large industrial enterprise. Its main contributions are a charging model for different types of traffic with different levels of priority, and a network simulation for verifying the impact of a QoS implementation.

Résumé

Afin de pouvoir transporter le trafic temps réel comme la vidéo, les réseaux IP, notamment les intranets, ont récemment commencé à intégrer différents niveaux de priorités pour les flux de communication. Par ce biais, il est possible de satisfaire les exigences de ce type de trafic qui sont bien plus contraignantes que celles affectant un trafic de type transfert de données. Après la question : "comment spécifier différents niveaux de priorités ?" se pose aujourd'hui la question : "comment facturer l'utilisation de ces différents niveaux de priorités ?"

Cette thèse traite de l'introduction de la Qualité de Service (QoS) dans le réseau IP d'une grande entreprise industrielle. Elle présente un modèle de facturation différenciée pour différents types de trafic plus ou moins prioritaires, et une simulation de réseau pour vérifier l'impact de l'implémentation d'un modèle de QoS.

Acknowledgments

This thesis benefited from the help of many people. I would like to thank Professor Jean-Pierre Hubaux for letting me conduct my thesis at the ICA institute at EPFL, and Professor Gerald Maguire for his support for my thesis at KTH. I would like to thank Georges Biro and Adventis for offering me a valuable industrial experience and competent support. I am thankful to André Rudin, Moritz Gyssler, Ivo Maritz and Greg Bassett at Roche for welcoming me in their projects. I am very grateful to Jean-Philippe Martin-Flatin, who has provided great support and valuable comments during our discussions throughout my thesis.

Table of Contents

1. Introduction

1.1. Background

Traffic on corporate networks is increasing fast. This is partly due to the introduction of many new applications in the market offering services such as videoconferencing, multimedia, and other bandwidth-demanding services. It may also be due to the increasing use of the Internet and the fact that more and more tasks are performed by computers. In addition, today's applications are more demanding, largely because there is an important cost trade-off between fast and efficient programming.

This increase in the traffic negatively affects the performance of corporate networks. During peak hours, this can result in congestion and therefore in insufficient service levels for business-critical applications. To support this new amount of traffic and, more importantly, to ensure the operation of critical applications on the intranet, some measures must be taken.

There are different approaches for solving these problems. The first solution that springs to mind is increasing the bandwidth of the network by allocating more resources. This would, of course, solve the problem in the short term. Since a corporate network is often differently loaded during different times of the day, with peak hours occurring during office hours, we need to increase the bandwidth to support the maximum load under the worst conditions to be able to guarantee proper operation. This can result in an inefficient use of resources, so this we explored complementary solutions in this thesis.

To be able to guarantee a certain level of service to business-critical applications, we need to introduce Quality of Service (QoS) in the network. QoS aims to guarantee that a certain application is provided with the network resources that it needs for proper operation, e.g., bandwidth and response time. During peak hours, critical applications must have priority over less critical tasks such as, in many cases, Web browsing or FTPs. There are many different ways of implementing QoS in a network; we will explore some of the most popular.

When we let some applications have a higher priority than others, we must also introduce some incentives to prevent the abuse of the high priority class. Otherwise, we might come to a stage where all applications request the highest priority, which would bring us back to square 1. This thesis explores the introduction of usage-based charging on a corporate network. Network resources are very expensive to an enterprise, and their use should be divided between its different business units. By introducing usage-based charging in the network, these costs can be fairly shared on a usage basis. Clearly, fairness is a function of the corporate aims and the pricing formula used.

The introduction of QoS and charging requires a well-defined policy for the network. The enterprise must formulate a policy for the use of its network in a cost-efficient way. There are many different parameters to network policing. What applications should be allowed a certain QoS? At which hours are which users allowed to use how much bandwidth of the network? Who should be charged and at what price? There are other aspects to policy-based networking such as security, but these are beyond the scope of this thesis.

This project aims to clarify the need for introducing QoS and charging in a large corporate intranet. It will focus practically on tests for implementation of QoS and charging in a large multinational enterprise.

1.2. Project specification

This diploma work was carried out at and in collaboration with two enterprises: Adventis Communications Engineering and F. Hoffmann – La Roche (Roche). Adventis is a consulting company specialized in telecommunications. Roche is an international enterprise in the pharmaceutical business. The objective was to assist in two related projects: the introduction of QoS in the network and charging for network usage.

In the QoS project, the goal was to set up a test environment in a network simulator for verifying the effects of introducing QoS in the network. In the charging project, the goal was to investigate methods for implementing charging on the corporate network and to model an implementation scheme.

1.3. Structure of the report

Chapter 2 presents an overview of Roche's corporate network.

Chapter 3 describes what QoS is and why it is important. We present the implementations that exist today.

Chapter 4 discusses how QoS could be introduced in Roche's corporate network.

Chapter 5 shows a simulation test suite for verifying the impact of introducing one QoS model in Roche's corporate network. This simulation is done with a network simulator called `ns`.

Chapter 6 presents the charging models that exist for IP networks today.

Chapter 7 discusses how charging could be introduced in Roche's corporate network.

Chapter 8 presents our conclusions and discusses future work.

2. Presentation of Roche's Corporate Network

2.1. Roche in brief

The Roche Group is one of the world's leading research-based healthcare groups active in the discovery, development and manufacture of pharmaceuticals and diagnostic systems. The Group is also one of the world's largest producers of vitamins and carotenoids and of fragrances and flavors. The majority interests in Genentech, one of the world's leading firms in biotechnology, strengthen Roche's position in the healthcare market.

The activities of the Group in the areas of pharmaceuticals, diagnostics, vitamins and fine chemicals as well as fragrances and flavors focus on the prevention, diagnosis, monitoring and treatment of diseases and on the promotion of general well-being.

2.2. Physical Network Structure

Roche's corporate network, henceforth referred to as the RCN, consists of several large area sites interconnected world wide as shown in Figure 2-1. Smaller area sites are, in turn, connected to a larger area site. The RCN comprises roughly 500 routers and 300 Frame Relay lines plus a number of leased lines. We count about 40.000 end users in the complete network. Investigations have shown that to support a number of users with application server services, printers and other networking equipment, we need about 1.3 host stations per user. In the entire RCN we have about 52.000 host stations.

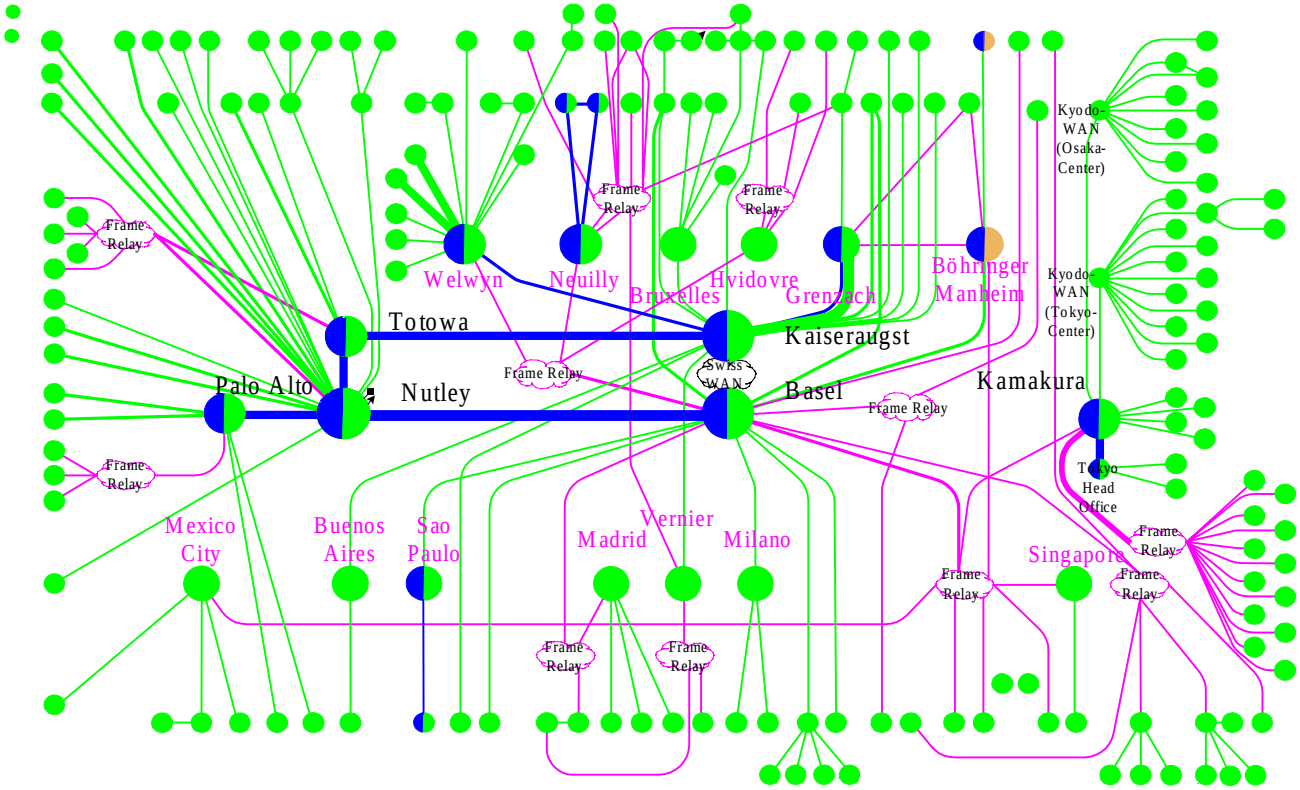

Figure 2-1: Roche's Corporate Network

The site at which this thesis is carried out is located in Basle and Kaiseraugst in Switzerland. The LANs of the two locations are interconnected via an ATM backbone and form one site. The physical structure of this site is shown in Figure 2-2. This site has 7600 end users and about 10.000 host stations. The routers in the squares marked with core, connects this site with the rest of Roche's Corporate Network.

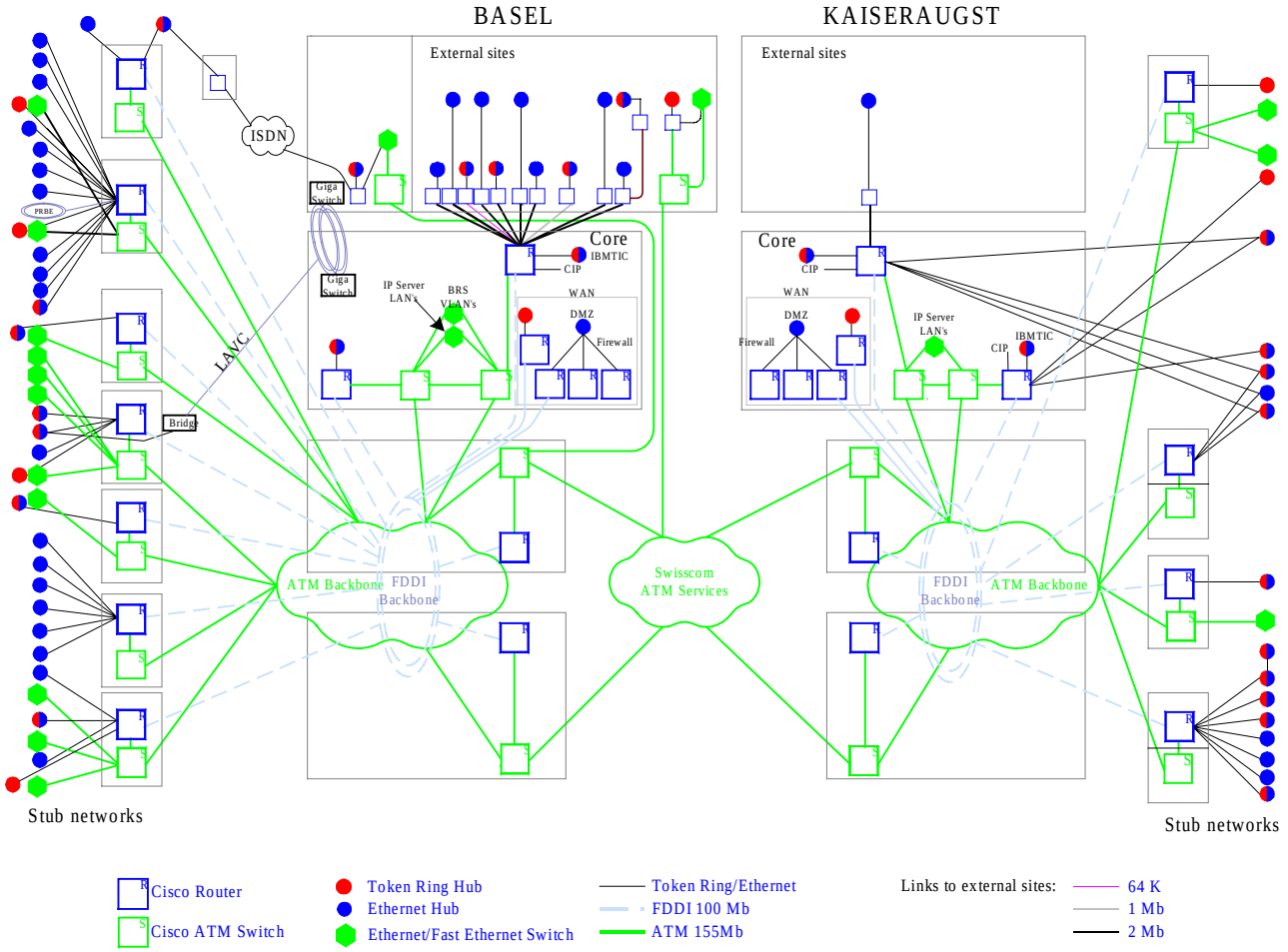

Figure 2-2: Basle/Kaiseraugst SiteLAN

For what concerns network equipment, Roche is a Cisco customer and uses almost exclusively Cisco routers. For end systems, Roche has a Windows NT environment.

2.3. Protocols and applications used over the RCN

The Roche network is almost exclusively an IP network. According to Roche's Network Management there is, however, a fraction of traffic, which is less than 10% and decreasing, that belongs to other protocols. These protocols are DECnet, IPX, and Appletalk.

There are several large applications used over the RCN. The most important are SAP/R3 and several Oracle database applications. E-mail and use of the Internet and the corporate intranet (WWW, FTP, Telnet, Usenet, etc.) represent other shares of the traffic over the RCN.

For what concerns HTTP Internet or intranet traffic, there are proxy servers set up at larger sites. IP addresses change when traversing the proxy server, why the real end user of a traffic flow can not be determined. The share of traffic that traverses the proxy server in Basle/Kaiseraugst is about 5-10% during daytime, and normally less than 1% during night.

2.4. IP address allocation scheme

Roche is assigned four class B addresses and some class C addresses for their network. A range of IP addresses is assigned to a certain site, and the site can then decide how to divide this range of addresses into local subnets. The information about IP address allocation to a specific network device is kept updated in a Network Management Tool (NMT), which is an Oracle database with a front-end tool called Remedy. There is no policy saying that a site has to register what IP addresses are assigned to a certain network device, and therefore this is not always the case. According to Roche Network Management it is possible to resolve the link between IP address and user name / cost center for about 80% of the addresses today. The other 20% of the addresses are locally registered in databases or spreadsheets that are not connected with the NMT used globally.

For the assignment of an IP address to a specific network device, there are two cases that have to be distinguished:

1. DHCP administered addresses are used to obtain an efficient address allocation, since the allotted classes of IP addresses are sparse for the RCN. Also, the use of DHCP simplifies the task of keeping records of assigned addresses since it is done automatically by scripts that communicates this information between the DHCP server and the NMT. DHCP is mostly used for end user devices such as PC’s, workstations, etc. The default IP address lease time is 7 days.

2. Statically assigned IP addresses. These are usually servers and routers in the network.

Since different business units share the same office area, IP subnets could be shared. This means that an IP subnet could be shared also between cost centers. It is important to remember this when choosing granularity for the charging model.

3. QoS in IP Networks

3.1. What is QoS?

Quality of Service is the term that describes the methods for introducing a differentiated service model in the networks.

All applications used over the network are not created equal. Some applications need more predictable service than others do, as for instance interactive applications such as telnet. Some applications are more sensitive to delay or delay jitter than others are, as for instance telephony over the Internet. Some applications are very sensitive to packet loss, as for instance router configuration messages via ICMP. It is clear that we might get our network to work better if we somehow could organize all this instead of just sending everything into the network on a first come first served basis.

QoS is the term used for this organization. By differentiating one type of traffic from another we could provide them with different service levels. There are several methods for doing this, and we will describe the most popular further down in this chapter.

3.2. Why is QoS important?

Good functioning of the applications used on the company intranet is crucial for business. A company can not afford badly functioning of business-critical applications.

Traffic on company intranets is increasing very fast. This is probably due to the introduction of a large amount of new applications in the market offering services such as videoconferencing, multimedia, and other bandwidth-hungry services. It may also be due to the increasing use of the Internet.

Network equipment and links are costly resources in a corporate network. The solution for supporting higher amounts of traffic might not always be to increase the size of the pipe once the network gets congested. As we must deal with the network congestion in a cost-efficient way, this thesis explores an alternative solution.

Apart from supporting the needs for the applications, there is also a policy point of view. Business policy might aim to give certain business-critical applications higher priority or even reserve bandwidth for them. Such a management facility is commonly called “controlled link-sharing”.

3.3. How is QoS measured?

Packets in a flow from a sender to one or more receivers will be affected by network characteristics on the way. There are four very important characteristics of a packet flow; **bandwidth, delay**, **jitter**, and **reliability,** as described in [19].

Bandwidth is the maximal data transfer rate available to a flow between a sender and a receiver. The upper bound of the bandwidth available is the physical link capacity of the link with the lowest capacity on the path between the end-points in a simple topology. In more complex topologies several links could run in parallel between end-points and the upper bound on the bandwidth gets

more complicated to calculate. Other flows may also be using parts of the path between the end-points, reducing the bandwidth available to the flow in question.

Delay is the time it takes for a packet to travel from a sender, through the network, to the receiver. Delay is due not only to transmission links, but is also increased by router holding time. If the packet has to be queued in the router it will experience higher delay.

Jitter is the variation of the delay. Mathematically it is the absolute value of the first derivative of the sequence of individual delay measurements.

Reliability measures the probability that the data arrives properly at the receiver. Errors can be introduced either on the physical link layer where a bit or a number of bits get changed during transmission, (bit-errors and burst-errors). Or, routing and protocol processing in the system can introduce degradation in reliability, as the order of the packets can be changed (packet reordering) or packets can be lost (packet loss).

These are characteristics that are directly measurable and that can be modeled mathematically. Now we have to remember that behind each flow is a transfer protocol and an application. Different types of protocols and applications behave differently when encountering the limitations described above. Taking the jitter parameter as an example: A user doing a file transfer would not experience any degradation in quality, although the TCP protocol might work a bit inefficiently. A user talking on the telephone over IP, on the other hand, would experience degradation in quality due to the loss of signal.

Tools are evolving to measure application specific response-times, taking into consideration the user need at the moment. Examples of such tools are ETEWatch from Candle Corp. and Smartwatch from Landmark Systems Corp. Packages that work only with one specific application are for example Stopwatch Pro from Envive and Luminate for SAP/R3 from Luminate Software Corp, both of which analyze SAP R/3. DataCom [27] recently published an evaluation of response-time measurement tools.

3.4. Congestion avoidance

The standard way an IP-network is normally configured is for best-effort traffic. This means that no measures are taken to separate one type of traffic from another. All packets are competing on the same basis. Buffering is done at the intermediate stations (e.g., routers) and packets are forwarded in a FIFO manner. Routing mechanisms try to make sure that the packet travels the best way through the network.

It should be noted that there are fields in the IPv4 header to specify QoS levels in terms of delay, throughput, and reliability [33]. These are bits 3-5 in the TOS field and they can be used for specifying the delay to be normal or low; the throughput or the reliability to be normal or high. However, very few applications set these QoS parameters.

The best-effort scenario with FIFO forwarding works ok until the point when the network or a link gets congested due to traffic overload. At this point the routers will start to drop packets, since their queues are finite. This might lead to an even worse situation where applications try to retransmit data and possibly introduce an even larger load on the network. However, there are already some mechanisms that deal with this problem.

3.4.1. TCP Rate Control

The TCP protocol [34] has some basic mechanisms to avoid introducing more congestion to the network when losing packets. It is called TCP rate control. The key parameter for this behavior is the congestion window (*cwnd*). This parameter defines the number of segments that a sender can transmit without receiving an acknowledgement (ACK) from the receiver. The first control function is the *slow start* of a TCP connection. When initializing a TCP connection the sender transmits only one segment. Each time the sender receives an ACK from the receiver, the congestion window (*cwnd*) is increased by one segment size. This effectively doubles the transmission rate for each round trip time (RTT) cycle. A TCP connection starts with an initial *cwnd* value of 1, and a single segment is sent into the network. The sender then awaits the reception of the matching ACK from the receiver. When received, the *cwnd* is increased from 1 to 2, allowing two packets to be sent. When each ACK is received from these two segments, the congestion window is incremented. The value of cwnd is then 4, allowing 4 packets to be sent, and so on.

The other mechanism is called *congestion avoidance*. In the event of packet loss, as signaled by the reception of duplicate ACK's, the value of *cwnd* is halved, and this value is saved as the threshold value to terminate the *slow start* algorithm (*ssthresh*). When *cwnd* exceeds this threshold value, the window is increased in a linear fashion, opening the window by one segment size in each *RTT* interval. The value of *cwnd* is reduced to 1 when the end-to-end signaling collapses and the sender times out waiting for ACK packets from the receiver. Since the value of *cwnd* is below the *ssthresh* value, TCP switches to *slow start* control mode, doubling the congestion window with every *RTT* interval if the ACKs are getting back.

The intent of the algorithm is to reach a steady state where the sender injects a new segment into the network at the same rate at which the receiver accepts a segment from the network. The algorithm works fine for longer data flows such as a file transfer. Shorter flows, such as Web browsing via HTTP, are not likely to reach this point.

The major disadvantage of this scheme is that when several TCP connections are competing over a congested line, they all could experience packet loss at approximately the same time, which leads to what is called *global synchronization*. All flows decrease their transmission rates and invoke the TCP slow start mode. When increasing the transmission rate by doubling the *cwnd* the point of congestion might be reached again, repeating the process. A byproduct of this is a large number of retransmissions, which could ultimately result in complete congestion collapse.

3.4.2. Random Early Detection

Random Early Detection (RED) [20] is a queue managing mechanism used for lowering the risk of global synchronization. The idea is to start dropping packets when a queue reaches a threshold value instead of waiting until it is full and drop the tail of it. Instead of dropping just any packets, RED selects randomly individual TCP flows and drops their packets. The result is that the sender of those flows invokes the TCP slow start mode. The advantage is that this does not happen to all flows at the same time, thus avoiding the global synchronization.

It should be noted that RED is a very simple queuing mechanism. It does not require much computational overhead as other queuing techniques do. With RED, it is simply a matter of deciding who gets into the queue in the first place – no packet reordering or queue management takes place. When packets are placed into the outbound queue, they are transmitted in the order in which they are queued.

Cisco has implemented a weighted RED mechanism, WRED [11]. The idea with this extension to the mechanism is to differentiate between TCP flows. WRED drops packets in a weighted scheduling order, where packets of low priority flows are more likely to be dropped than packets of higher priority flows. The ability to set a certain priority to a flow is discussed further in the differentiated service model described in section 3.5.1.

3.4.3. Traffic Shaping Non-Adaptive Flows

So far we have discussed only the TCP transport protocol which is an adaptive protocol due to its use of feedback. As described above, the TCP sender slows down its transmission rate in case of congestion. However, there are applications using other protocols than TCP above IP. UDP [32] is the transport protocol used by many real-time multimedia applications. UDP is a connectionless transport protocol and is not concerned about recovering lost packets or reordering packets. It transmits the data when received by the application and does not slow down because of packet loss. It is easy to imagine that this type of protocol can easily generate congestion in a network unless some precautions are taken.

What could be done to those flows is rate shaping the traffic when it enters the network. This way we can limit the flow to a certain maximum bandwidth. Traffic shaping can also be used to decrease the jitter in the flow, by making sure to release the packets with the same time space in between them. The price we pay for a decrease in jitter is an increase in delay, since we have to delay to shape.

A very common model for doing traffic shaping is the leaky bucket model [36]. We imagine a bucket with a hole in its bottom. The user's offered load to the network is modeled as what is poured in to the bucket. The load that is accepted by the network is represented by what is coming out of the hole in the bottom. If the user offers more load than what is accepted by the network, he will start to fill up his bucket, which is synonymous with storing his data packets in a queue. When the bucket is full, it will flow over. This is represented by a full queue with packet drops as a consequence. We realize that the user can never exceed a certain transfer rate represented by the size of the hole in the bottom of the bucket.

Another similar model is the token-bucket model [31]. In this model our bucket contains tokens that are produced at a fixed rate. For each packet, or for a certain volume, that a user offers to the network, he must use a token. If he does not offer a load to the network, the bucket will fill up with tokens that are not used, until it is full. The full bucket represents the maximum burst size that the network will accept from the user. If the user tries to transfer data at a higher speed than tokens are generated, the bucket will eventually be emptied. When there are no tokens to use for an offered packet, it is discarded. The difference from the leaky bucket model is that the user is allowed to transfer at a speed higher than the transfer rate represented by the token generation speed, but the mean value of the accepted load will always be equal to or below this value.

3.5. QoS models

The above mechanisms help us use our network efficiently, but they do not let us differentiate certain traffic from other. Differentiation would be preferable, since different users or applications have different needs. The following paragraphs discuss the two most important QoS models for introducing different levels of service for network traffic: differentiated services and integrated services.

3.5.1. Differentiated Services

There is an IETF Working Group called DiffServ [23] that is working with a differentiated services model on the Internet [4]. Differentiated services aims to give some traffic flows better service than others. To distinguish between flows we need a way to mark packets with a priority. The priority can be used either to let some traffic flows have precedence over others by reordering packets in the queue so that higher priority packets get sent first. Or, the priority could serve as a discard preference. A packet with a lower priority level would be discarded more easily in the event of congestion than a packet with higher priority.

The most common way to mark packets is via IP Precedence, as described in the initial Internet Protocol RFC [33]. A subfield of the TOS (Type of Service) octet in the IPv4 header is used for this. The three leftmost bits in the TOS field are used, giving a possible maximum of eight different priority levels. The priorities are set as 0 for lowest priority and 7 for highest. The IETF Working Group proposes as an extension to use the complete TOS-field as priority field. They propose to call the field for the Differentiated Services (DS) field, [29]. Six bits are used for setting priority, giving a possible maximum of 64 different priority levels. The two leftover bits are reserved for future use.

The idea of differentiated services is to set the priority level of the packet when it enters the network. This is done by a *packet classifier*. A packet can be classified as a certain priority by examining its source and destination IP address and/or by other information found in the packet header such as a TCP or UDP port number. The network manager has to define a policy for what traffic should be classified as what priority level, then the core network has only to implement scheduling algorithms to queue and forward the packets to their destinations. It is especially important to apply those mechanisms at network bottlenecks, as it is in these areas where congestion is most likely to occur.

The basic modification of the single level FIFO queuing algorithm to enable differentiated services is to divide traffic into a number of categories, and then provide resources to each category in accordance with a predetermined allocation structure, implementing some form of proportional resource allocation.

A basic modification of the FIFO structure is to introduce *Priority Queues*. The idea is to create a number of distinct queues for each interface and associate a relative priority level with each one. Packets are scheduled from a particular priority queue in FIFO order only when all queues of a higher priority are empty. In such a model, the highest priority traffic receives minimal delay, but all other priority levels may experience resource starvation if the highest precedence traffic queue remains occupied. See Figure 3-1 for the scheduling principle. Note how the low priority traffic gets completely starved by the two higher priority traffic flows. This model is simple to implement, but to ensure that all traffic receives some level of service we need more sophisticated scheduling algorithms.

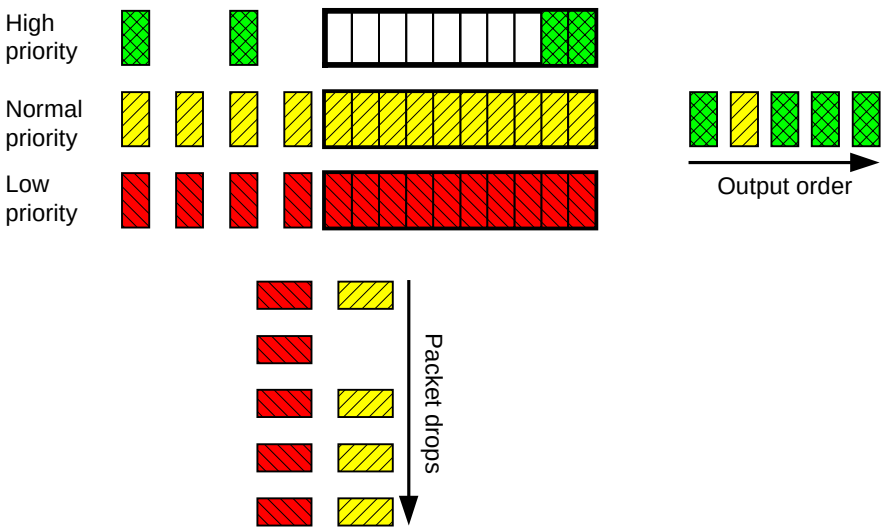

Figure 3-1: Priority Queuing

A more sophisticated method would be to classify each traffic flow as belonging to its own queue. This could be done by differentiating flows by criteria such as source and destination address, source and destination port, ProtocolID, and TOS field. The router assigns each flow its own queue. It then applies its scheduling mechanism to these queues, so that the packets gets scheduled on a per flow basis.

We would service each queue in order to its relative weight. This approach is called General Processor Sharing, GPS. The equation for calculating the bandwidth received for each flow would be:

$$BW_{MyFlow} = \frac{(\Pr ec_{MyFlow} + 1)}{\sum_{i} (\Pr ec_{Flowi} + 1)} \times BW_{total}$$

BW_{MyFlow}	The share of the bandwidth allotted to the flow in question.
$Prec_{MyFlow}$	The IP precedence set for the flow in question
$\sum_{i} (\Pr ec_{Flowi} + 1)$	The sum of the (IP precedence + 1) for all flows.
BW_{total}	The total bandwidth available on the link.

We could use a *weighted round-robin* scheduling algorithm to service each queue. As an example we use four queues with priority 3, 1, 0, and 0. Our weighted round robin schedule will send four packets from queue number one, two packets from queue number two and one packet each from queues number three and four. Then we start over by sending packets from queue number one again. See Figure 3-2 for the scheduling principle. When packets are equally large, this mechanism fairly shares the bandwidth between all flows on our link. When packet sizes vary we would come to a situation where a flow with larger packets would occupy more bandwidth than a flow with smaller packets of the same priority. To avoid this, we would be better off using a *deficit weighted round-robin* algorithm, which modifies the round robin algorithm to use a service quantum unit. This is also known as a *bit-wise round-robin* algorithm. A packet is scheduled from the head of a weighted queue only if the packet size minus the per-queue deficit counter is less than the weighted

quantum value. The next packet in the queue is tested using a weighted quantum value, which has been reduced by the size of the scheduled packet. When the test fails, the remaining weighted quantum size is added to the per-queue deficit counter and the scheduler moves to the next queue. This algorithm performs with an average allocation that corresponds to the relative weights of each queue on a bandwidth basis.

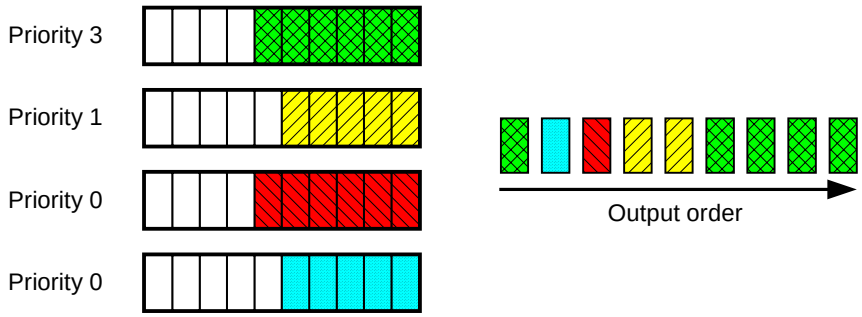

Figure 3-2: Weighted Round Robin Queuing

Another fair queuing algorithm with the possibility to weight packets is *Weighted Fair Queuing*, WFQ. WFQ introduces the concept of finishing time of a packet. Instead of serving each queue in a round-robin fashion, it serves the packet with the smallest finishing time first. Finishing time is calculated as follows:

If a flow is active (i.e., there are already packets in the queue for this flow.):

$FT(Pkt_{k+1}) = FT(Pkt_k) + Size(Pkt_{k+1})*(\text{transmission time in secs/byte})*(\text{active flows})$

Pkt_{k+1} is the first packet in this queue. Pkt_k is the last packet that got sent away from this queue. $FT(Pkt_x)$ is the calculated finish time for packet x.

Otherwise;

$FT(Pkt_0) = Now + Size(Pkt_0)*(\text{transmission time in secs/byte})*(\text{active flows})$

In the example shown in Figure 3-3 there are two active flows. I have normalized the $Size(Pkt_{k+1})*(\text{transmission time in secs/byte})*(\text{active flows})$ to the numbers between brackets to avoid a too complicated equation. The right end of the packet corresponds to the arrival time on the time-line. We note that the packets A2 and A3 get scheduled before the packet B1 although they arrived later.

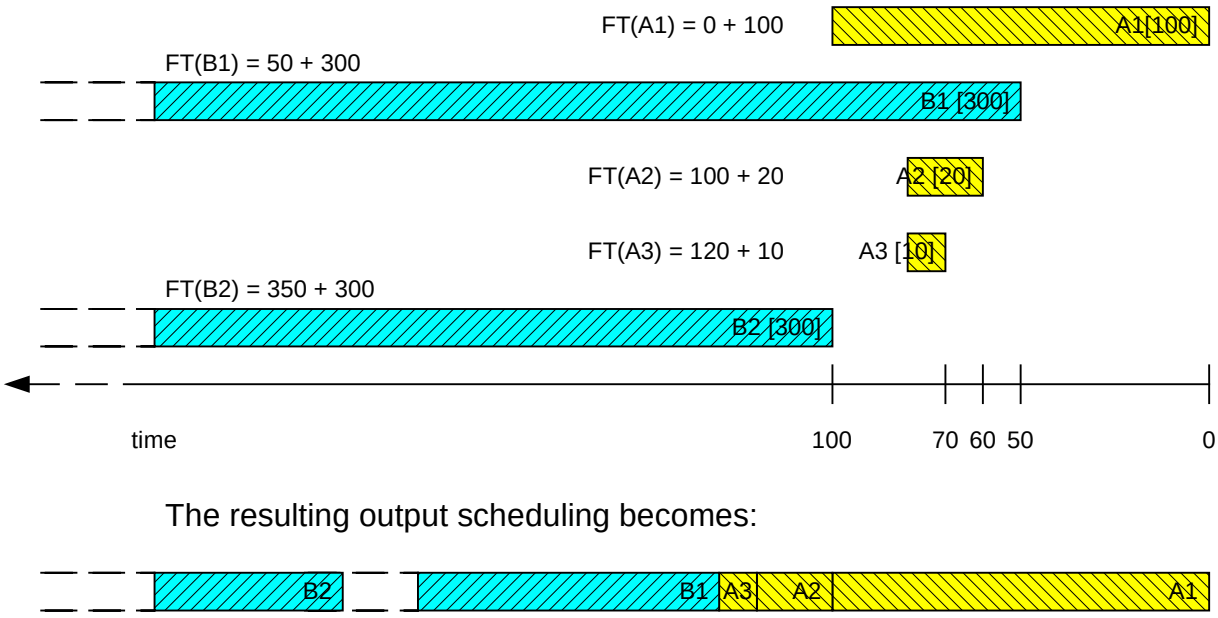

Figure 3-3: Weighted Fair Queuing

This gives the effect that flows with smaller packets, which is often the case for interactive traffic flows, get served before flows with larger packets. Note that we do not assign more bandwidth to these flows, though, this mechanism only prevents smaller packets from getting stuck after large packets, as they could do in a round robin scheduling discipline. Also, the benefit for a low bitrate flow is that when a packet belonging to this flow arrives, the WFQ mechanism will schedule it quite soon, since the last packet in its flow was scheduled quite some time ago.

It has been proved by Parekh [30] that this algorithm provides an absolute upper bound on the network delay on multihop networks. Given that WFQ is used at every hop for a particular data flow and the traffic injection source conforms to certain token bucket model assumptions, it has been shown that the worst case queuing delay is bounded within a network. An important benefit of this is that WFQ can be used to provide strong guarantees for a given data flow within a heterogeneous networkApparently there seem to be different definitions of WFQ - those who take priority into account and those who do not. The above equation does not take priority into account, but there is a Cisco implementation of WFQ [13] that does. The Cisco WFQ model checks for the IP Precedence bits in the TOS field of the IP packet and assigns bandwidth according to the priority of the packet. A flow with higher precedence will receive more bandwidth than flows with lower precedence. To achieve this, the equation for the finishing time must be changed. Cisco implements this by the following equations:

If a flow is active (i.e., there are already packets in the queue for this flow.):

$FT(Pkt_{k+1}) = FT(Pkt_k) + Size(Pkt_{k+1})*(\text{transmission time in secs/byte})*4096/(Prec + 1)$

Pkt_{k+1} is the first packet in this queue. Pkt_k is the last packet that got sent away from this queue. $FT(Pkt_x)$ is the calculated finish time for packet x. Prec is the precedence set for the packet.

Otherwise;

$FT(Pkt_0) = Now + Size(Pkt_0)*(\text{transmission time in secs/byte})*4096/(Prec + 1)$

Cisco puts the flows into a hash table, using the classification criteria described above. The value 4096 is the maximum number of entries in this hash table. This value can be set between 16 and 4096, where 256 is the default, to tune the functionality of the WFQ mechanism.

The result of the differentiated service model is that some traffic gets treated better than other traffic. It should be remembered, though, that there are no guarantees for a certain service level. The share of the bandwidth received is still dependent on what other traffic is competing on the network.

3.5.2. Integrated Services

Integrated services is the term used for an Internet service model that includes best-effort service, real-time service, and controlled link sharing. The aim of the integrated service model is to provide guarantees for a certain service level to a traffic flow. This is done by reserving resources for it.

The IETF Integrated Services Working Group [24] has proposed an extension to the Internet architecture and protocols to provide integrated services [5]. This proposal does not suggest any changes to the Internet architecture of today, but rather an extension. They suggest running real-time services on the same infrastructure as non-real-time data. They also suggest using the same protocol - the IP protocol.

The major change to the Internet service model of today is the introduction of resource reservation. Intermediate network stations (routers) along the path from the sender to the receiver must have the ability to reserve resources in order to provide special QoS for specific traffic flows. With resource reservation it is possible to give guarantees of a certain service level to a traffic flow. It should be noted that the introduction of resource reservation makes the integrated service model very different from the differentiated service model. In the latter we can not give any guarantees of delay or bandwidth – we can only guarantee that some traffic gets treated better than others.

To be able to setup a resource reservation scheme along the path we need a setup mechanism. This mechanism is suggested as a standalone protocol that sets up the service level agreement between the end stations and the network. One such protocol is the Resource Reservation Protocol (RSVP) [6]. RSVP carries information about the desired QoS. This information is called the *flowspec* and it is passed to an admission control agent at each router. The admission control agent then decides whether it is possible to reserve the resources needed for this service on the relevant links. If all routers along the path can reserve the resources needed, then the connection can be setup and transmission can start. The router will then translate the flowspec into packet classification and packet scheduling mechanisms that provide the desired QoS.

RSVP also handles the case of multicast transmission. In this case each receiver of the multicast transmission can specify what service level he wants.

The definition of the *flowspec* is opaque to the resource reservation protocol and can therefore be implemented using different schemes. [35] lists a number of parameters used to characterize a traffic flow, which can be used as a *flowspec* in RSVP.

Integrated services and RSVP are not yet widespread in usage. Before they can be used in a larger implementation, applications must become QoS aware. Applications must be able to ask for a certain level of service. The other approach would be to classify traffic at the ingress point to the network and start the RSVP signaling from there, but it is difficult to infer a QoS level for a flow without knowing anything about the application. This might work in some cases, but the inference might not always be correct. The network device doing this is called an RSVP proxy.

The scaling properties of RSVP and *Integrated Services* have not yet been thoroughly tested. Since it requires a setup mechanism that must apply all along the path of the flow one could imagine some scaling problems across the Internet. On the other hand, RSVP may scale very well for a private network. One could imagine implementing RSVP in a corporate network, provided the applications are QoS aware.

3.6. QoS in IPv6

IPv6 [17] introduces more elaborate possibilities for QoS. In the proposed standard of December 1998 there is a Traffic Class field of 8 bits and a Flow Label field of 20 bits that can be used for QoS purposes. In the RFC 2460 we find the following two paragraphs:

“The 8-bit Traffic Class field in the IPv6 header is available for use by originating nodes and/or forwarding routers to identify and distinguish between different classes or priorities of IPv6 packets. At the point in time at which this specification is being written, there are a number of experiments underway in the use of the IPv4 Type of Service and/or Precedence bits to provide various forms of "differentiated service" for IP packets, other than through the use of explicit flow set-up. The Traffic Class field in the IPv6 header is intended to allow similar functionality to be supported in IPv6.”

“The 20-bit Flow Label field in the IPv6 header may be used by a source to label sequences of packets for which it requests special handling by the IPv6 routers, such as non-default quality of service or "real-time" service. This aspect of IPv6 is, at the time of writing, still experimental and subject to change as the requirements for flow support in the Internet become clearer. Hosts or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, pass the field on unchanged when forwarding a packet, and ignore the field when receiving a packet.”

3.7. QoS in non-IP technologies

Since this project is about IP networks, we will not go into detail in QoS implementations in other protocols. It should be noted though, that there are several other protocols supporting some type of QoS. Some of the most popular are ATM, Frame Relay and IEEE 802.1p, which are all link level protocols. If these technologies are used in an IP network, we need to translate IP QoS into non-IP QoS, to be able to use the facilities that these protocols offer.

4. QoS in Roche's Corporate Network

4.1. Motivation

There are several applications considered as business critical on Roche's Corporate Network. Considered the increasing load on the network there is a wish for introducing a more consistent service for these applications, especially during peak hours. The desire is to have fast response times for the applications used. The applications that are considered as business critical at the moment are: SAP, Oracle Clinical, and two more Oracle database applications. There is no intent to differentiate between users.

4.2. Differentiated Services in the RCN

The model chosen by Roche to evaluate is the differentiated service model using IP Precedence and WFQ. Using the differentiated service model on the RCN would mean assigning the above applications a certain priority and applying a queuing mechanism that recognizes these priorities. On Cisco routers, the preferred method for doing this is using IP Precedence and WFQ. Since Roche has almost exclusively Cisco routers this seems to be the obvious way to go.

For determining what flows/packets belong to a certain application on the network we need to work out what characterizes these flows. At a minimum we could look for the application servers IP addresses in the Source or Destination address field in the IP header of the packets. If we want more granularity we could also look in the Source or Destination port field in the IP header for application specific ports. This is useful if more than one application is active on a server. It should be noted, though, that not all applications use specific port numbers. Some applications simply use a range of ports that are free, which are negotiated in the first connection. Also, the Cisco IOS 11.2 release does not support differentiating packets on a port number basis when setting the priority level.

Installation consists in gathering data about application servers' IP addresses and configure the key routers for marking packets with IP precedence and for enabling WFQ. See further down in this section for commands used for doing this.

Advantages with this model are that it is fairly simple to implement and does not require additional hardware or software. Benefits are that business-critical applications should get priority over best-effort traffic.

Disadvantages are that we have to keep a record of all server addresses and update the router configuration if they change. With a good database this could be done automatically by scripts if the security issue about who can log in to a router can be solved.

On Cisco routers, there are three steps for setting the IP precedence bits for a certain traffic flow. First we classify the traffic by making an access list with the parameters source/destination address/port as described above. Incoming packets are matched against this access list and their precedence bits are set. This function is called Policy-based Routing (PBR). Finally we apply this strategy to the specific interfaces we want on the router. In the Cisco documentation [12], we find the following commands:

The commands for making access lists for matching packets are:

ip access-list extended *name*
permit host *source* **any**
permit any host *destination*

Parameters:

- *name* name of the access list
- *source* the IP address of the application server
- *destination* the IP address of the application server

This creates an access control list (ACL) that we will use to match the flows belonging to the application. The first row creates the list. The first permit statement matches all traffic coming from the application server. The second permit statement matches all traffic going to the application server. The two permit statements must be repeated for every application server and for each application. The commands for setting the IP Precedence bit for a flow conforming to an ACL are:

route-map *map-tag* **permit** *[sequence number]*
match ip address *name [...name]*
set ip precedence *value*

Parameters:

- *map-tag* name of the policy map
- *name* the name of the previously defined access list
- *value* the IP precedence level to set.

We will assign this route-map to an interface on the router for enabling IP Precedence routing. The command for assigning a route-map to a specific interface is, when in interface configuration mode:

ip policy route-map *map-tag*

Parameters:

- *map-tag* name of the policy map

This must be done for each interface that we configure on the router. In addition, WFQ must be enabled on the interface. To enable it when in interface configuration mode, the command is:

fair-queue *[congestive-discard-threshold [dynamic-queues [reservable-queues]]]*

The parameters within square brackets are optional and are used to fine-tune the WFQ mechanism.

4.3. Tests

It was decided to test this model both in a network laboratory (NetLab) with real hardware and in a network simulator, which was part of this thesis project. See chapter *5 Simulation of QoS Benefits* for the simulation.

The goal of the tests was to verify the functionality of the WFQ and IP Precedence on the Cisco routers and to estimate the effect of introducing QoS in the RCN. A test network topology was set up, as shown in Figure 4-1.

It was decided to use FTP and Ping for the functional tests, and to use the application SAP R/3 for testing the benefit for a real application when introducing QoS. There are at least two good reasons for choosing SAP R/3 as a test application. First, it is a productive application where we can see the benefits directly. Second, in SAP there is a possibility to record macros, which will simplify the test process. We can record a typical SAP transaction and save as a macro. The next time we want to run it, we do not need to go through the complete process again, but merely select the macro to run. It can either be run in the background or in the foreground, where each step in the macro is confirmed by a keypress.

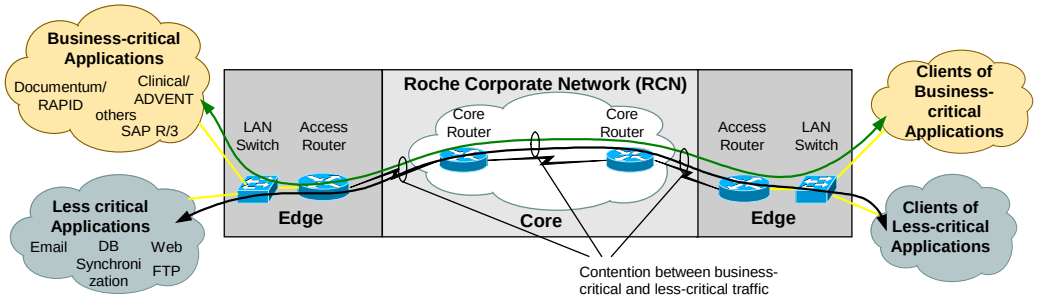

Figure 4-1 : NetLab test setup

4.4. Test results

The tests in the NetLab [2] and the tests with the network simulator show that the business-critical applications would benefit from a scenario with WFQ and IP Precedence. Test results show that it is important to characterize the traffic pattern of the application intended for prioritization, as the benefit from this QoS model varies with parameters such as packet size and throughput. Chapter 5 discusses this issue further. It was decided to implement this variant of differentiated services in the RCN.

The NetLab tests also discovered that we increase the load on the CPU on the routers largely when using Cisco IOS 11.2 and classifying packets. Using Cisco IOS 11.3 shows a much more moderate increase in CPU load, why an upgrade is recommended before deploying these services on the routers that should do the classifying of packets.

5. Simulation of QoS Benefits

The purpose of this simulation is to verify the impact of introducing QoS in the form of IP Precedence and WFQ on Cisco routers in the corporate network. The theoretical result coming out of this test is checked against the tests carried out on hardware in the NetLab.

5.1. Simulation model

For the test setup, five parts need to be included in the model:

- The **network topology** includes the network host stations, the routers, and the links between these entities. The configurable parameters of the links are duplex, bandwidth, and delay.

- The **queuing mechanism** will be configured to work on the router nodes in the network. The mechanism we will use is WFQ. The queuing mechanism controls the output on the link.

- The **traffic flows** are established between host machines in the network. Traffic flows will be used for generating background traffic and to simulate FTP and SAP traffic on the network.

- The **traffic classifier** will be used to work on the edge routers of the network. The purpose of this classifier is to set the IP Precedence bits on the packets belonging to the flows to be prioritized.

- The **measurement mechanism** will be used for measuring flow start and end times. We need this to be able to compare flow statistics with and without QoS applied.

These five parts should be considered when choosing the Network Simulator to work with.

5.2. Choice of simulator

Due to the relatively short time for the simulation, I did not have time to make a thorough evaluation of different simulators, so I chose to rely on recommendations of others. Also, since my budget for this was zero, it limited the possible choices. At EPFL we had an old version of a commercial simulator called OpNet, but it was available neither at Adventis, nor at Roche. A new license for this simulator costs around $100’000. The simulator, which was recommended by EPFL staff, was the UCB/LBNL/VINT Network Simulator - NS v.2 (version 2) [37], hereafter referred to as ns.

This simulator is a discrete event simulator targeted at networking research. It was developed at the University of California, Berkeley. The first implementations of ns were developed primarily on Unix (SunOS, FreeBSD, and Linux), but ns was also ported to Windows-95 and NT. It is free of charge and can be downloaded from <http://www-mash.cs.berkeley.edu/ns/>.

The simulator is written in C++. As a command and configuration interface it uses OTcl, which is an object-oriented version of Tcl. My study of the ns documentation [18] showed that it should be possible to implement the above elements and to conduct the tests we have set up.

5.3. Implementation: the ns simulator

In ns, the network topology is represented by nodes and links. The links connect the nodes with each other. The nodes represent both the routers and the host stations. The links can be configured for a certain delay and bandwidth. One-way links or full duplex links can be used.

The queuing mechanisms in ns are configured per link. This gives the same functionality as if they were implemented on an interface port of a router. The queuing schedule is applied to the packets before they are sent to the receiver side of the link.

The traffic flows are set up by agents that sit on the nodes in the ns topology, and applications connected to these agents. The agents represent the transport layer. There are implementations of TCP and UDP, which is enough for our tests. Applications are representing the upper layer protocols. There are implementations of a File Transfer Protocol (FTP) and Constant Bit Rate traffic (CBR) as applications. I have configured a Client/Server application model for the SAP traffic in OTcl. Each agent, and therefore traffic flow, can be assigned a FlowId. The queuing mechanism, for instance, uses this FlowId when setting up the queues.

I did not find an implementation of a Packet Classifier in ns. Instead, ns offers the possibility to assign a priority to the node agents. This way the traffic flow created by the agent will have a certain priority already from the beginning. This change of idea does not matter for our simulations, since it was not the Packet Classifier itself that was to be tested. The important thing is that we can set the priority of certain flows.

When run, ns produces a log file. In the log file, we find information about each packet in the simulation, and what happens to it. The packet can be sent, received or dropped, for example. Timestamps are associated with these entries. In addition, ns provides a discrete clock mechanism that can be used within applications to signal a start or stop time for example. This gives us the possibility to measure the traffic flows, as we wanted.

5.4. Test model setup

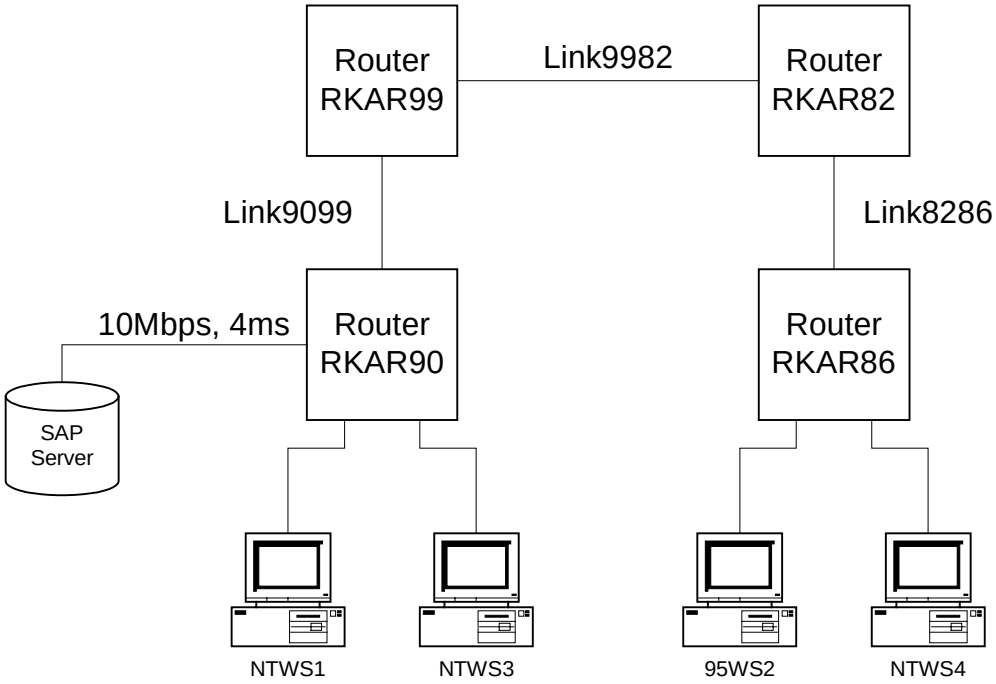

Figure 5-1: Topology setup in ns

Figure 5-1 shows the test topology that I configured in the simulator. Background traffic is run between the stations NTWS3 and NTWS4. FTP traffic is run between stations 95WS2 and NTWS1. SAP traffic is run between stations 95WS2 and SAP Server.

Background traffic is created by 15 concurrent UDP flows. The packet size of the UDP datagrams is 1310 bytes. Each flow sends a packet every 0.02 seconds. This creates a total load of 7.5 Mbps in each direction in the test network.

Two types of applications will be tested:

- FTP File Transfer of a 100 kbytes file
- SAP macro zb160172. This macro creates a new test user in SAP and deletes the same user again. It consists of 7 screens. The macro starts by sending one packet from the client to the server. The server returns one packet to the client and this continues for 7 rounds. The packet sizes vary between 40 and 931 bytes.

For each test, three cases will be run:

- Baseline: The application will run without any background traffic to disturb it.
- Background: The application will run together with the background traffic.
- Priority: The traffic from the application will be prioritized and run together with the background traffic.

The tests are conducted in four different configurations, as shown in Table 5-1.

	Link 9099	Link 9982	Link 8286
Configuration 1	**64 kbps, 1 ms**	**2Mbps, 1 ms**	**64 kbps, 1 ms**
Configuration 2	**64 kbps, 1 ms**	**2Mbps, 50 ms**	**64 kbps, 1 ms**
Configuration 3	**10 Mbps, 1ms**	**2Mbps, 1ms**	**10 Mbps, 1ms**
Configuration 4	**10 Mbps, 1ms**	**2Mbps, 50ms**	**10 Mbps, 1ms**

Table 5-1: The four configurations in the test setup

5.5. Test results

5.5.1. FTP and SAP test

File Transfer 100k	Baseline (seconds)	Background (seconds)	Increase of RTT
Configuration 1	**13.9**	**231.9**	**1568%**
Configuration 2	**14.4**	**229.6**	**1494%**
Configuration 3	**0.49**	**9.64**	**1867%**
Configuration 4	**1.11**	**10.25**	**823%**

Table 5-2: Round Trip Time for FTP File Transfer of a 100 kbytes file.

SAP macro zb160172	Baseline (seconds)	Background (seconds)	Increase of RTT
Configuration 1	**0.98**	**5.29**	**440%**
Configuration 2	**1.76**	**5.65**	**221%**
Configuration 3	**0.18**	**0.24**	**33%**
Configuration 4	**0.96**	**1.04**	**8%**

Table 5-3: Round Trip Time for SAP macro zb160172

An FTP File transfer takes 9-20 times longer when encountering background traffic depending on the configuration. The mathematical result should be 16, since the bandwidth should be shared equally between the 16 flows. The deviation from this result could be explained by:

- TCP setup time. The FTP flow is not running at maximum speed until the maximum TCP window size is fixed.

- For configuration 4, the link delay is more important when running the baseline test than when running the background test. The background traffic introduces more delay to the FTP flow than the link itself, and the result converges with that of the configuration 3 test.

The SAP macro takes 3-5 times longer when encountering background traffic for the configurations 1 and 2. For the configurations 3 and 4 we can see almost no difference. This time, the deviation from the mathematical result can be explained by two facts:

- SAP traffic gets more bandwidth than it needs. This is why we can see almost no difference in the configurations 3 and 4. The SAP traffic gets a share of the bandwidth equal to 2Mbps/16 = 128kbps, which is actually more than the 64kbps it receives in the configurations 1 and 2 without background traffic. We note that it is consequently faster in configurations 3 and 4 with background traffic.

- The SAP traffic benefits from the WFQ mechanism because it is an interactive flow. When its packets arrive in the queue they get scheduled quite soon, since the last SAP packet was sent quite some time before.

5.5.2. FTP and SAP test with new parameters

The following tests were carried out after the NetLab tests with the new parameters as defined during the tests in the NetLab. I changed the names of the configurations so that they correspond to the names in the NetLab report. The new configurations are shown in Table 5-4.

	Link 9099	Link 9982	Link 8286	Number of UDP flows
Configuration A	**64 kbps, 1 ms**	**2Mbps, 50 ms**	**64 kbps, 1 ms**	**5**
Configuration B1	**10 Mbps, 1ms**	**2Mbps, 1ms**	**10 Mbps, 1ms**	**15**
Configuration B2	**10 Mbps, 1ms**	**2Mbps, 1ms**	**10 Mbps, 1ms**	**15**
Configuration B3	**10 Mbps, 1ms**	**2Mbps, 1ms**	**10 Mbps, 1ms**	**30**

Table 5-4: Configurations corresponding to new NetLab definitions

The SAP macro was run in foreground mode in the NetLab, so I added a think time of 1s per packet at the client side. This should correspond to the time it takes to change the window on the client and for the user to press enter. The results are shown in Table 5-5 and Table 5-6.

FTP 100 kbytes	Baseline			Background		
	RTT (s)	Speed (kbps)	% of BW	RTT (s)	Speed (kbps)	% of BW
Topology A, NS	**14.4**	**55.6**	**87**	**86.9**	**9.21**	**14**
Topology A, NetLab	**N/A**	**60.2**	**94**	**N/A**	**4.50**	**7**
Topology B1, NS	**0.49**	**1633**	**80**	**6.76**	**118**	**5.8**
Topology B1, NetLab	**N/A**	**1862**	**91**	**N/A**	**19.8**	**1**

Table 5-5: RTT and BW usage for FTP File Transfer of a 100 kbytes file.

SAP test zb160172	Baseline (s)		Background (s)		RTT increase	
	NS	NetLab	NS	NetLab	NS	NetLab
Topology A	**7.8**	**9.63**	**9.8**	**23.18**	**26%**	**141%**
Topology B2	**6.17**	**5.57**	**6.24**	**10.20**	**1%**	**83%**
Topology B3	**6.17**	**N/A**	**6.29**	**N/A**	**2%**	**N/A**

Table 5-6: Round Trip Time for SAP macro zb160172.

By comparing with the NetLab results we can see that:

- The network simulation of the file transfer is closer to the theoretical result than the NetLab file transfer. However, they both show a significant increase in response time when background traffic is competing with the file transfer, so the main behavior of the WFQ function is verified.

- SAP traffic seems to be more affected by background traffic in the NetLab tests than in the network simulation. However, neither in the NetLab tests do we see an increase of the RTT of 500% and more, as we see for the file transfer. Hence we have verified that the SAP traffic already benefits from the WFQ algorithm without using priorities.

5.5.3. Interactive flow test

When conducting the above tests, and during the meetings with Roche's QoS project group, I realized that it could be interesting to see if an interactive flow benefits from WFQ mostly because it is interactive, or mostly because its packets are small. I made one more traffic flow setup, where I simulated an interactive traffic pattern with small or large packets. I used access links of 512 kbps and set the "think time" at both sides of the flow to 1 second, i.e., when receiving a packet, the station waits 1 second to send the response. This way, when a packet arrives at the queue, it is not affected by the finish time by the most recent packet in the flow. It gets a finishing time based only on its packet size. The background traffic I used was 15 concurrent UDP streams with packet size 635 bytes, thus in between the sizes of the two measured flows. So, the flow with small packets will compete against larger packet flows, while the flow with large packets will compete against smaller packet flows.

Interactive flow type	Baseline	Background	Increase of response time
Packet size = 120 byte	**13.20s**	**13.43s**	**1.7%**
Packet size = 1200 byte	**13.76s**	**17.25s**	**25%**

Table 5-7: Round Trip Time for interactive traffic.

We can see that neither of the flows are affected very much by the background traffic, but the flow with larger packets is affected more. We have thus verified that WFQ favors interactive traffic, and especially if its packets are small.

5.6. Conclusions

The main functionality of the WFQ mechanism was verified. The NetLab tests and ns show similar behavior when encountering background traffic, although some differences exist. We found that:

- WFQ favors interactive flows with small packet size, and shares the bandwidth equally between larger flows of type file transfer.

- Both the file transfer and the SAP macro are affected more by background traffic in the NetLab than in the network simulator. This is especially true in the topologies with a large core trunk of 2Mbit bandwidth, where the file transfer takes 5 times as long in the NetLab than in the simulator.

The impact of prioritizing the traffic flows could not be verified, because the WFQ implementation in ns did not take priority into account. However, other experience was gained when conducting these tests:

- SAP traffic already benefits from the WFQ mechanism, without using priority, due to the fact that it is an interactive flow with quite small packets. In the topology with a 2Mbit core trunk, many flows are needed to disturb the SAP traffic. On the 64 kbit access link, it is more important to prioritize the SAP traffic.

- A large traffic flow such as an FTP File Transfer suffers from background traffic. As we could expect from a Fair Queuing mechanism, it has to share the bandwidth with the background traffic, making it decrease its transfer speed as a function of the number of active flows on the link.

The findings of these simulations gives certain suggestions concerning traffic load and prioritization:

- When determining what traffic could benefit from prioritization, it is very important to examine the traffic pattern of the application. An interactive flow already benefits from the WFQ mechanism, and it would thus not be so important to prioritize it as it would be for an application with a larger traffic flow, (e.g., database replication).

- When configuring test scenarios, it is important to find a good background traffic pattern to get the right impact. We see for example that the background traffic has almost no impact on the

SAP traffic in the topologies with a 2 Mbit core trunk. We would probably need many more flows. Since WFQ works on a per-flow basis, it is important to know how many concurrent flows there are on a typical link on the live network to device an accurate model.

Network simulation is an interesting field and can be used to verify changes to network structure before actually committing them. The network simulator used for these tests unfortunately did not implement the Cisco WFQ mechanism. There is a commercial network simulator from MIL 3 (http://www.mil3.com/) named OpNet that is supposed to have this implementation in the version 6.0, which is soon to be released. The OpNet simulator is also closely related to HP OpenView, and can extract network topology from it. This could be an interesting tool for the Roche Corporate Network.

5.7. Evaluation of ns

My evaluation of ns is that it is a complex simulator, which requires a lot of time to understand properly. Not only does it require the understanding of network simulation in terms of setup, but also knowledge of Tcl/OTcl to configure it. If you need to change the source code and recompile ns, a good knowledge of C++ and a thorough understanding of the programming environment is required. A good start for learning ns is a tutorial written by Marc Greis [22].

Once a good understanding is gained, ns is flexible in that you can always implement a desired mechanism by introducing a new piece of code. This is why I think ns is a good simulator for testing new algorithms such as a new routing protocol, for example. For modeling larger networks with many traffic flows and a complex topology in a production environment, I would not suggest ns. Building large topologies in ns is a time-consuming task, although there are some topology generators [38, 39] made to simplify this task. There are commercial simulators more suitable for this, but then they also cost quite a bit more than ns, which is free of charge.

The environment used both at Adventis and at Roche is Windows NT. The compilation of ns failed when I used Windows NT and Microsoft Visual C++ v. 6.0. I applied the patches for Windows users, provided on the ns homepage, and I read the ns news group (ns-users@mash.cs.berkeley.edu). I found that other people also had problems with compiling ns under Windows NT. The port to Windows NT is fairly recent and still maybe incomplete. During the month of November 1998, there were discussions on this matter in the above-mentioned newsgroup. I found no solution for how to compile ns in my Windows NT environment, so I downloaded an already compiled version of ns for Windows NT. I installed it in my environment, and this worked fine. The drawback was that I could not change any of the main ns functionality that was built in the C++ code. I could still, however, configure ns and implement the test model using Otcl. Another approach could have been to install another OS, such as Linux, and to compile ns there. I decided that this would be too time-consuming for this project, as I was not enough familiar with the installation process of Linux.

When I ran some initial tests I discovered that the WFQ mechanism did not work as I expected it to. It did not seem to use the priorities that I had set. Looking at the C++ source code for the WFQ implementation proved this. The WFQ implementation in ns does not take priority of flows into account. This is a major concern to us, since this is what was intended to be tested Without this functionality, we will not be able to predict the impact of introducing QoS. To implement a WFQ mechanism like the one on Cisco routers would require changing the C++ source code and recompiling ns. Due to the platform problems described above, I decided that this would be too time-consuming. However, the rest of the test setup was possible to implement in ns, and we gained a lot of experience by using this simulation model.

My recommendation is to use ns in some Unix environment for the time being. Most of the ns users seem to be running some kind of Unix OS and in the ns news group there are more responses to questions concerning Unix than Windows. It would be easier to get help if running under Unix.

6. Charging in IP Networks

6.1. Motivation

Network resources such as links and routers are not free. How should the users of these resources share the cost for them? There are two major movements on the Internet market today: those who believe that access to the Internet should always be paid for on a flat-rate basis, and those who believe in usage-based charging. One major motivation for flat-rate charges is that technology evolves fast, and prices on bandwidth are steadily decreasing. One assumption is that it will be so cheap to add additional bandwidth, so that there will be no need to introduce a usage-based model that in itself adds cost overhead to the network. On the other hand, when QoS and services that differentiate several types of traffic are introduced in the network, we introduce different service levels. To make sure that these service levels are used efficiently, we need to work out a model that states how traffic should be prioritized. One such model can be usage-based charging, introduced as a cost incentive for an efficient use of the network.

6.2. Charging models

This section introduces the two main charging models that exist to date: flat-rate charging and usage-based charging. These can be configured in many different ways, and we will study the most important configurations.

6.2.1. Flat-rate charging model

The most commonly used charging model today is the flat-rate model, whereby the user pays for a certain bandwidth whether he uses it or not. The lease of a link to an Internet Service Provider (ISP) is an example of this model. For instance, the user, or an organization, subscribes to a link of 256kbit/s by paying a monthly fee. This fee is fix and is not affected by the amount of bandwidth actually used. Of course, no more than 256 kbit/s of bandwidth can be used. Another type of contract is the lease of a Frame Relay [21] link. Here, the fee is dependent on the actual Frame Relay port speed and the Committed Information Rate (CIR) specified. CIR is the bandwidth that is reserved for the user, which should be available at any time. If there is no congestion in the network, the user is allowed to use more bandwidth, up to the port speed, without paying extra fees.

Larger enterprises can have their own private networks, which consist of leased lines between different sites of the company. There is a cost for these links, as well as for routers and other network equipment, and a manpower cost for maintenance of the network. Flat-rate charging in such an organization could mean to split all the costs for the corporate network down to its subdivision cost-centers, according to the speed of the access links.

The flat-rate charging model can address the distinction of traffic with different priorities on a basic level. We could for example base the subscription fee on what applications are used by the end system, and estimate the impact on the network imposed by a certain application, considering also its priority level. We cannot address a dynamic use of prioritized traffic, where the end system can assign a certain level of priority for selected data flows.

The flat-rate charging model is also called a subscription-based charging model, since the end user subscribes to a certain service.

The major advantages of the flat-rate charging model is that it is fairly easy to implement and not very costly. It is also easy to understand allotted costs, which can be important for example for budgeting.

The disadvantages are twofold. First, it can not be accurate, since no measurements are done. As a result, costs may be unfairly allotted to end users. Second, the distinction between different levels of service is only done at a very basic level and does not support the dynamic use of priority levels.

6.2.2. Usage-based charging model

As can be derived from its name, usage-based charging is concerned with allotting charges to an end user or an organization depending on how much they use the network. For the usage-based model there are three important things to define:

1. **What should be charged for, and how?** Thus, when someone is using the network, his traffic flow would traverse one or more links or routers. Should he pay a volume-based fee independent of how many links he traversed, or should he pay on a per-hop basis?

2. **What/Who is the accountable entity?** To what organizational granularity should we charge? Options include, the site, the cost center, the end user, the IP subnet, or the application.

3. **By what end-point attributes should we differentiate between flows?** What are the characteristics of a flow, apart from the accountable entities, that should be differentiated? Options include, the volume, the priority, the TOS byte, the protocol, the time of day, etc.

As we can see, the number of options to these questions is quite vast. We can build several different usage-based charging models by altering these options. What options are suitable for a specific network? This depends on the network and organizational topology and must be considered for every special case, especially for what concerns option one. For option number two, there are some standard configurations that are often used:

- In a large enterprise with many interconnected sites, the first level of granularity can be the site. Thus, the utilization of the corporate network will be shared between the different sites of the enterprise on a usage basis. The accountable entity is the site.

- Going further down in granularity we find an application-based charging model. In this configuration, we set the accountable entity to an application cost center. We measure how much network resources the application and its users consume.

- A finer granularity is the user-based charging model. We set the accountable entity to the user and we measure the resources he uses on the network.

- A variant of the above charging models is group-based charging. A group can consist of a number of users, applications or sites.

For what concerns the end-point attributes, the three most commonly referred attributes are the following:

- Maybe the most intuitive charging characteristic of a flow is the volume of data that is transferred. A user making a large file transfer over the network uses much more resources than someone sending a mail does. Volume can be measured in different units, such as bytes or packets, or even in time units.

- With the introduction of QoS to the network, some traffic is given higher priority, so it feels natural to set a higher price for high-priority traffic than for best-effort traffic. Therefore, priority is an important end-point attribute.

- Another important issue is the time of day. A network is often differently loaded at different times of the day, with peak hours generally mapped to business hours. We could imagine having different tariffs depending on the time of day that someone is using the network, to give an incentive to transfer bulk data only when the network is lightly loaded.

Separate from the configuration is the *pricing*. How much to charge for a certain traffic flow can be independent of the model used. We must just make sure that the actual costs are distributed over the network users. Pricing issues are outside the scope of this thesis.

The advantages of a usage-based charging model are that (i) it is accurate; (ii) it provides a metric to ensure fairness; (iii) it addresses different levels of priority by distinguishing traffic by end-point attributes; and (iv) it invites use of the network in a cost-efficient way.

The disadvantages are that a usage-based charging model adds cost to the network in terms of reporting, collection and post-process overhead. We address this issue further in section 6.3.

6.3. Specifications and architecture for usage-based charging

Maybe the cornerstone in a usage-based charging model is a mechanism to measure the use of the network resources. Even without defining what network usage is, we realize that we need some measurement function to base our bills on. Measurements could for example include the number of bytes or packets sent to the network by an end system. Furthermore, we need an architecture that describes where to install measurement mechanisms and how to manage the data generated by them.

6.3.1. ITU-T Recommendation

The International Telecommunications Union has defined the Recommendation X.742 for a generic measurement function [26]. This model introduces three concepts:

- The *usage metering process* – This process is responsible for the creation of usage metering records as a consequence of the occurrence of accountable events in systems. The usage metering process is also responsible for logging of the usage metering records. Several accountable events may result in a single usage metering record. In general, the use of a service that demands the use of several resources will give rise to several usage metering records.

- The *charging process* – This process is responsible for collecting the usage metering records which pertain to a particular service transaction in order to combine them into *service transaction records*. In addition, pricing information (according to a tariff-system) is added to the service transaction records. The charging process is also responsible for logging the service transaction records.

- The *billing process* – This process is responsible for collecting the service transaction records and selecting from these the ones which pertain to a particular service subscriber over a particular time-period and produce the bill from these.

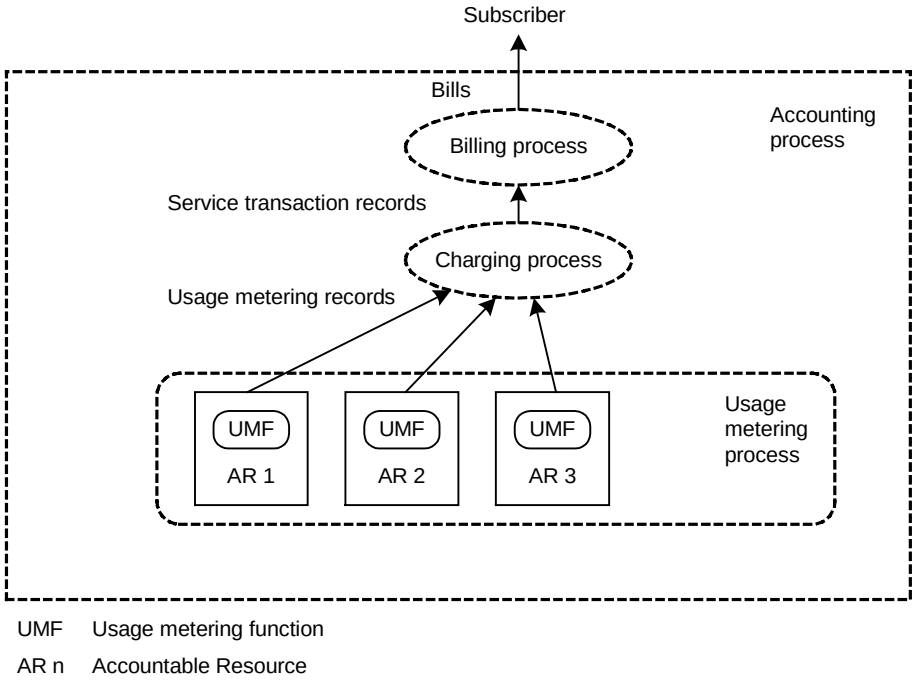

Figure 6-1: Example of modeling the accounting process

The Recommendation X.742 specifies only the usage metering process. The list of requirements for this process is vast, and we will only describe the general functionality of the metering process. It is modeled as several objects with specific functionality, which are connected to each other.

- The *accountable object* is the resource for which usage should be measured. Usage metering is always associated with an accountable object.

- A *usage metering data object* (or simply *data object*) is contained in an accountable object. The data object is responsible for gathering data about the use of a resource. It contains information that identifies the user, the service being provided, and a measure of the quantity used, together with other qualifying data. It is also responsible for logging this data as *usage metering records.*

- The *usage metering control object* (or simply *control object*) is responsible for controlling the usage metering of one or more accountable objects. It provides operations for controlling the collection of usage data from an accountable object, and for starting and stopping the collection. A control object manages a data object.

Before accounting can be started, at least one instance of the usage metering control object must be created. The control object will define *reporting triggers*, which specify the events that cause a data object to log a usage metering record. They include periodic events marking the passage of time for which a resource is in use and specified stimuli relating to other aspects of resource use. The control object will also identify the units of usage. Figure 6-2 shows the relationship between the different objects.

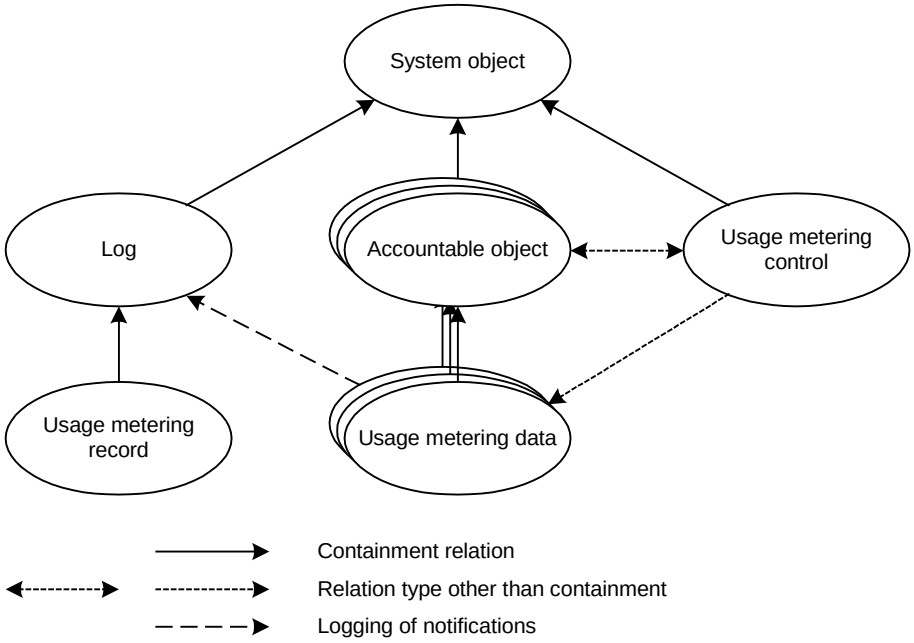

Figure 6-2: Example of structural relationship of managed objects

The specification in the Recommendation X.742 is of generic application and must be interpreted in the environment where it is supposed to be used. In our case, this will be an IP network, where the accountable objects are network resources.

6.3.2. IETF Working Groups

The Internet Accounting Working Group, which is now closed, published in 1992 the RFC 1272 [28], which proposes an architecture for accounting on the Internet. The idea of the proposed architecture is to use independent administrative domains with well-defined boundaries, as shown in Figure 6-3.

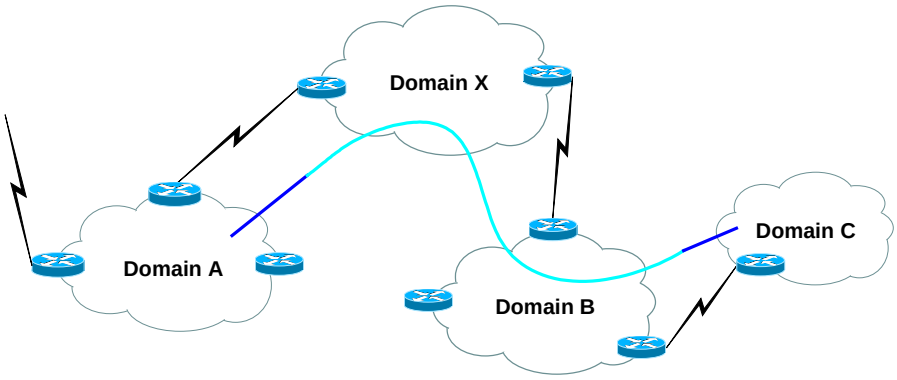

Figure 6-3: Example of independent domains

The administrator of a domain distinguishes between two types of traffic:

1. traffic with one or more end points within his boundaries (darker marked flow in the figure); and

2. traffic crossing his boundaries (lighter marked flow in the figure).

Within his boundaries, he will account for the end users, which can be individual users or departments within his domain. For traffic only crossing his boundaries, he is usually not interested in accounting for the end users, which are outside his administrative domain, but rather to the granularity of an adjacent domain. He is primarily interested in knowing between what adjacent domains the flow goes, so that he can charge the adjacent domains for their use of his resources. Likewise, the administrators of the adjacent domains will charge him for the resources he has used in their domains.

For instance, let us consider a user in domain A who connects to a server in domain C. Network resources will be used in all four domains in the above example. The domain X will forward traffic from A to B, and likewise, domain B will forward traffic from X to C. This is the lighter colored part of the flow shown in Figure 6-3. The administrator of domain C sends an aggregate bill to B with charges for the resources he has used on the domain C. It is now up to B to allocate these costs down to the appropriate granularity. It is important to note here that it is the responsibility of B to account for what entity used resources on domain C. The bill from C does not specify this information. If, as in our example, the actual user of the resources was an adjacent domain, then B will bill this adjacent domain, X, for the part of the bill from C that concerns the resources X has used. X thus receives an aggregate bill, which includes charges for resources used on both domain B and domain C, although he is only in direct contact with B. Likewise, X adds his charges and bills A. The administrator of domain A can now allocate the costs down to the appropriate granularity, e.g., the cost center of the user who submitted the traffic.

Since a bill received from an adjacent site does not specify what entity in the own domain used the resources billed for, we must make sure that we account for this information on the domain border. This requires that we know how much the adjacent domain will charge for transporting our traffic to the destination (e.g., X would not charge A the same amount for traffic going to C, as for traffic going to B). This is much like the telephony system, where we pay different tariffs depending on where we are calling. It is cheaper to make a local call than a long distance call. We need to set up contracts at the domain borders with negotiated tariffs with adjacent domains.

The result is that X has a table of costs to other domains on the network. When X receives a packet from A, he will look at the destination address of the IP header, map it to what destination it is going and apply the appropriate tariff. Every contract holds information of ranges of IP addresses. If this process should be successful, then it is important that IP addresses are well organized and not changed too often between domains, since this information will have to be updated in all contracts. If the network is organized in a hierarchical structure and hierarchical routing is implemented, it would greatly simplify the process of setting up the contracts between domains. IPv6 [17] solves this problem by providing IP addresses consisting of two parts: one that describes the location and another that uniquely defines the host.

An issue that is not considered by the RFC is whom we bill for a flow with a specific source and destination. Do we want to bill only the source or only the destination, or do we want to bill them half the amount each? There are incentives for all solutions. It might seem intuitive to charge the sender (the host corresponding to the source IP address in the packet header) of a flow, since he is responsible for sending out the traffic on the network. In the case of a client-server communication, though, it is the client requesting information to be sent to him, why he should be charged for that. On the other hand, this could be taken care of on the server side, where a measurement mechanism on the server make sure to account for the use of server resources. This is transparent to the billing model, but must be deployed the same way in the subset of the network communicating with the same type of contracts, since otherwise there will be mismatches.

The currently active IETF Working Group that deals with traffic measurement on the Internet is the Realtime Traffic Flow Measurement (RTFM) Working Group [25]. RTFM continues the work initiated by the Internet Accounting Working Group. The RTFM Working Group specifies four basic entities for accounting in RFC 2063 [8]. Figure 6-4 shows the relationships between these entities:

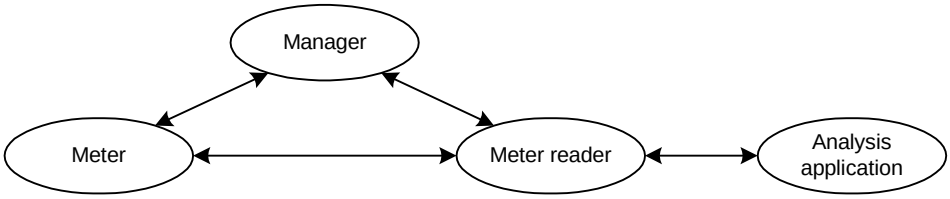

Figure 6-4 : Relationships between accounting entities

- the *meter* performs measurements and aggregates the results of those measurements;
- the *meter reader* collects data from one or more meters; it is responsible for the integrity and security of meter data on short-term storage and in transit;
- the *manager* configures meters and controls meter readers; and
- the *analysis application* processes, formats and stores meter data.

In the following sections, we will describe the first three of these entities. The functionality of the analysis application depends on the context where it is used. In our billing scenario, it can, for example, be used for translating the records of usage data into billing records in a billing application. This corresponds to the charging process in the ISO environment described in section 6.3.1.

6.3.3. Meter

The function of the meter is to examine a stream of packets going through its location. The meter can be located on a link, on a router, or on a LAN. The packets seen by the meter are classified into certain groups. A group can correspond to a combination of an accountable entity and end-point attributes, e.g., all traffic going to and from a group of hosts (department or domain) with premium priority. The classification is done by executing a series of rules belonging to an active rule set on the meter. Rule sets are downloaded to the meter by the manager and enforce the charging policy defined for the network. This rule set works with data such as packet source and destination, and other information it can find about the packet. These data are called *flow attribute values*. The meter keeps a flow table with *traffic flow records*. A traffic flow record is created for each flow defined by the rule set used. It contains information about the accountable entity and the end-point attributes defined, as key fields in the flow table. In addition to this, there are also kept value fields for each flow record. This information is normally represented by counters that are increased when a packet arrives, but it can also be for example time-stamps. The counters can represent, for example, the number of packets or the number of bytes forwarded in this flow.

An important factor when defining the rules for classifying packets into groups is the trade-off between overhead (cost of accounting) and detail. The granularity of the accountable entity defines how deep we want to go with the charging. Do we want to charge a specific user or host, or would it be more appropriate to charge a group of hosts, such as a department? The granularity of the endpoint attributes defines how we want to differentiate flows with the same accountable entity

from each other, such as maintaining state information about the higher layer protocols. The finer granularity we want, the larger the overhead. Each entity/attribute combination generates a field in the flow table. These data need to be stored in the meter, which requires meter memory, and when they are sent over the network, they are using network resources.

RTFM proposes a meter with a current rule set and a standby rule set. The standby rule set is coarser in granularity, and the meter will switch from the current rule set to the standby in case of lack of memory. A meter can run different rule sets on the same time, creating different tasks. Each task has a current and a standby rule set. The data produced by each task can be collected by one or more meter readers.

The meter can be implemented in several different ways. Here are examples of four types of technologies:

- Network monitors: These measure only traffic within a single network. It can be a small host station connected to the LAN and running a traffic meter program.
- Line monitors: These count packets flowing across a circuit. They would be placed on inter-router trunks and on router ports.
- Router-integral meters: These are meters located within a router, implemented in software. They count packets flowing through the router.
- Router spiders: This is a set of line monitors that surround a router, measure traffic on all of its ports and coordinate the results.

The meter location is another cost-critical decision. To be able to measure inter-domain traffic, we need meters at the domain boundaries. To be able to measure all intra-domain traffic, we need meters on the stub networks. Thus, to measure all traffic within the domain, we need meters on almost every router in the domain. This is probably not very cost-efficient. Instead, we should define the number of sub-domains we want to divide our network in, so that we can measure only important traffic.

6.3.4. Meter reader

The meter reader is the entity responsible for collecting data from the meters in the domain and for structuring the information it receives. The meter data that should be read can be either a static set of variables whose values are incremented, or a stream of records that must be periodically transferred and removed from the meter's memory.

It is extremely important to assure a reliable delivery of the meter data to the meter reader. This can be achieved by having:

- an acknowledgement retransmission scheme
- a redundant reporting to multiple meter readers
- backup storage located at the meter

The collection can be done in two ways: either the meter reader polls the meters for information, or the meters push the data to the meter reader using, for example, SNMP traps. There are advantages and drawbacks with both. When using polling, the meter reader has control over when data is sent,

and can possibly coordinate this with other actions. The main disadvantage is that polling messages add overhead to the network and meter traps are required all the same, as a function for the meter to send data if its buffers are getting full. Only using meter traps means that the meter reader has no control over when it receives information.

A major point of consideration is the memory/bandwidth trade-off. We have two options. First, we can use much memory at the meter and couple this with an efficient bulk-transfer protocol. Alternatively, we can use a minimal amount of memory at the meter, but this requires a more bandwidth consuming collection method, since data must be collected more often.

6.3.5. Manager

The manager is the entity used for managing the meters. It communicates with meters and meter readers via the network. It is a user front-end application for implementing the charging policy of the network. A meter reader can only be controlled by one single manager, while the manager can control several meters or meter readers.

The manager communicates with the meter to provide the following functions:

- Download rule set: The manager communicates the rule sets that enforce the charging policy to the meter.
- Specify meter task: The manager specifies what rule sets should be current and standby for each task.
- Set high water mark: Set a percentage level of the flow table capacity that defines when the meter should switch to its standby rule set to conserve the meter's flow memory.
- Set flow termination parameters: If the meter still runs out of memory, the flow termination parameters specify what flow records to purge first. These may include already reported flows, oldest flows or flows with the smallest number of observed packets.
- Set inactivity timeout: This is a time in seconds since the last packet was seen for a flow. Flow records may be reclaimed if they have been idle for at least this amount of time, and have been collected in accordance with the current collection criteria. This frees up meter memory.

The meter can, in turn, notify the manager about lack of memory or high usage, so that the manager can take precautions.

The manager communicates with the meter reader to specify what readers the meter reader should collect data from. In the case that the meter reader is polling the meter for information, it also specifies the collection interval and what data should be collected. The manager can also explicitly order a meter reader to poll a meter, for example if the manager has received a notification by the meter stating that it is about to run out of memory. It is the manager that specifies how the meter reader should aggregate the data it receives from the meters.

6.4. Implementations

Together with the architecture for accounting, the RTFM Working Group has defined a meter MIB (Management Information Base in the SNMP management framework) [9]. A free software implementation of this model, with meter, meter reader and manager, is NeTraMet [7, 10]. NeTraMet is developed by Nevil Brownlee, who is also a member of the RTFM Working Group.

Commercial implementations include Cisco's NetFlow switching and NetFlow FlowCollector [16]. These products are not implemented using the RTFM meter MIB. NetFlow is a switching technique implemented in the Cisco routers that can also export measured data. The NetFlow FlowCollector is an application that receives and manages NetFlow exported data. Together with HP, Cisco has announced the "Internet Usage Platform and Billing Analysis Solution" [15]. There are also other applications built on top of the NetFlow FlowCollector, such as Belle Systems' Internet Management System [3].

7. Charging in Roche's Corporate Network

7.1. Motivation

The Roche Corporate Network (RCN) is structured around a backbone with main sites connected to it. Smaller sites are connected to the main sites, and could, in turn, have "child sites" connected to them. Each site has costs for Wide Area Network (WAN) implementations such as links and WAN routers. It is not at all sure that the site that implements or maintains a link or other WAN equipment is the only site using it, since the site might serve as an interconnecting entity between other sites. This cost must then be divided by those who benefit from it.

The organization of the company is not constrained by the network structure. Business units can be a division of a site, as well as span over site or country boarders. Each of these units uses the WAN differently. This is why it is important to fairly divide the WAN costs between them.

With the introduction of QoS in the RCN, some traffic will be given a higher priority than other traffic. It is also very important to be able to charge for this.

7.1.1. Today's model of charging

The model used for dividing the costs for the RCN today is a weighted subscription-based charging. Thus, each business unit pays a share of the cost based on the number of employees it has and a weighting factor that takes into account how much load they put on the network. This weighting factor is estimated by some network monitoring and general knowledge about network usage, but no real-time measurements are taken into account.

7.1.2. Customer requirements analysis

In 1998, a customer requirement analysis was performed by a consulting firm (At Rete[1]) for Roche. The customers are, in this case, the business units of the Roche organization. The results of the analysis were that the current cost division is perceived as unfair, and that there is a wish for a new charging model. The cornerstones of the expected new model are that it should be fair, accurate, simple, flexible, and efficient:

- Fair in the sense that the costs should be fairly divided by the actual users of the WAN.
- Accurate in the sense that the charging should be based on accurate data. Thus, the usage of resources in the RCN should be measured before they can be charged for.
- Simple in the sense that it should be easy to understand the algorithm and to calculate the share of the cost for each cost center. This is very important for budgeting.
- Flexible in the sense that it should be easy to recalculate the charging algorithm when the structure of the network or the organization change.
- Efficient in the sense that the cost of implementing the charging structure should not be so large that it outweighs the benefits.

[1] at rete corporate networking ltd, Zurich, Switzerland

7.2. IP address resolution issues on Roche's corporate network

For user-based charging, the main issue is: How to recognize end users? From an IP packet in a flow, we can get the source and destination IP addresses and the source and destination ports. From this information we would like to resolve who is the owner of the flow in reality. This could be a user or an application server, for example.

In Roche's network, as described in section 2.4, an IP addresses is registered to the MAC address of the network device, and therefore to end user and cost center. This data is stored in a Network Management database, and it is accurate in 80% of the cases, since not all addresses have been registered with the Network Management Tool application. In the cases where DHCP is used, there are scripts running every six hours to update the Network Management database with the IP addresses mapped to MAC addresses. The default lease time for an IP address is 7 days, which means that if a machine is used more often than once a week, it will retain its IP address from time to time. We should however be aware that this information might be inaccurate for a small fraction of the end-stations during intervals shorter than 6 hours. Server addresses are static, and a database is being built up to add additional information about application servers, such as source and destination TCP or UDP ports of the application.

If we manage to resolve the IP address to the machine ID, then we still might have a problem if different users share the same machine. If these users belong to the same cost center, this issue is probably not so important, but if this is not the case, then we have to find a way to split the cost for this machine between its users.

In case the IP address belongs to an application server, we must conclude whether only one application is active on the machine, or if we might have more than one. If we have more than one application on the same machine, we might try to resolve which one is active in thy flow by looking at the port numbers they use. This may work in some cases, but there are also applications that simply use a range of ports that are free, which are negotiated in the first connection. In these cases it is not possible to identify the application by the port numbers used. Again, if the applications belong to different cost centers, we will have to find a way to split the cost for this machine.

To resolve the problems related to the user IP addresses, we suggest one of the following:

- Update the database that ties IP address to users and make sure it is regularly maintained. The charging application needs to interact with this database.

- Install a user registration utility that interoperates with the NT logon server, for example the Cisco User Registration Tool (URT) [14]. The charging application can then interact with the URT database.

To resolve the problems related to the server IP addresses we suggest to install the two applications on different machines if we can not resolve the problem by simply splitting the cost equally over the applications.

Other implications concerning IP address resolution are related to traffic via a proxy server, and multicast traffic, where the real source or destination address is not visible. To resolve these addresses to the real source or destination address we need to interact with other units on the way, such as proxy servers or multicast routers. This type of traffic represents about 5-10% of the total WAN traffic during daytime, and normally less than 1% during night. We suggest to apply flat-rate

charges for this type of traffic for the moment, not to complicate the charging model with too many database interactions.

7.3. Evaluation of scenarios for implementing charging in the RCN

The following sections discuss different scenarios for implementing charging in the RCN. We start by confirming a distributed approach to the charging model in section 7.3.1. In sections 7.3.2 and 7.3.3, we look at different scenarios for charging. These include both flat-rate and usage-based charging models.

7.3.1. Distributed approach

The RCN is a large network interconnecting distant sites. The WAN links connect these sites together and it is on these links that we can measure WAN activity. The ends of a WAN link belong to two different sites, and therefore it makes sense to use a distributed model to describe the charging model for the RCN with the sites as independent charging domains.

For this distributed model, we can separate the charging of WAN costs in two steps:

1. divide overall cost of WAN between sites (intersite charging); and

2. let each site charge back its costs to the appropriate granularity (charging within the site)

The term charging within the site should not be confused with charging for traffic within the site. The costs allocated to a site by the intersite charging are distributed to business units within the site at the proper granularity. If these tasks are independent, we can imagine different scenarios for both cases and combine them for optimal efficiency. Also, the benefit of separating these two tasks is that each site can possibly use different methods for charging within the site depending on their needs.

7.3.2. Intersite charging

This is the first step of the distributed model as described in section 7.3.1. We have set the accountable entity to a site and we will now discuss the parameter 1 in section 6.2.2 – what should be charged for, and how?

7.3.2.1 Calculated fixed price intersite charging

The idea of this scenario is to distribute Roche's total costs for the WAN to sites, depending on the bandwidth of their access links. In the hierarchy of the RCN, sites belong to an area and areas belong to a region. We define links between sites as area links, regional links or backbone links. An **area link** is connecting two sites within an area. A **regional link** connects two sites within different areas, but within the same region. A **backbone link** connects two sites from different regions. We collect information about all the links from each site and classify the links according to the above criteria.

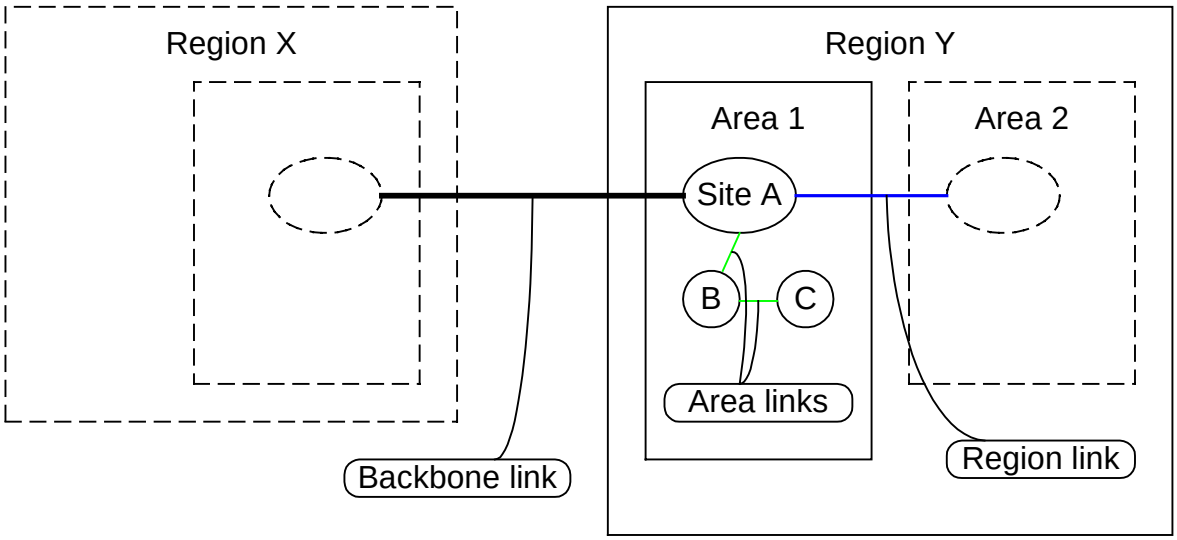

Figure 7-1: Link definitions

For each region, we calculate the total bandwidth of the backbone links connecting to the region. The sum of the bandwidth for backbone links for all regions make the network's total of backbone links. The share of the cost for the backbone that is allotted to a region is calculated as follows:

$$\text{Region's share of total cost} = \frac{\text{Region's total of backbone links}}{\text{Network total of backbone links}}$$

The same calculations are made for dividing regional costs to areas. The share of the cost that should be allotted to an area within the region is calculated as follows:

$$\text{Area's share of regional costs} = \frac{\text{Area's total of regional links}}{\text{Region's total of regional links}}$$

The last step, to get to the sites share of the WAN costs, is calculating on a sites area links. The share of the cost that should be allotted to a site within the area is calculated as follows:

$$\text{Site's share of areas costs} = \frac{\text{Site's total of area links}}{\text{Area's total of area links}}$$

This way we have allotted each site a fraction of the total WAN cost. If we take for example a site A, which belongs to the area B and to the region C. Region C uses 30% of the backbone links. Area B uses 50% of the regional links and site A uses 10% of the area links. We get the following equation: Regional cost

$$\text{Costs for site A} = 0.1 \times (\text{Local area cost} + 0.5 \times (\text{Regional cost} + 0.3 \times \text{Backbone cost}))$$

Advantages of this scenario are that it is simple and flexible. When adding a link, we need to update the table and then we get the new portions for each site. It is quite fair for the end sites since the bandwidth of their access links should correspond to the load they impose on the network. It is not so costly to implement since it does not require any measurement equipment at all. It should be quite efficient.

Disadvantages are that it is not accurate since we are not doing any measurements. It is unfair to a site that serves as an interconnecting site that forwards traffic from one site to another. This site will be charged too much, since the bandwidth of its links is proportional to its own load plus the load of the traffic that it is forwarding. There are no incentives for use of the network in a cost-efficient way.

7.3.2.2 Usage-based intersite charging on a per-hop basis

The idea with usage-based intersite charging is to allocate the costs of the RCN to the different sites depending on how much they are using the network. To be able to do this, we first have to define usage of the network.

To define usage of the network as resource utilization would mean to account for every traversed link and router of a flow. This is the model described in 6.3.2, with autonomous charging domains. In this scenario, we would need to measure traffic at every site border. Each site would then need to account for the traffic flows starting or ending on the site and for the flows traversing the site.

For each link, the site would exchange bills with the adjacent site on the other side. This could be done either explicitly between sites, or implicitly by sending the statistics to the Network Management group. The latter then distributes the aggregated bills back to the sites.

The setup consists in installing measurement equipment at each site border and defining the contracts between the sites. This includes the mapping of IP addresses to sites and the implementation of an automated process for updating. The flexibility of this scenario is dependent on how we could implement the algorithm for setting up the contracts and how we could automate the process of keeping a table of IP addresses.

The advantages of this scenario are that it is fair and accurate. The accounting for use of prioritized traffic is possible by differentiating flows by end-point attributes. We can also set different tariffs for the links depending on the time of day they are used. This scenario invites use of the network in a cost-efficient way.

The disadvantages are that it is not at all simple. It is costly to implement and therefore the efficiency must be carefully calculated.

7.3.2.3 Usage-based intersite charging on a per-access basis

Another way of defining the usage of the network is on a per-access basis: we apply the same tariff, no matter how far data is transferred. This model is a little less fair from a cost division point of view, since short distance users are billed as much as long distance users, who use more resources. Nevertheless, if we see the WAN as a communication medium that everyone should have access to for the same price per load, then this model makes sense.

In this scenario, we measure only traffic starting from or ending at our site. We measure the traffic we send and receive on the network and get charged for it by Network Management. The security aspect is very important here. The site itself can not be responsible for the measurement equipment, since this will be the basis for its bills.

The setup consists in installing measurement equipment at each site border and in solving the security issues related to who is responsible for the measurements.

The advantages of this scenario are that it is accurate, flexible, quite simple and quite fair. It enables the accounting for use of prioritized traffic by differentiating flows by end-point attributes. We can also set different tariffs for the links, depending on the time of day it is used.

The disadvantages are that it is costly to implement and therefore the efficiency must be well calculated.

7.3.3. Charging within the site

This is the second step of the distributed model, as described in section 7.3.1. We have defined the cost assigned to the site and we will now discuss how to distribute these costs over the units within the site.

7.3.3.1 User-based charging

The idea of user-based charging is to set the accountable entity to the end users and charge them for the bill the site has to pay for WAN usage. We might actually bill the cost center of the end user, but the administration of the cost center might be interested in knowing who used what resources. Thus it is interesting to account on a user level.

An important issue here is to understand how to resolve who is the end user of a flow. The IP address header serves us with the information about IP addresses and port numbers. This information must then be mapped to an accountable entity, which might be an end user or an application server. Section 7.2 describes issues related to IP address resolution.

In case the end user is an application server, it might be interesting to save information about what remote IP address was accessing the server, so that the cost center for the server can, in turn, bill the users of its resources. If the application server is not capable of resolving this itself, we might need to provide this information. This will increase the size of the database that we need for storage of user-based charging information.

At the site border, we need to measure the traffic entering or leaving the site. Depending on what intersite charging scheme we adopted, we might need to save different types of information. If we chose to deploy either the fixed price scenario or the usage-based scenario on a per-access basis, it would make sense not to take distance into account for the end users either. If we chose to deploy the usage-based scenario on a per-hop basis, then we should also charge the end users on a per-distance basis. The distance table is already set up by the intersite charging model.

The minimum information that we need to save is the volume and the priority per end user. If we do not consider the other end of the flow, this should be feasible without creating too large a database.

Some simple calculations will give us a rough idea of the maximum size of the database if we do not consider the remote IP addresses. The information we need to store at a minimum would be:

Local IP address	4 byte(s)	
Volume	8 byte(s)	
Priority	1 byte(s)	
Distance	1 byte(s)	(Optional field, depending on intersite charging model.)
Time	1 byte(s)	(Optional field, depending on intersite charging model.)
Total record size	15 byte(s)	

In the case of Basle/Kaiseraugst, we have 10000 local IP addresses and 2 levels of priority. We conduct one calculation imagining 10 possible distance tariffs and 2 different time tariffs, for high and low load, and another calculation, without taking distance or time into account. The maximum size of the database would be:

10000 x 2 x 10 x 2 x 15 bytes = 5.7 Mbytes. (Distance and time included)

10000 x 2 x 13 bytes = 254 kbytes. (No distance, nor time, included)

These databases are small, so they do not meet any diskspace or processing limitations for deployment. Yet, an issue might be, as described above, that we have to store the remote IP address also, at least for the case where the local end user is an application server.

For the calculation of the database size with remote IP addresses related to application servers, we need all the above information plus a remote IP address of 4 bytes, so the total record size would be 18 bytes. In Roche's network, we have about 42000 remote addresses outside Basle/Kaiseraugst, and we have about 20 application servers to account for in the Basle/Kaiseraugst site. This yields the following maximum size for the database:

42000 x 20 x 2 x 10 x 2 x 19 bytes = 609 Mbytes. (Distance and time included)

42000 x 20 x 2 x 17 bytes = 27.2 Mbytes. (No distance, nor time, included)

These figures are not unreasonably high. We should also note that this is the maximum size possible for the database without storing extra information, based on that we create a record for each possible combination of the above figures, which will not very likely be the case. Additional information, such as tariff and cost center, will also have to be stored with the record, but the order of magnitude should remain acceptable.

What we have gathered now is information about IP addresses. Our next step is to map the IP addresses to end users. As stated in section 7.2, not all IP addresses are possible to resolve to end user/cost center at the moment. For a successful deployment of this scenario, this issue must be resolved. In the first step, what need to be resolved are only local IP addresses. This way we can charge the local cost centers for their share of the billed WAN traffic. We can back up our figures by also showing what remote IP addresses used application-server resources within our site. The second step concerns resolving IP addresses for the entire RCN. This would make it possible to inform the cost centers of the application server about the share of resources used per remote end user.

The setup consists in installing measurement equipment at the site border, and in maintaining the database of server addresses. We must also solve the IP address mapping to end user by updating the database with, in the first step, all local IP addresses.

The advantages of this scheme are that it is fair and accurate. It enables the accounting for use of prioritized traffic by differentiating flows by end-point attributes.

The disadvantages are that it is not simple to calculate on beforehand, since it is completely usage-based. It requires resources to solve the IP-address-resolution issue. The flexibility of this scenario is dependent on the maintenance of the IP-address-resolution database.

The assessment of the efficiency of this scheme should take into account the feasibility of IP address resolution in practice.

7.3.3.2 Application-based charging

In the application-based charging scenario, we will set the cost center responsible for an application as the accountable entity. It is then up to the application management to distribute the cost to its users. This is interesting since some WAN applications use more bandwidth than others, and not all users use all WAN applications.

The idea is to identify key applications and measure their use of the WAN. This requires measurement points at site boundaries for traffic entering or leaving the site. The traffic to be measured are the flows with the source or destination addresses equal to an application server. This requires all application server addresses to be known in the entire RCN.

We should distinguish between two types of traffic flows belonging to an application: the application server is either local or remote. If the application server is local, it is interesting to report the remote addresses to the application cost center for it to be able to charge its users. On the other hand, if the application server is remote, it is important to collect information about the local addresses for the application cost center. With this information, the application cost center knows which of its users used resources on the remote server.

The calculation of the database size is achieved in two steps: first for local servers, then for remote servers. For local servers, the calculation is identical to the previous scenario (*User-based charging*):

42000 x 20 x 2 x 10 x 2 x 19 bytes = 609 Mbytes. (Distance and time included)

42000 x 20 x 2 x 17 bytes = 27.2 Mbytes. (No distance, nor time, included)

For the remote servers, assuming we have 100 servers to account for in the RCN, we get a maximum database size of:

100 x 10000 x 2 x 10 x 2 x 19 bytes = 725 Mbytes. (Distance and time included)

100 x 10000 x 2 x 17 bytes = 32.4 Mbytes. (No distance, nor time, included)

As for the previous scenario, these figures are not unreasonably high. It should be possible to create a database with the information that we need.

The setup consists in installing measurement equipment at the site border and maintaining the database of server addresses. The IP address resolution issue is also important here, but even without resolving this, we are still able to charge the application owner if we have the addresses of the servers.

It is important to verify how much of the WAN traffic is produced by these applications to give a measure of the efficiency of this scenario. If this proves to be the major part, then we could imagine a flat-rate cost division of the rest of the traffic, assuming that utilization of Internet/intranet or mail traffic is almost uniform per user.

The advantages of this scenario are that it is quite fair and it is quite flexible if we choose only to maintain the database of server addresses.

The disadvantages are that it is not so accurate per end user. It is not so simple to calculate on beforehand, since it is usage-based.

7.3.3.3 Subscription-based Charging

The idea of this scenario is that each business unit subscribes to the RCN with a certain service level. In other words, the business unit can decide to what priority it wants to subscribe different shares of its traffic.

The setup consists in setting up service-level agreements with the customers.

The advantages of this scenario are that it is very simple and flexible.

The disadvantages are that it is not usage-based, why it can be even more unfair than the current charging model with weighting factors. It is not accurate since we are not making any measurements. It is not efficient, since we do not improve the current charging model.

7.4. At Rete's proposal

In 1998, the consulting firm At Rete proposed a charging scenario for Roche's Corporate Network [1]. They chose the approach with fixed-price intersite charging, as described in section 7.3.2.1. For charging within the site, they suggested the application-based charging method, as described in section 7.3.3.2; they also proposed an optional flat-rate charge for nonaccountable traffic.

When we analyzed the customer requirements in section 7.1.2, the cornerstones of the expected new model were that it should be fair, accurate, simple, flexible, and efficient. These were the end users' expectations. Now, as far as the administrators are concerned, At Rete identified four objectives:

- *Fast implementation.* A complete product should be implementable in a short time frame.

- *Easy implementation.* This assumption is made to support the above statement. The design should be as simple as possible, so as to avoid possible sources of delay because of complicated implementation steps.

- *Incentive for cost-efficient use of the network.* The introduction of charging in Roche's network should make application owners or end users aware of WAN costs, so that their behavior converges toward a cost-efficient use of the network resources.

- *Low maintenance cost.* The cost of maintaining the charging application should be minimized.

When At Rete compared the administrators' objectives with the user requirements, they came to the conclusion that the best cost-benefit ratio would be obtained by resorting to fixed-price charging for cost-division between sites, and application-based charging for charging back these costs to the units within the site.

Concerning charging within the site, their conclusions were threefold:

- Application-based charging is the scenario that provides the best cost-benefit ratio.

- User-based charging is not appropriate for the time being, because it has very dynamic input data and a high amount of data need to be processed.

- Subscription-based charging is not recommended either, because it does not provide a higher perceived fairness than the currently implemented solution.

7.5. Our proposal

We believe that for intersite charging, a usage-based scenario would be too costly to implement in the short term, because it would require the installation of measurement mechanisms at every site border in the RCN. Instead, we propose to adopt usage-based charging within the site. If this strategy is deployed in a majority of Roche's sites, we will be able to evaluate the usage-based scenarios for intersite charging with the experience gained from charging within the site. We are thus following the same line as At Rete for intersite charging in the short term.

Within the site, the solution that we advocate is user-based charging. In addition, we believe that in the first step also apply a flat-rate charge to the end users, covering the traffic that we do not account for in this model, such as non IP traffic and Web traffic via a proxy.

Let us compare our proposal with At Rete's in the light of the customer requirements presented in section 7.1.2 (fairness, accuracy, simplicity, flexibility and efficiency) and the administrator objectives presented in section 7.4.

While At Rete emphasized on the administrators' requirements of fast and easy implementation, we are a little less stringent with these objectives. We believe that by decreasing the demands on time and ease of implementation, we can gain in terms of fairness and accuracy. We realize that it might be more complicated to deploy the user-based scenario than the application-based scenario, but we believe that the benefits of it will overcome this delay in the long term.

Like At Rete, we believe that subscription-based charging will not provide us with a more fair charging algorithm than the one already implemented today. Its benefits in terms of simplicity and flexibility are outweighed by its deficits in fairness and accuracy.

The major issue with user-based charging is the resolution of IP addresses. We believe that this issue can be solved in the startup phase, by updating the Network Management Tool database with the missing IP addresses. When we start to do measurements and map IP addresses to end users and cost centers, we will find a number of addresses that are not registered. These addresses must be carefully examined before continuing the project. If their proportion is high, then a major update of the database must be done, whereas if this fraction is small, we can continue to deploy the implementation and maintain the database while it is running. In the long term, we should consider installing a user registration utility that interoperates with the NT logon server, for example Cisco User Registration Tool [14]. If we are already working at a user-based level, switching to the user registration utility should be smooth. We only have to change what database to interact with for end-user identification.

The gain in terms of accuracy and fairness is that we will have a charging application with end-user granularity. This gives a strong incentive for a cost-efficient use of the network: each user is individually concerned. Since the end user can as well be an application server, there is an incentive for application owners to place servers so as to minimize network cost, and thus optimize the cost efficiency in the network.

Flexibility and the cost of maintenance are linked. The flexibility of user-based charging is dependent on how flexible it is to maintain the database of IP addresses for both end users and application servers within the site. For the application-based charging, it is a question of maintaining the database for application servers within the entire RCN. In both scenarios, if we want to record not only the IP addresses that have used resources of a server, but also the end user, then we need to maintain the IP address resolution database for the entire RCN. We recommend

starting with local addresses, and then, with the experience gained from this, we can extend the update to remote addresses. With the increasing use of DHCP, which has an automated registering process in the RCN, we believe that the maintenance of the IP address resolution should not be too complicated. Another view of the flexibility is the integration with network policing on a user basis. With charging on a user basis, we are taking the first step towards this integration.

For what concerns the simplicity of the user-based charging model, it is clear that it is not at all simple to calculate beforehand how much a certain cost center will be billed, as it is completely usage based. This is true for both models, although the figures for the application-based model will probably be a bit more stable, since application usage is more uniform. If the data is considered too fluctuating, it is possible to use the measurements as provisions for the next year's budget.

It should be noted that both At Rete's and our proposals well meet the customer requirements and the administrators' objectives. They only differ a bit on the aspects to be emphasized. Whether the focus should be put on having a simple and fast implementation soon, or on having a fair and accurate charging application, is a matter of corporate management policy.

8. Conclusion

8.1. Summary and contribution

This thesis was centered around two related projects at Roche: one concerning the introduction of QoS in the network, and another concerning the charging for the use of network resources.

In the QoS project, we have tested a model for differentiated services on Roche's corporate network. The model is based on IP Precedence and Weighted Fair Queuing, which are standard components in Cisco routers today. The tests were successful, and it has been decided to deploy this model in Roche's network. Apart from assisting in the testing of the hardware, carried out by Roche, I have provided an additional test environment using a network simulator.

In the charging project, this thesis provides an extensive study of how charging is deployed in IP networks today. It also explores different scenarios for charging in Roche's Corporate Network, and proposes a solution with possible further enhancements in the long term. The proposed scenario combines flat-rate charging and usage-based charging with a fine granularity (end user).

8.2. Benefits for the student

This project has been carried out in collaboration with several institutions. The author is enrolled at KTH, Stockholm, and he conducted this thesis as an exchange student at EPFL, Lausanne. Moreover, this thesis was an industrial project in collaboration with Adventis, a consulting company specialized in telecommunications, and their customer Roche, a large industrial corporation in the pharmaceutical business. During the startup of the project, we realized that the challenges of this project were not all technical, as the wishes of a university do not necessarily meet those of an industrial. Coping with and solving these occasional divergences proved to be a very valuable experience.

It was also a great experience to work in a large enterprise with a high-tech corporate network. This has given me a much deeper knowledge in the field of network management, and experience with modern networking products used in the industry today. These include Cisco hardware, Cisco software, and network management applications such as Cabletron Spectrum, HP Openview Network Node Manager, HP Openview Netmetrix, and NetScout.

It has also been very inspiring to work with a consulting company, where the environment is dynamic and offers many possibilities and challenges. Responsibility and initiative are key words in such enterprises.

Last but not least, this project gave me the opportunity to develop many technical skills, especially in IP networking and QoS, which are hot topics on the market today. I have also gained a good experience in the field of network simulation.

8.3. Future work

With respect to charging, our work in this thesis was limited to the analysis phase. The next steps would naturally be the high-level design, the detailed design and the implementation of our charging model. With the experience gained in the implementation phase and during initial

maintenance, we could also evaluate more complex models for charging in the corporate network, as described in chapter 7.

Another possible direction for future work is the introduction of policy management applications. There are more and more tools and applications on the market for network policy management. Policy management applications aim to enforce the network policy actively by measuring the conformance to service level agreements and controlling access to network resources as a function of the user, the time of the day, already consumed resources or other configurable criteria. Network devices such as routers or servers communicate with a policy database server that stores the network policy. Policy management applications can help automate many processes otherwise addressed by monitoring and manual follow-up. These applications could be very interesting for Roche.

Finally, based on the experience that we gained in network simulation, we think that traffic modeling can be a good tool for predicting the future performance of the network. The idea is to measure typical traffic patterns from applications used on the network, and feed a network simulator with this traffic multiplied by the number of subscribers to the applications and the number of transactions per time unit. The network simulator models the network and predicts the performance given the traffic load. From simulations, it should be possible to foresee possible bottlenecks in the network and act upon them proactively.

9. References

[1] At Rete. *Feasibility Study & Cost Estimates for End User Chargeback.* Powerpoint presentation. Roche internal project document. October 1998

[2] At Rete, Roche. *QOS project: "Limited CoS" tests.* Roche internal project document. March 1999

[3] Belle Systems. *Internet Management System.* Available at: <http://www.belle.dk/prodframeoct98.html>, February 1999.

[4] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang and W. Weiss. *RFC 2475. An Architecture for Differentiated Services.* IETF, December 1998.

[5] R. Braden, D. Clark and S. Shenker. *RFC 1633. Integrated Services in the Internet Architecture: an Overview.* IETF, July 1994.

[6] R. Braden, L. Zhang, S. Berson, S. Herzog and S. Jamin. *RFC 2205. Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification.* IETF, September 1997.

[7] N. Brownlee. *NeTraMet Release Notes.* Available at: <http://www.auckland.ac.nz/net/NeTraMet/>, January 1999.

[8] N. Brownlee, C. Mills, G. Ruth. *RFC 2063. Traffic Flow Measurement: Architecture.* IETF, January 1997.

[9] N. Brownlee. *RFC 2064. Traffic Flow Measurement: Meter MIB.* IETF, January 1997.

[10] N. Brownlee. *RFC 2123. Experiences with NeTraMet.* IETF, March 1997.

[11] Cisco Systems. *Advanced QoS Services for the Intelligent Internet.* Available at: <http://www.cisco.com/warp/public/732/net_enabled/qos_wp.htm>, January 1999.

[12] Cisco Systems. *Cisco IOS Release 11.2 Network Protocols Configuration Guide, Part 1.* September, 1998.

[13] Cisco Systems. *Cisco IOS(TM) Software Quality of Service Solutions.* Available at: <http://www.cisco.com/warp/public/732/net_enabled/qosio_wp.htm>, January 1999.

[14] Cisco Systems. *CiscoAssure User Registration Tool.* Available at: <http://www.cisco.com/warp/public/734/capn/caurt_ai.htm>, October 1998.

[15] Cisco Systems. *HP and Cisco Deliver Internet Usage Platform and Billing and Analysis Solutions.* Available at: <http://www.cisco.com/warp/public/146/april98/28.html>, February 1999.

[16] Cisco Systems. *NetFlow Services and Applications.* Available at: <http://www.cisco.com/warp/public/732/netflow/napps_wp.htm>, October 1998.

[17] S. Deering and R. Hinden. *RFC 2460. Internet Protocol, Version 6 (IPv6) Specification.* IETF, December 1998.

[18] K. Fall and K. Varadhan. *ns Notes and Documentation.* Available at: <http://www-mash.cs.berkeley.edu/ns/nsDoc.ps.gz>, December 1998.

[19] P. Ferguson and G. Huston. *Quality of Service in the Internet: Fact, Fiction, or Compromise.* In proc. INET'98. August, 1998.

[20] S. Floyd and V. Jacobson. "Random Early Detection Gateways for Congestion Avoidance". *IEEE/ACM Transactions on Networking,* Vol. 1, No. 4, August 1993.

[21] Frame Relay Forum. Available at: <http://www.frforum.com/index.html>, March 1999.

[22] M. Greis. *Tutorial for the UCB-LBNL-VINT Network Simulator ns.* Available at: <http://titan.cs.uni-bonn.de/~greis/ns/ns.html>, October 1998.

[23] IETF DiffServ Working Group. Available at: <http://www.ietf.org/html.charters/diffserv-charter.html>, March 1999.

[24] IETF IntServ Working Group. Available at: <http://www.ietf.org/html.charters/intserv-charter.html>, March 1999.

[25] IETF RTFM Working Group. Available at: <http://www.auckland.ac.nz/net/Internet/rtfm/TOP.html>, March 1999.

[26] ITU-T. *Recommendation X.742. Information Technology – Open Systems Interconnection – Systems Management: Usage Metering Function for Accounting Purposes.* ITU, Geneva, Switzerland, April 1995.

[27] M. Jander. "Clock Watchers". *Data Communications International,* September 21, 1998.

[28] C. Mills, D. Hirsh and G.R. Ruth. *RFC 1272. Internet Accounting: Background.* IETF, November 1991.

[29] K. Nichols, S. Blake, F. Baker and D. Black. *RFC 2474. Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers.* IETF, December 1998.

[30] A. Parekh. *A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks.* Technical Report LIDS-TR-2089, Laboratory for Information and Decision Systems, MIT, MA, USA, 1992.

[31] C. Partridge. *Gigabit Networking.* Addison Wesley, Reading, MA, USA, 1994.

[32] J. Postel. *RFC 768. User Datagram Protocol*. IETF, August 1980.

[33] J. Postel. *RFC 791. Internet Protocol.* IETF, September 1981.

[34] J. Postel. *RFC 793. Transmission Control Protocol.* IETF, September 1981.

[35] S. Shenker and J. Wroclawski. *RFC 2216. Network Element Service Specification Template.* IETF, September 1997.

[36] W. Stallings. *Data and Computer Communications.* 5th Edition. Prentice Hall, Upper Saddle River, NJ, USA, 1997.

[37] UCB/LBNL/VINT. *Network Simulator - ns version 2.* Available at: <http://www-mash.cs.berkeley.edu/ns/>, October 1998.

[38] UCB/LBNL/VINT. *Network Simulator: Scenario Generation.* Available at: <http://www-mash.cs.berkeley.edu/ns/ns-scengeneration.html>, October 1998.

[39] D. Zappala. *Topology Generation for Network Simulation.* Available at: <http://www.cs.uoregon.edu/~zappala/topology/>, October 1998.