Engineering a Quality of Service-Enabled Network: a Case Study

ENGINEERING A QUALITY OF SERVICE-ENABLED NETWORK: A CASE STUDY

Fulvio Risso
Dipartimento di Automatica e Informatica, Politecnico di Torino,
Corso Duca degli Abruzzi, 24, 10129 Torino - Italy

ABSTRACT

This paper presents a case study concerning a network operator that wants to provide services with a certain degree of quality to its customers. Users are presented with a set of services with different Quality of Service capabilities and the network is engineered in order to satisfy the requests and to maximize provider's revenues. This paper presents the service model adopted and the technical choices that allowed this network to work, taking in mind scalability issues. Finally, it presents the problems that arose when this network was put in place by using both commercial and experimental routers.

I. INTRODUCTION

Recently, the Internet has seen a growing interest in Quality of Service (QoS) issues. This is due to the increase of the traffic in the network and to the different services that are going to be provided through it. Particularly, commercial interests generated the need to differentiate the service on the application basis (for example bulk data transfers against real-time data) and on the customer basis (on-line traders are more sensitive to quality than home web-surfers).

Up to now, the Internet Engineering Task Force (IETF) proposed two models to provide QoS guarantees on the Internet. The Integrated Services [1] (IntServ) model allows for a per-flow reservation and it is able to provide "deterministic" guarantees with its Guaranteed Services [3] specification. That is, a user can book in advance the amount of bandwidth it needs on a specified path and it can determine the end-to-end delay. The network will check for resource availability and it will accept the booking only if the request can be satisfied. A second specification, the Controlled Load [2], defines a service that corresponds to the one experimented on an unloaded network. This model does not guarantee an end-to-end delay and it focuses only on bandwidth parameters.

The IntServ model is extremely interesting especially because it matches exactly what the customers want (i.e. the certainty that the resources will be available). However it does not scale and it is very expensive because of the overhead to manage several thousand of flows at the same time in core routers. Moreover, such model is based on a run-time booking procedure that requires an AAA (Authentication, Authorization, Accounting) system in order to generate appropriate bills.

The Differentiated Services [5] (DiffServ) model does not aims to provide absolute guarantees and it is therefore simpler and more scalable. The key idea behind DiffServ is to specify the way a router will forward traffic; end-to-end services are not taken into account, although they can be build by inserting some admission control mechanisms on top of a DiffServ network. The Expedited Forwarding [6] (EF) specification provides a way to carry out high quality services using an existing best-effort infrastructure. Basically, a small percentage of the packets are treated as "better" traffic and routers give absolute precedence to them (for example by means of a priority queuing mechanism). This mechanism can be coupled with an appropriate admission control to provide the "virtual leased line" service, which takes under control the maximum amount of EF traffic in each node of the network. A second specification within the DiffServ model, the Assured Forwarding [7], is simpler because it just classifies the traffic in several classes that have different drop probabilities in case of congestion.

The DiffServ model manages only aggregates of traffic (it does not handle single sessions or *microflows* according to the DiffServ terminology), therefore it provides looser guarantees than the IntServ model. For instance, DiffServ services are usually defined by means of Service Level Agreements (SLAs) between the customer and the network operator. SLAs are static agreements that cannot be changed at run-time by means of some form of user-driven signaling; furthermore, usually they do not apply to end-to-end services because they include only parameters like maximum sustainable burst, maximum rate at run-time and such.

This paper presents the lesson learned from an experimental network that has been defined under an EC project. Our prototype is an integrated network that is able to transport traffic using different quality classes and giving a certain degree of assurance to the customers. The objective of this project (and of this paper) is to demonstrate how existing technologies can be applied to successfully put in place a QoS-enabled network.

This paper is structured as follows. Section II presents the engineering phase from a customer viewpoint, i.e. the services that will be offered through the network. Section III presents how the network has been engineered from a technical point of view. Section IV shows some results and finally Section V gives some conclusive remarks.

II. NETWORK SERVICES

A. Standard Network Services

The fundamental choice for a network operator is to define the network services (and the corresponding prices) that will be offered to the customers. Each service corresponds to a given set of guarantees that are provided by the network. For example, an "expedited forwarding" service will guarantee that packets are delivered faster than normal packets.

In our network we defined the set of four network services listed in Table 1. The first one is the Leased Line service, an end-to-end service similar to a "virtual wire". The second one has similar characteristics, but it can be created at run-time after receiving an explicit request from the user. Premium service is

intended as a "better" service than Best Effort one; the difference among them can be seen only in case of congestion, in which Premium traffic has precedence over Best Effort. Both Premium and Best Effort are not intended to be point-to-point services; rather, they represent the maximum amount of data of that kind that can be sent by the customer. Obviously, the network operator cannot provide any guarantees regarding the amount of Premium traffic that will be delivered to the destination because it cannot know, in advance, where the Premium traffic will be directed [1].

Leased Line
It provides "absolute" bandwidth guarantees between two end-points, which must be declared explicitly.
On-Demand
It has the same characteristics of previous service, but it is created dynamically upon request. The user will ask the network for the service by means of some sort of signaling.
Premium
It assures a "better" service than the one experimented by Best Effort Traffic, but it offers no guarantees of any kind. The user knows that, in case of congestion, this traffic will have the precedence over Best Effort traffic.
Best Effort
Standard Best Effort traffic; no guarantees of any kind.

Table 1. Basic Quality of Service classes.

Additionally, we defined that each class can behave in three different ways (explained in Table 2), bringing the total number of services to twelve. The difference between a "Standard" and a "Plus" service is that the former does not allow the user to exceed the guaranteed bandwidth, while the latter allows excess traffic. Obviously there are no guarantees that the excess traffic will be delivered to the destination: this will occur only if the network has free resources and the network operator is willing to transport it[2]. In any case, the network operator can impose a limit on the amount of excess traffic allowed. "Gold" services guarantee an "express delivery", i.e. they tend to minimize the end-to-end delay. In order to keep the delay under control, a Gold class does not permit to exceed the allocated bandwidth.

Standard
A "standard" class behaves normally; this class does not support excess traffic compared to the value defined in the SLA.
Gold
This class is targeted to time-sensitive traffic. Traffic belonging to a "gold" class will be delivered as soon as possible; like previous qualifier, no excess traffic is allowed.
Plus
A "plus" class behaves normally; however this qualifier makes the service able to accept excess traffic, provided that the network has free resources and that the network operator is willing to transport it.

Table 2. The three modalities for each class

There are some exceptions to previous definitions. For instance, Best Effort traffic cannot be Gold or Plus (just the standard class exists), but also Premium Plus makes little sense. For instance, customers could decide not to send Best Effort traffic if they are allowed to exceed the Premium Plus rate, because that traffic could receive a better service in case of congestion (provided that other customers are still using Best Effort). In our network the Premium Plus is allowed, but its up to the provider to decide what to do with when that traffic exceeds the target rate.

A customer can buy several services at the same time and it can send different kinds of traffic on the same physical link. For that, the customer has to specify which part of his traffic belongs to which class; for instance, it has to agree with the network provider a way to identify the service class each packet belongs to.

B. Link Sharing

Our network supports also bandwidth distribution according to the hierarchical link-sharing [13] model. In other words, let's imagine a link that is shared between two customers (A and B in Figure 1), and that each customer reserves a certain amount of bandwidth to its real-time traffic (RT class). The remaining traffic will be assigned to the best-effort (BE) class. In case the customer A does not have any real-time traffic, a traditional bandwidth-sharing model (for example the one obtainable with a Weighted Fair Queuing scheduler) will assign the unused bandwidth to all the remaining classes. This is clearly undesirable from the point of view of customer A, which paid for the 50% of the bandwidth and it is not able to send that much. Hierarchical link-sharing will allow customer A using up to 50% of the link bandwidth, independently of the customer B traffic. This is the key idea behind link-sharing: the network operator can control exactly how bandwidth is assigned to each customer. Recursively, even customers can control how bandwidth will be used in their interior classes.

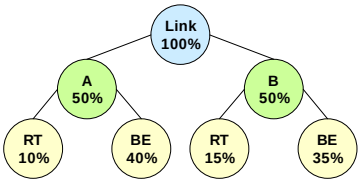

Figure 1. Hierarchical link-sharing.

The hierarchical link-sharing issue is orthogonal to the ten classes defined before. In our network, hierarchical link-sharing is carried out by defining a special "virtual class", called Virtual Container, which will not carry traffic but it will contain the standard network services inside it. In other words, the network configuration of Figure 1 requires two Virtual Containers (for customer A and B), each one containing two classes (for example a Premium Gold and a Best Effort) to transport traffic. Virtual Containers apply to end-to-end services and their creation requires defining explicitly the two end-points. Network classes inside the Virtual Container must be point-to-point, therefore only point-to-point services are allowed inside it. A special version of the Virtual Container, called Virtual Container Plus, allows the customer to exceed its allocated bandwidth.

The Virtual Container can be represented as a "virtual" link inside a "physical" link: the configuration allowed on a link will be allowed on a Virtual Container as well, and the traffic in it will be scheduled like in presence of a "true" link. This service is especially targeted for Virtual Private Networks because it gives the customer the possibility to manage its bandwidth with a finest granularity.

Traditional classes do not follow the link-sharing model. For instance, a Leased Line and a Leased Line Plus configured between points A and B and belonging to the same customer

[1] The terminology used to define the network services has nothing to do with any other terminology used in other contexts (e.g. DiffServ).

[2] The network operator could choose not to transport the excess traffic for several reasons, for example because it is expecting traffic coming from a more profitable class.

cannot exchange traffic. In other words, if the first class does not have any traffic in it, the available bandwidth cannot be exploited by the second class even if they belong to the same customer.

C. Services and costs

One of the objectives of our network is the minimization of billing costs; therefore a carefully chosen flat rate is the best solution. The billing model is presented in Table 3 is very simple and only On-Demand classes do not use a flat-rate billing model. The table lists all the classes starting from the most profitable ones; inside each service, the Gold version is always more expensive than the Plus version, which is more expensive than the standard version. Obviously, services that have assured guarantees (for example the On Demand and Leased Line) cost more than services that offer less guarantees (for example Premium). Basically, the price varies according to the allocated bandwidth and the path.

Cost of the Best Effort class is slightly different because its price depends only on the total link bandwidth provided to the customer. In other words, this price represents the amount of money due for having a network connection with that speed.

Class	Cost category
Virtual Container Plus	*Very High*
Flat rate depending on (1) the allocated bandwidth, (2) the maximum amount of gold traffic, (3) the path between two end-points. This price includes all the child classes (that come for free).	
On-Demand Gold	*Very High*
Price that depends on the requested amount of time, the allocated bandwidth and the path.	
Virtual Container	*High*
Same as Virtual Container Plus.	
Leased Line Gold	*High*
Flat rate that depends on the allocated bandwidth and the path.	
On Demand / On Demand Plus	*Moderate*
Price that that depends on the allocated bandwidth and the path. The On Demand Plus is more expensive than the On Demand.	
Leased Line / Leased Line Plus	*Moderate*
Flat rate that depends on the allocated bandwidth and the path. The Leased Line Plus is more expensive than the Leased Line.	
Premium Gold	*Moderate*
Flat rate that depends on the allocated bandwidth.	
Premium / Premium Plus	*Low*
Flat rate that depends on the allocated bandwidth. The Premium Plus is more expensive than the Premium.	
Best Effort	*Very Low*
Flat rate that depends only on the total link bandwidth.	

Table 3. Billing model; each class has a different cost.

D. QoS Guarantees

As far as bandwidth is concerned, our services are able to provide 'almost absolute' guarantees. As will be explained in Section III.A, scalability is achieved by means of a DiffServ-like model, which involves using session aggregation. However is well known that maximum burst sizes grow with the number of aggregated streams merged together. It follows that aggregation cannot guarantee absolute bandwidth guarantees over short time periods unless we have infinite buffers into the network devices. The provider (and the customer) must be aware that the absolute bandwidth guarantees provided by Leased Line and On-Demand classes must be considered over a sufficient time scale and that some packet loss may occur.

Our network does not provide delay guarantees at all. Although the literature presents several approaches in which delay guarantees can be provided in a DiffServ network (among the others, the ones based on network calculus [17] and on the core-stateless [18] idea), this is not the case for off-the-shelf components. Our objective was to make the network running; therefore we had to rely on production technologies that do not support delay bounds.

The network services previously defined are intended one-way (i.e. simplex); each class guarantees the in-order delivery except in case of network problems.

III. ENGINEERING THE NETWORK

Previous Section presented the network behavior from the customer perspective: the classes he can choose, their guarantees, their prices.

This Section presents how an operator can configure its network to carry out these services. Since the Virtual Container classes can be seen as a special case, they will not be explained in the following Sections; the entire Section III.D will later explain how these classes can be managed.

Our proposal will distinguish the access side from the core side of the network. For Access Link we intend the physical link that connects a single customer to the provider's network. All the links that are not access links will be considered Backbone (or Core) Links.

A. Quality of Service Models

Our network does not guarantee any delay parameters (apart from the fact that in some class the traffic will be forwarded faster); therefore only bandwidth guarantees are required. These can be obtained by means of an Expedited Forwarding-like model coupled with an appropriate Admission Control. Although our network does not have an explicit admission control, the network operator knows the maximum amount of traffic (split in the several quality classes) coming from each consumer. As shown in Figure 2, if customer *Send_A* will send at most A_i traffic belonging to class *i* (and customer *Send_B* will send at most B_i), the peak traffic on the C-F link (concerning class *i*) will be $A_i + B_i$. Therefore, as soon as bandwidth guarantees are concerned, each link will have a single forwarding class (for each quality class *i*) whose bandwidth will be the sum of all the sessions that use the link. The traffic will be aggregated on a '*per hop*' base and this is possible only if the network operator knows the path of each session.

The number of scheduling classes required is therefore quite limited and (in the first stage) it can be approximated with the different number of quality of service classes supported by the network. Basically, each point-to-point QoS class (Leased Line and On-Demand) will make use of the previous model.

The remaining QoS classes (Premium, Best Effort) will adopt the same model listed before, but they can provide only statistical bandwidth guarantees because it is not know, *a priory*, where the traffic will be directed. Since there are no rules that specify the amount of bandwidth to be reserved to these classes on each link, the decision is an engineering choice. For example a network provider could choose to reserve up to 20% of each link to Premium traffic and to keep the network under control in order to check if this choice is appropriate.

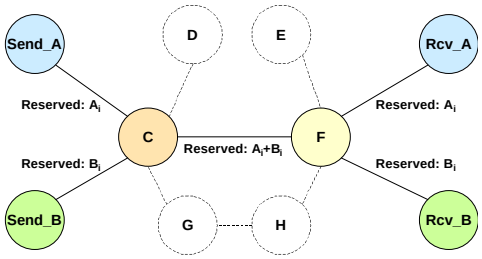

Figure 2 Quality Model: Expedited Forwarding with Admission Control.

The model adopted in our network does not implement the Expedited Forwarding mechanism exactly as it is specified in [6]. There are two main differences. First, delay-sensitive traffic is not always sent as soon as possible, while EF traffic does not have any concerns about other traffic. Since highest priority packets have absolute precedence, an increase in this traffic could bring other classes to starvation. The 'fast forwarding', obtainable with a priority queuing scheduler, is then coupled with a mechanism controlling that the class does not exceed its allocation. Second, a DiffServ model does not have any signaling mechanism, which is instead required by the On-Demand classes.

B. Access Router architecture

The architecture of the access router that is able to satisfy our objectives is very simple and it is shown in Figure 3.

The router has four different mechanisms that co-operate. A meter measures the amount of incoming traffic for each class. A marker sets the Traffic Class (TC) field of the IP/IPv6 header so that following routers will be able to identify the QoS class easily. In case the traffic is not allowed, it can be either marked with another TC value or dropped. A queue management mechanism is the third component that decides which packets have to be discarded in case of congestion. Finally, the scheduler distributes the bandwidth between the classes.

First two blocks are basically used to limit the amount of traffic inserted into the network. This will assure that the traffic is respectful of the service parameters and they control the usage of the resources. The last two blocks are devoted manage the bandwidth either in case of congestion or when a class is not using its bandwidth entirely. In the first case, they have to decide which class will have the right to transmit; in the second case, they have to decide how bandwidth will be re-assigned to other classes.

Due to lack of space, Figure 3 groups together Leased Line and On-Demand classes. In any case, Leased Line and On-Demand classes use different meters; after that, their traffic is aggregated and treated in the same way. Therefore the scheduler can be configured in order to serve the traffic aggregate (On Demand Plus the corresponding Leased Line), making no differences where the traffic is coming from.

1) Meter

This component is very simple: it controls the amount of traffic sent by the customer, comparing it with the service contract signed with the provider. The meter can accept either marked traffic (in case the user wants to mark the traffic it its side of the network), or it can identify the proper class by means of some network parameters (usually based on the 5-tuple Source/Destination Address, Protocol, Source/Destination Port). All the traffic will be forwarded to the marker/dropper in order to take the appropriate actions.

2) Marker/Dropper

The marker sets the TC field of each packet using distinct values for each class. This allows queue manager and scheduler to differentiate the service among classes by checking at this parameter only.

The management of the excess traffic is the fundamental choice that determines how this block will work. Basically there are three alternatives:

- **Discarding excess traffic**: this makes the management simpler, but it does not take into account that excess traffic can be transmitted for free when the network is not congested.
- **Inserting excess traffic into another class with lower privileges**: this is not a good choice because there are no guarantees that packets belonging to a single session, transmitted using two different classes, will arrive in order. Out-of-order deliver should be avoided because it is especially harmful for TCP traffic, although there are some cases in which out-of-order delivery cannot be avoided due to IP problems.
- **Marking the traffic as 'excess traffic'** and inserting it into the same forwarding class of the 'regular' traffic: this does not suffer from the reordering problem, although it delegates the appropriate action (forward / drop) to the next blocks (queue management and scheduler).

In our network we chose the first option for delay-sensitive traffic (i.e. 'Gold' classes) in order to control the amount of Gold traffic inserted into the network. Moreover the priority scheduler cannot provide low delays when the high-priority traffic becomes an important part of the overall traffic because the delay experimented by Gold classes increases.

For the remaining classes we chose the third option: the traffic exceeding the service contract is marked "*out of profile*" traffic, i.e., the TC value is different from the one assigned to "in-profile" traffic. For that, the service will depend on the rules specified on queue managers and schedulers. Each class can potentially have two different values for the TC (one for in-profile and another for the out-profile traffic); values are shown in Figure 3.

This block appears like a "manual" admission control because it keeps under control the amount of traffic inserted into the network; this explains why our DiffServ-like model works.

According to the DiffServ terminology identified in [8], each TC value corresponds to a Per Hob Behavior (PHB), while the TC values assigned to the *in* and *out of profile* traffic correspond to a Per Domain Behavior (PDB). For instance, a PDB is the expected treatment that an identifiable group of packets will receive from "edge-to-edge" of a DiffServ domain, and it is associated with a group of PHBs.

3) Queue manager

As shown in Figure 3, the number of queues is less than the number of QoS classes. Keeping the Virtual Container classes apart, there are only four queues:

- **Guaranteed Queue**: it contains all the Leased Line and On-Demand traffic (Standard and Plus)
- **Guaranteed Gold Queue**: it contains the Leased Line Gold and On-Demand Gold traffic
- **Data Gold Queue**: it contains the Premium Gold traffic
- **Data Queue**: it contains the Premium (Standard and Plus) and the Best Effort traffic.

These queues are managed with a WRED [10] (Weighted RED) mechanism, i.e. a RED-based [9] algorithm that discards incoming packets based on the average queue length and on a different sensitivity parameter for each type of traffic. Basically, WRED looks like several REDs applied to the same queue and with different parameters for drop probability, which are selected according to the TC value of the incoming packet. This combination can selectively privileges 'better' traffic when the interface begins to get congested, and provides differentiated performance characteristics for different classes of service. For instance, all packets in the Guaranteed Queue will be forwarded if the link is not congested. As soon as the traffic increases (and the scheduler is no longer able to give service to all packets in queue), the queue grows and the queue manager will drop some excess packets (marked with TC equal to C in Figure 3). If the congested state persists, the queue manager will discard all the excess traffic.

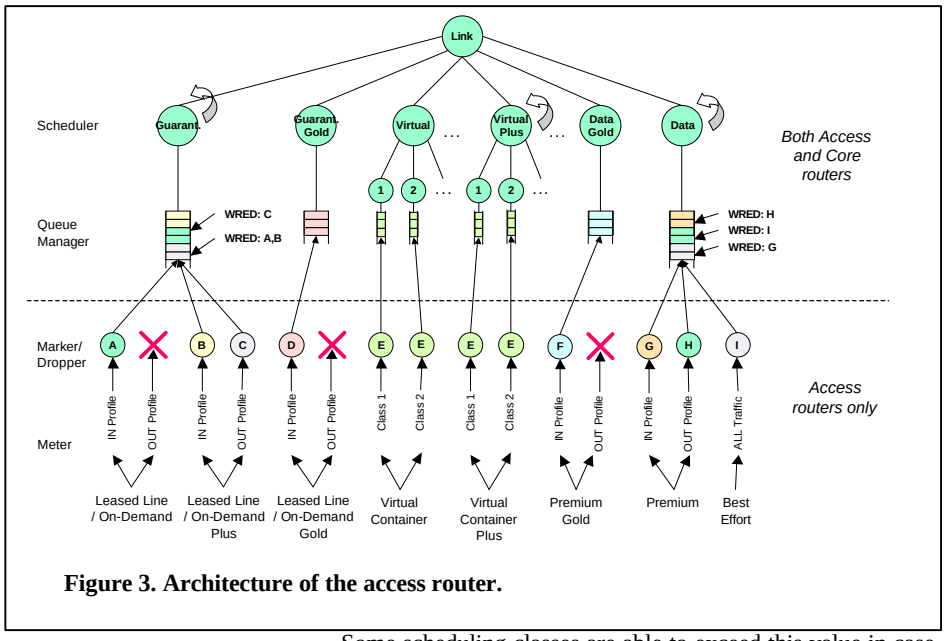

Figure 3. Architecture of the access router.

The network provider chooses the values for the dropping probability according to its service parameters. Although each QoS class has a different TC value, in our network we chose to treat in the same way the traffic coming from Leased Line and On-Demand (marked with A), and the 'in-profile' traffic coming from Leased Line Plus and On-Demand Plus (marked with B). However the network provider will have the option to differentiate the traffic by choosing different WRED settings for each TC value.

4) Scheduler

The scheduler operates with four scheduling classes (that share the same link on which the scheduler is active) corresponding to the four queues defined in the previous Section. Each class is configured with a certain amount of bandwidth that is guaranteed also when the link is congested. This value is calculated in order to satisfy the service contract. For example, the bandwidth assigned to the Guaranteed scheduling class will be the sum of the bandwidth granted to the Leased Line (Standard and Plus) and the On-Demand (Standard and Plus) QoS classes (Section III.A):

$$(1)\quad B_{Guaranteed} = \sum_{ll=1}^{\#LeasedLine} B_{ll} + \sum_{llp=1}^{\#LeasedLinePlus} B_{llp} + \sum_{od=1}^{\#OnDemand} B_{od} + \sum_{odp=1}^{\#OnDemandPlus} B_{odp}$$

Some scheduling classes are able to exceed this value in case the link is not congested. These classes are called "borrowable" because are able to 'borrow' bandwidth from the ones that do not have enough traffic. Scheduling classes that transport Gold traffic are not borrowable, therefore the amount of bandwidth imposed is strictly respected. Vice-versa, data classes can exceed their allocation and they can transport excess traffic (the one marked as 'out of profile') when the network is not congested.

Particularly, the network operator cannot know in advance the amount of Data Gold traffic present in the network. Since Data Gold does not represent a point-to-point service, the amount of that traffic can converge over few links even if the marker/dropper does not allow excess Gold traffic, thus exceeding the planned values. Therefore we define the Gold classes as 'not borrowable' in order to keep the delay-sensitive traffic under control and not to bring the 'regular' traffic to starvation (Section III.A).

The excess traffic is distributed according to the bandwidth assigned to each scheduling class: if the bandwidth assigned to the Guaranteed class is X and the one assigned to the Data class is 2X, then 2/3 of the excess traffic will be the one waiting into the Data class.

Gold traffic is forwarded faster thanks to a bandwidth-limited Priority Queuing (PQ) mechanism. In other words, the scheduler has two classes at higher priority and the traffic in them will be forwarded as soon as possible, provided that these classes are not exceeding their allocated bandwidth.

C. Backbone router architecture

Although the architecture of the access router is very simple (no per-flow state is kept and the amount of resources needed is very low), the core router is even simpler in order to satisfy scalability issues. The core router does not have the Meter/Marker/Dropper because these blocks are devoted to the admission control that has already been done on the access side of the network. The core router needs only to manage queues and to schedule packets according to the bandwidth settings.

Queue management is performed in the usual way: the network operator will select the proper thresholds for the WRED algorithm in order to discard excess traffic in case of congestion. Furthermore, core routers have to configure properly the bandwidth on scheduling classes in the same way used by the access side routers (Section III.B.4). A scheduling class will be configured as the sum of the bandwidth of all the contributing classes. For instance, both access and core routers must take into account the number of sessions that use the link and reserve the bandwidth accordingly.

A difference arises in the Data / Data Gold scheduling classes because traffic in there is not predicable deterministically. In them, the network provider has to assign a certain amount of bandwidth to those classes, hoping that it is enough. Our suggestion is to make a statistical evaluation of the traffic *a priori*, then monitor the network and see if the bandwidth reserved to those classes is sufficient. If the Data / Data Gold classes have a continuous backlog, the bandwidth assigned to them is probably not enough and it needs to be modified accordingly.

D. Virtual Container classes

Virtual Container classes are outside the previous description. Basically, each Virtual Container class virtualizes a link with a certain amount of bandwidth. Therefore, the scheduler cannot aggregate that traffic, and each Virtual Container corresponds at least to a scheduling class. This explains why the Virtual Container classes are expensive (Table 3): the number of Virtual Container should be kept low in order to make the network scalable. Moreover, the available codes into the TC field could not be enough to classify also Virtual Container traffic; therefore the standard classification method based on address, protocol and port of each packet could be needed. This, again, adds complexity to the network routers.

Inside each Virtual Container class, the same point-to-point classes presented in previous Sections can be defined: the network provider (and the customer) has to decide which is the best configuration, taking in mind scalability issues.

E. Distributing the Bandwidth

Perhaps one of the most important points of this paper is related to an economic viewpoint. The key idea here is to differentiate the way bandwidth is assigned to the scheduling classes on the access link and on the backbone in order to maximize provider's revenues and to respect the service contract with the customer.

Bandwidth on the access-link is allocated almost entirely in a static way. The total link bandwidth is partitioned between Guaranteed, Virtual Container and Data scheduling classes:

$$(2)\quad B_{AccessLink} \geq B_{Guaranteed} + B_{GuaranteedGold} + B_{DataGold} + B_{Data}$$

Best Effort class does not have any guaranteed bandwidth (for instance, the price for Best Effort traffic does not depends on the bandwidth allocated to that class) and it gets the bandwidth not allocated to other classes, therefore:

$$(3)\quad B_{Data} \geq B_{Premium} + B_{PremiumPlus}$$

On the access link, On-Demand services will be allowed as soon as the bandwidth guaranteed to other classes (i.e. all but the Best Effort one) is left untouched. In other words, On-Demand services will borrow bandwidth only from the Best Effort class: new requests will be accepted provided that the sum of previously allocated services (i.e. all the existing classes but Best Effort) and the new request is less or equal than the total link bandwidth (Equation (2)). Best Effort traffic can then starve; the user must be aware that if too many classes are allocated, this will certainly happen.

Bandwidth distribution on backbone links is different from the one on the access side of the network. The key point is that the network provider gives absolute guarantees to the Guaranteed scheduling classes, but it gives statistical guarantees to the Data classes. That is, if several customers are concentrating their Premium traffic onto a small number of links, the traffic will start being discarded and the customer knows that the network is congested. However, let us suppose that a new On-Demand session asks for resources: the network can accept the request and it will probably reduce the amount of bandwidth assigned to the Data class, i.e. it will reduce the amount of Best Effort traffic. If this process is repeated again and again, soon Best Effort traffic will be reduced to zero. In case of new On-Demand requests, the network provider can choose to reduce the bandwidth allocated to the Premium class. Customers could experiment drops in the Premium class but they will believe that the network is congested and they will not protest against the network provider[3]. In other words, this behavior is still compatible with the service contract.

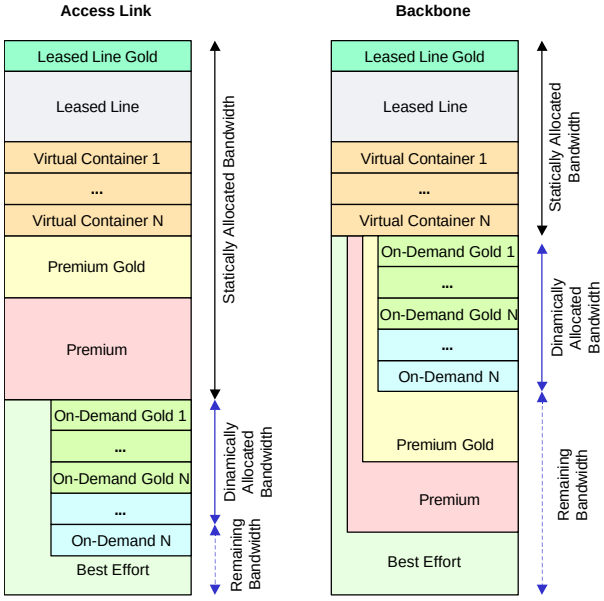

Figure 4. Bandwidth allocation on the access link and on the backbone; *Plus* classes are not shown.

Figure 4 shows how this is possible: only the Best Effort traffic can starve on the access link. On backbone links, all classes with statistical guarantees can starve and this depends whether the network operator receives a request for a more

[3] The provider can put in place a similar behaviour also when an existing session must be re-routed, for example because a network path does no longer exist.

profitable class. In this case it could reduce the bandwidth assigned to the least profitable classes bringing to starvation Best Effort first, then Premium traffic, and so on. However the network provider must be careful to keep the amount of Gold traffic under control (i.e. it must not accept too much Gold traffic): the delay parameters will start worsening if it becomes an important percentage of the overall traffic.

This process must be applied carefully: the customer will become upset if its Premium traffic will be dropped too often. Therefore, the network provider has to monitor the network and take the appropriate actions (e.g. increase the link bandwidth) if this "cheating" process happens frequently. For instance, also customers that use only Best Effort could become upset if their traffic is dropped too often.

IV. ON-FIELD EXPERIENCE

Our network has been tested with the *ns-2* simulator and implemented using both FreeBSD boxes (with the ALTQ [14] toolkit on it) and Cisco routers.

Except for some minor differences we were able to find all the required basic blocks. Basically, we needed hierarchical schedulers (we used the standard one in Cisco and D-CBQ [12] in ns-2 and FreeBSD), queue managers (WRED in Cisco and RIO[4] [16] in FreeBSD), marker/droppers and meters.

The most important problem is that the bandwidth of the scheduling classes must be resized at run-time because of the new requests coming from On-Demand classes. In that case, the router should take into account the parameters of the new sessions and modify (e.g. resize) the corresponding classes accordingly. However, this feature is currently unavailable. A proposal concerning RSVP flow aggregation [11] is not applicable because it supposes that sessions share ingress and egress routers in an aggregated domain. However this is not the case since the sessions are aggregated on a per-hop basis.

In the first stage, we used a pragmatic approach: On-Demand classes are managed like Leased Line ones, i.e. the customer must explicitly call the network operator in order to create a new On-Demand class. In this way, no signaling (i.e. RSVP requests) is allowed from the customer and the network operator will statically (and manually) configure On-Demand classes along the path. For instance, the network provider can implement a web-based reservation procedure instead of using signaling protocols. The customer will book through a web page and a set of scripts will modify the router configuration at run-time.

The lack of a proper signaling protocol is clearly a non-trivial limitation, therefore we started working on a COPS-RSVP-like [15] implementation that is able to accept RSVP request and configure the proper forwarding classes. Our COPS-RSVP implementation modifies the standard behavior in that no new scheduling classes are created into the router; instead, an existing class is resized. In any case, if a new class has to be created, it can be appended to another 'parent' class according to the Virtual Container paradigm. In our prototype routers still exchange the RSVP refresh message of each session (they do not use "bulk" reservations like in [11]), but the traffic is transmitted using a limited number of forwarding classes. Although RSVP refresh messages are not aggregated, this is a negligible overhead compared on keeping the state of each session. This implementation is 'work in progress' and it is available only on FreeBSD machines.

[4] RIO (RED with In and Out priority) is a simplified version of WRED that supports only two types of traffic (the IN and the OUT one).

A. Simulations

Simulations have been made in order to characterize the behavior of a simple network engineered with the above criteria.

Figure 5 shows the testbed used in the simulations. Several CBR sources have been configured that sends data from left to right. These sources have been assigned to guaranteed classes (Leased Line, Leased Line Plus, Leased Line Gold) in order to know the maximum amount of traffic present in each class. Traffic in these classes is always slightly less than the maximum value in order not to stress the network.

Several TCP sources with exponential activation times and random durations are activated in the network. These TCP sources send from left to right and vice versa in order to simulate real traffic, and they have been assigned to the remaining classes. The Data Gold class transports traffic generated by CBR sources with exponential on-off distribution; often the amount of traffic in that class exceeds its maximum capacity. The most congested link is the third one because of the higher number of sources that uses that link.

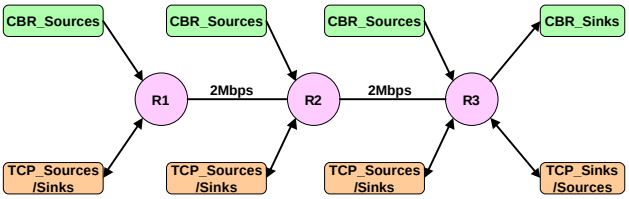

Figure 5. The network used during simulations.

The bandwidth distribution obtained by our network is shown in Figure 6, while the percentage of the dropped packets is shown in Figure 7. These graphs show that the bandwidth distribution obtained is the one expected. The differences are due mostly to the approximations of the scheduling algorithm that has been used.

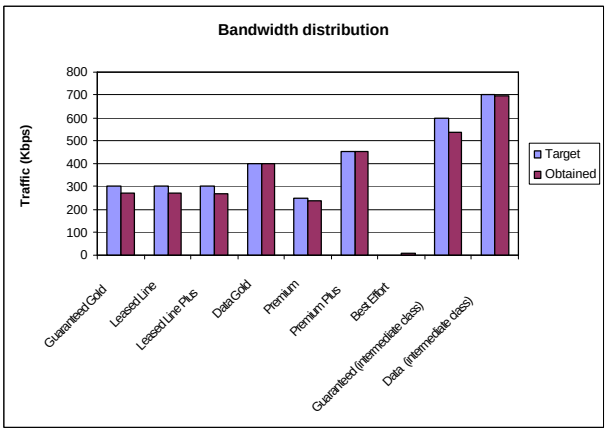

Figure 6. Bandwidth on the most congested link.

Also packets dropped are the ones expected: no drops occur in the guaranteed classes, while Premium, Best Effort and Premium Plus shows a certain percentage of drops compared to the global amount of packets generated by these TCP sessions. The Data Gold class experiment drops as well because

sometimes the amount of traffic that is larger than the class capacity.

In our simulations we decided not to shrink to zero the least profitable classes in order to show the behavior of the network in case of heavy Data load. In any case, the Best Effort class is heavy penalized compared to the Premium traffic and the obtained bandwidth is almost zero. Also the amount of drops is definitely worse that the other classes although the TCP NewReno sources are able to adapt themselves to the available bandwidth.

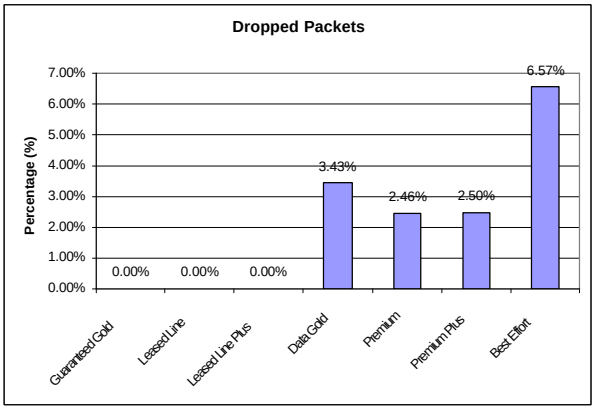

Figure 7. Drops on the most congested link.

The scheduling delays experimented by the packets (shown in Figure 8) demonstrates that the Gold classes guarantee far smaller delays than the other classes. The higher delays showed by the fifth trace (Data Gold class when the traffic exceeds the target bandwidth) confirms that when the traffic exceeds the class bandwidth the packets are queued and the delay increases. This delay can be limited by decreasing the buffer's capacity of the routers at the expense of higher drops. Obviously, the delay experimented by the data Gold traffic when the link is not congested (fourth trace) is comparable with the one experimented by the Leased Line Gold.

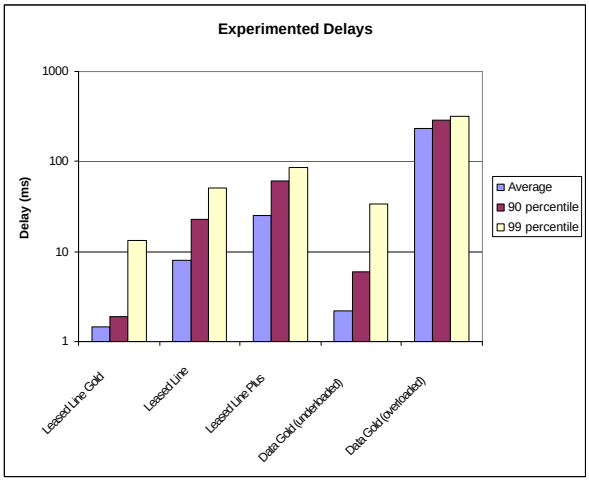

Figure 8. Delay distribution on the most congested link.

V. CONCLUSIONS

This paper presents several contributions. First, it identifies a set of network services that can be representative of a commercial service implemented by a network provider; these services are defined in terms of their characteristics from the end-user point of view. Second, after defining the quality model (Diff-Serv-like), it identifies the building blocks needed to implement these services into the network; then it proposes a configuration model that takes in mind performance and scalability. Third, it identifies some tricks that can be exploited by the network provider to increase its revenues while still respecting the service contract signed with the customer. Fourth, it verifies the applicability of the solution using both a simulative approach (in order to verify that the bandwidth guarantees are respected) and by creating an experimental network with commercial (Cisco) and experimental (FreeBSD) routers. Finally, it proposes some modification to the currently defined signaling protocols in order to be able to resize forwarding classes instead of simply creating new ones.

The results are interesting since they demonstrate that advances services can be implemented in a network built with off-the-shelf components. Furthermore, even most interesting results could be obtained by modifying the currently deployed signaling protocols in order to better support our QoS model. Preliminary results on Cisco routers and FreeBSD machines confirm the findings obtained by simulations; therefore they are not reported here.

The most critical aspect of this work is the need to monitor the network in order to keep the traffic under control. However any DiffServ-like model shares this problem. A valuable extension of this work could be the implementation of a monitoring framework, perhaps integrated into a COPS model. A COPS Policy Decision Point (PDP) could collect information coming from the network and it could take the appropriate decisions (selected by the network operator) when the network changes its state. For instance, this represents a general problem of IP networks that were not engineered for integrated services and whose management model is definitely poor compared to other technologies.

ACKNOWLEDGEMENTS

The author wishes to thank the member of the IRISI[5] project and the Telecom Italia Research Laboratories (TILab), which partially supported this work.

REFERENCES

[1] R. Braden et al., *Integrated Services in the Internet Architecture: an Overview*, RFC 1633, IETF Network Working Group, June 1994.

[2] J. Wroclawski, *Specification of the Controlled-Load Network Element Service*, RFC 2211, IETF Network Working Group, September 1997.

[3] S. Shenker et al., *Specification of Guaranteed Quality of Service*, RFC 2212, IETF Network Working Group, September 1997.

[4] R. Braden et al., *Resource ReSerVation protocol (RSVP) - version 1 functional specification*, RFC 2205, IETF Network Working Group, September 1997.

[5] Inter-Regional Information Society Initiative, a program launched by the Piedmont's regional government and founded by the European Commission.

[5] S. Blake et al., *An Architecture for Differentiated Services*, RFC 2475, IETF Network Working Group, December 1998.
[6] V. Jacobson et al., *An Expedited Forwarding PHB*, RFC 2598, IETF Network Working Group, June 1999.
[7] J. Heinanen et al., *Assured Forwarding PHB Group*, RFC 2597, IETF Network Working Group, June 1999.
[8] K. Nichols and B. Carpenter, *Definition of Differentiated Services Per Domain Behaviors and Rules for their Specification*, RFC 3086, IETF Network Working Group, April 2001.
[9] B. Braden et al., *Recommendations on Queue Management and Congestion Avoidance in the Internet*, RFC 2309, IETF Network Working Group, April 1998.
[10] Cisco Systems, *Random Early Detection (RED)*. Available at http://www.cisco.com/warp/public/732/Tech/red/.
[11] F. Baker, *RSVP Reservation Aggregation for IPv4 and IPv6 Reservations*, RFC 3175, IETF Network Working Group, September 2001.
[12] F. Risso, *Decoupling Bandwidth and Delay Properties in Class Based Queuing*, Proceedings of the 6th IEEE Symposium on Computers and Communications (ISCC 2001), Hammamet, Tunisia, July 2001.
[13] S. Floyd and V. Jacobson, *Link Sharing and Resource Management Models for Packet Networks*, *IEEE/ACM Transaction on Networking*, Vol. 3 No. 4, August 1995.
[14] K. Cho, *A Framework for Alternate Queueing: Towards Traffic Management by PC-UNIX Based Routers*, Proceedings of USENIX 1998 Annual Technical Conference, New Orleans LA, June 1998.
[15] S. Demiliani, *Engineering a New Generation Network*, Laurea Thesis Dissertation, Politecnico di Torino, work in progress.
[16] K. Nichols, V. Jacobson and L. Zhang, A Two-bit Differentiated Services Architecture for the Internet, RFC 2638, IETF Network Working Group, July 1999.
[17] J.C.R. Bennett, K. Benson, A. Charny, W. F.Courtney, J. Y. Le Boudec, *Delay Jitter Bounds and Packet Scale Rate Guarantee for Expedited Forwarding*, to appear in ACM/IEEE Transactions on Networking. June 2002.
[18] I. Stoica, S. Shenker and H. Zhang, *Core-Stateless Fair Queueing: Achieving Approximately Fair Bandwidth Allocations in High Speed Networks*, Proceedings of SIGCOMM'98, Vancouver, Canada, pp. 118-130.