SSC Tech. Report SSC/1998/020,
http://sscwww.epfl.ch, May 1998.

A Survey of Differentiated Services Proposals for the Internet

Constant Gbaguidi[1], Hans J. Einsiedler, Paul Hurley, Werner Almesberger, and Jean-Pierre Hubaux.

***Abstract*- Differentiated services are a suitable solution to Quality of Service (QoS) provisioning in the Internet while the number of users keeps growing. The solution is suitable, because it scales well with increasing number of network users and it does not alter the current Internet paradigm much. In this article, we review the state of the art in this "new" area, and compare some of the main existing differentiated services architectures. We outline the common solutions across these architectures, thus paving the road to a unified architecture. Lastly, we mention the issues that have not been thoroughly addressed yet.**

1. Corresponding author. Institute for computer Communications and Applications (ICA), EPFL-DI-ICA, IN-Ecublens, 1015 Lausanne, Switzerland. Phone: +41-21-6934679. Fax: + 41-21-6936610. Email: constant.gbaguidi@epfl.ch.

1. Introduction

The concept of differentiated services is as old as the Internet. Formerly, there was the basic need to differentiate the user data from control and management information. The latter requires less bandwidth, but much higher reliability, than the former [16]. A field in the header of the Internet Protocol (IP) Data Unit was designed for these purposes: the Type-of-Service field. The octet dedicated to this field indicates the specific treatment that the packet expects to receive from the network. The semantics of the Type-of-Service field is as follows:

- 3 bits are dedicated to the indication of the packet precedence (i.e., the importance or priority of the datagram)
- 3 bits define the Type of Service[2] (TOS) which corresponds to the Quality of Service (QoS) expected by the IP datagram: 1 bit is used to indicate the delay constraints (set to 0, this bit signifies normal delay, while the value 1 denotes low delay), 1 bit is used to express throughput requirements (it is set to 0 for normal throughput and 1 for high throughput), and the last bit is used to indicate the level of reliability (loss sensitivity) required
- the remaining 2 bits are reserved for future use. They must be set to zero meanwhile.

2. The TOS is part of the Type-of-Service field. To avoid confusion, we do not use any acronym for the latter field whenever we refer to it.

The specification of the TOS field (the 3 bits indicating quality of service requirements) was refined by Almquist [3]; instead of 3 bits, the TOS field was extended to 4 bits which are used to require minimization of the delay, maximization of the throughput, maximization of the reliability, minimization of the monetary cost, or normal (best-effort) service.

We might have expected the IETF to use the TOS when looking for a solution for QoS. However, the first reflex was to mimic the solution adopted for telecommunication networks, i.e., to create a lightweight version of "virtual circuit". This reflex is exemplified by the work carried out by the Integrated Services Group (IntServ) at the Internet Engineering Task Force (IETF). As explained in Section 2, this solution requires keeping a great deal of state information because of its flow-based nature. The IntServ solutions therefore could not scale well as the number of users keeps growing. For this reason, the IETF "revived" the TOS field for a new purpose: implementation of services with different profiles. Packets belonging to differing service profiles are handled differently on their way through the network.

In this article, we review the state of the art in the area of differentiated services, and we identify the pending issues of relevance. In Section 2, we give a comprehensive background on QoS provision in the Internet. This background lays the foundations for a better understanding of the motivation behind the concept of differentiated services: We will see that some of the ideas of the IntServ group are re-used with differentiated services. In Section 3, we review some of the relevant differentiated services architectures at play. We do not pretend to do an exhaustive review (in terms of all existing architectures), but we present the most important trends in the area. Concluding remarks are given in Section 4.

2. Background: The IntServ Contribution

The IETF proposed, many years ago, an architecture for integrated services in the Internet [4]. Much of the content of this Section is taken from this reference. The architecture proposed by the IntServ group was based on flow differentiation, in a way that mimics QoS enforcement in the telecommunications network.

In section 2.1, we present the services concerned with the integration sought by the IntServ group. In section 2.2, we present the IntServ service model. In section 2.3, we present the IntServ Reference Implementation Framework (RIF). An overview of the resource reservation protocol recommended is presented in section 2.4, followed by concluding remarks in section 2.5.

2.1 IntServ Services

The IntServ group identified three main categories of services to be concerned with the integration: the traditional best-effort services, real-time services and controlled link-sharing services.

Best-effort services are those that we currently experience on the Internet. They are characterized by the absence of any QoS specification at all. The network delivers the quality that it actually can. Examples of best-effort services are elastic applications, which tolerate the data not to arrive on time and wait for it. Several categories of elastic applications may be distinguished, e.g., interactive burst (Telnet, X, NFS), interactive bulk transfer (FTP), and asynchronous bulk transfer (electronic mail, FAX).

Real-time services are time-critical services that have very stringent requirements in terms of end-to-end delay, probability of loss and bandwidth. They might require from the network a *guaranteed service* - with a perfectly reliable upper bound on the delay - or a *predictive service* - with a fairly reliable delay bound. To meet the requirements of real-time applications, the network needs a characterization of the traffic that will be generated over the time period of the communication session. This characterization may be easy for retrieval services, such as video-on-demand, whose traffic can be easily controlled; statistics about a movie can indeed be easily computed. On the other hand, for "live" or interactive services (such as multiparty interactive games or even videoconferences), characterizing the traffic from the source has proved to be a difficult task.

Controlled link-sharing is a service that might be requested by network operators when they wish to share a specific link among a number of traffic classes, a traffic class relating to a group of users, a protocol family, and so forth. Network operators may set some sharing policies on the link utilization among these traffic classes. Specifically, some percentage of bandwidth may be assigned to each traffic class. Controlled link-sharing may also be needed when a link is shared among many companies which want to ensure that each of them is getting a minimum service. The link-sharing service is thoroughly addressed in [13].

We present next the IntServ model, which is made up of the requirements that the network must fulfill in order to support effectively the services concerned with the integration sought by the IntServ group.

2.2 The IntServ Model

A service model is defined as "a set of service commitments" [4], meaning that it should exhibit the *commitments that the network must fulfill* in order to support integrated services. The commitments identified in [4] can be classified into three categories: QoS commitments, resource-sharing commitments, and resource reservation commitments.

QoS Commitments. The QoS commitments considered are the bounds on the maximum and minimum delays.

The IntServ group focused on two categories of commitments: guaranteed service [24] and controlled-load service [27]. The *guaranteed service* provides the users (applications) with an assured amount of bandwidth, firm end-to-end delay bounds, and no queuing loss for flows that conform to the parameters negotiated at the connection setup. These parameters are classified into traffic descriptors (`Tspec`) and reservation characteristics (`Rspec`) [25][26]. The `Tspec` parameters are: the peak rate of the flow, the bucket depth (policing is done using a token bucket[1]) which is negotiated so as to correspond to the flow burst, the token bucket rate (average rate of token generation), the minimum policed unit, and the maximum datagram size. The `Rspec` parameters are: the bandwidth (amount of information to be processed within a unit of time), and the slack term. The latter represents the amount by which the end-to-end delay bound will be below the end-to-end delay required by the application.

The *controlled-load service* does not provide the network users with any firm quantitative guarantees. It simply assures that the users will get a service that is as close as possible to the one received by a best-effort flow in a lightly loaded network. This

1. Token bucket is explained in section 3.3.

assurance is granted, provided the flow conforms to the traffic characteristics (`Tspec`) negotiated at session setup. Examples of applications that can accommodate the controlled-load service are adaptive real-time services. Predictive real-time applications may also request the controlled-load service.

Resource-sharing Commitments. The resource-sharing commitments are enabled for the management of the network by the collective entities (e.g., corporate companies) that operate it. Hence, these commitments concern the collective policies that affect all the flows transported by the network. They fall into *three categories*: multi-entity link-sharing (sharing among companies, agencies, and the like), multi-protocol link-sharing (sharing among protocol families in order to prevent one family from starving the resources from the other families), and multi-service link-sharing (sharing among services, such as real-time and best-effort services). The IntServ resource-sharing service was influenced by the valuable experiments on multiple sharing scenarios conducted by Floyd *et al.* [13].

Resource Reservation Commitments. The resource reservation commitments concern the efficient use of the network resources in a way that avoids resource wastage. The reservation model must commit:

- to run properly in a multicast environment without resource wastage. When it comes to sparing the network resources, multicast must not be considered as a simple extension of unicast. Appropriate multicast mechanisms that save the network resources must be designed
- to accommodate heterogeneous service needs; for instance, the branches of the multicast tree may deliver different QoS levels
- to be designed around the elementary action of adding/removing an edge (sender or receiver) from the on-going sessions, without affecting the QoS being delivered to the other parties
- to be robust and scale well to large multicast groups, just like MBone [19] operates, unfortunately without resource reservation
- to provide for resource reservation in advance, as well as resource preemption.

To implement the commitments outlined above, four components are proposed by Braden *et al.* [4]: the packet scheduler, the admission control routine, the classifier, and the reservation setup protocol. The Reference Implementation Framework (RIF) that embeds these components is presented in the next section.

2.3 The Reference Implementation Framework (RIF)

We present the four components in the RIF and describe the latter's implementation for a router.

Packet Scheduler. The packet scheduler manages the forwarding of different packet streams using a set of queues and perhaps other mechanisms such as timers. Therefore, packet scheduling must be implemented at the point where packets are queued. This point typically corresponds to the link layer. The packet scheduler can embed a traffic estimator and policing functions.

Packet Classifier. The packet classifier operates upstream of the packet scheduler and maps each incoming packet into some class, in such a way that all packets in the same class get the same treatment from the packet scheduler. A class is an "abstraction that may be local to a particular router; the same packet may be classified differently by different routers along the path" [4]. Two approaches are possible for the classifier if the latter is intended to support QoS provisioning: (1) to abandon the IP

datagram model in favor of a virtual circuit model, or (2) to be allowed to look at more fields in the IP packet, such as the source address, the protocol number and the port. The IntServ group recommends the second approach which basically does not alter the existing IP scheme much.

Admission Control Routine. The admission control routine implements the decision algorithm that a router or host uses to determine whether a new flow can be granted the requested QoS without impacting earlier guarantees. The admission control routine takes responsibility for enforcing the reservation policies set by the network administrator. As such, the admission control routine must be consistent with the service model if the network is expected to behave as desired. If the admission control routine contradicts the service model, then the applications would never have their requirements satisfied. An admission control routine for the controlled-load service is proposed by Jamin *et al.* [17].

Reservation Setup Protocol. An adequate reservation protocol faces a fourfold trial mainly related to routing:

- to find a route that supports resource reservation
- to find a route that has sufficient unreserved resources for a new flow
- to adapt to route failure
- to adapt to a route change without failure.

The Resource ReSerVation Protocol (RSVP) [4,5,25,26,28] was recommended by the IntServ group. It is presented in section 2.4.

Implementation of the RIF for a Router. The instantiation of the RIF for a router is illustrated in Fig. 1. The routing agent is in charge of computing the routing tables held in the routing database and used further on by the packet scheduler to map the incoming datagrams to their corresponding output port. The reservation setup agent deals with setting aside the resources necessary for guaranteeing the QoS requested for the new flow. This resource allocation is performed only if the admission control function returns successfully. If so, the reservation setup agent configures appropriate fields in the traffic control database so that the requirements of the new flow get fulfilled as the packets arrive. The management agent serves to manage the router. Underneath the control and management layer, there is a media transfer layer which deals with packet processing and forwarding.

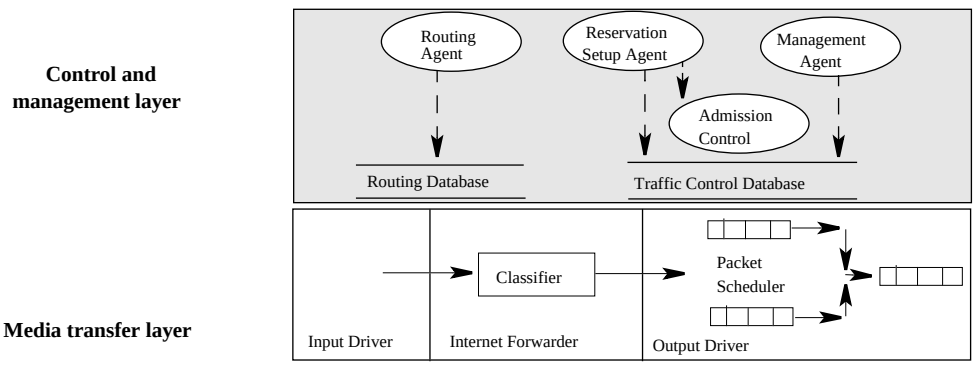

FIG. 1. Implementation of the RIF for a router.

The reservation protocol recommended by the RIF (i.e., RSVP) is presented in the next section.

2.4 The Resource ReSerVation Protocol (RSVP)

Description. The design goals of RSVP are [28]:

- to accomodate heterogeneous receivers
- to adapt to changing multicast group membership
- to exploit the resource needs of different applications in order to efficiently use the network
- to allow receivers to switch channels (select specific senders)
- to adapt to changes in the underlying unicast and multicast routes
- to control protocol overhead so that it does not grow linearly (or worse) with the number of participants
- to make the design modular in order to support heterogeneous underlying network technologies.

The use of RSVP is illustrated in Fig. 2 [25]. Sender S1 produces streams which are consumed by the receivers RCV1-RCV3. To make its service available to potential recipients, S1 must send the description of its flow to the routers on the multicast tree which is understood to be set up by other means (RSVP is not a routing protocol!). This flow description is sent in a message called `Path`, which carries the following information:

- `Phop`: the address of the previous RSVP-capable node that forwards this `Path` message
- the `Sender Template`: a filter specification identifying the sender in terms of the latter's IP address and, optionally, flow sending port
- the `Sender Tspec`: the traffic characteristics of the flow generated by the sender
- an optional `Adspec`: advertisement generated by the sender, updated at each hop along the communication path, and possibly used by the receivers to determine the level of reservation that suits better their needs. The `Adspec` provides Default General Parameters (e.g., the minimum end-to-end path latency, the path bandwidth, the integrated services hop count, and the path's maximum transmission unit), and the description of the type of network commitment available (currently either guaranteed service [24] or controlled-load service [27]). `Adspec` may be used to force all the receivers in a multicast session to choose the same service. For the time being, RSVP does not allow receivers of the same flow to select differing network services.

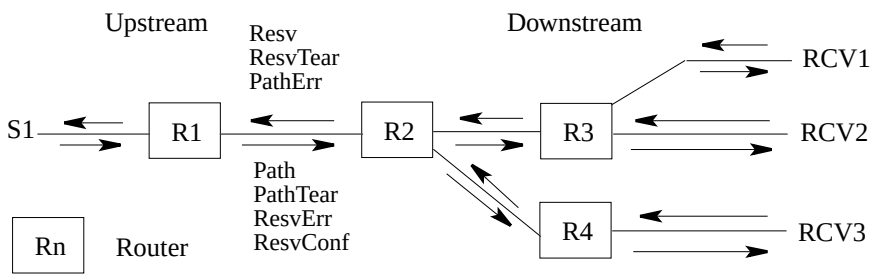

FIG. 2. Description of the use of RSVP messages [25].

`Path` messages are intercepted and checked by each router on the distribution tree. Whenever an error is detected, the router drops the message and notifies the event to the sender through a `PathErr` message. If the `Path` message is valid, then the router proceeds as follows:

- it updates the path state entry for the sender identified by the `Sender Template`; a new entry is created if none has already been allocated to this sender
- it sets a cleanup timer to the cleanup timeout interval; the cleanup timer is managed for each individual path state entry; its expiration triggers the deletion of the entry. Therefore, the `Path` message has to be sent periodically as long as the path is alive, in order to refresh the path state
- it generates a `Path` message with respect to the new state entry, and forwards the message downstream to the routers in the distribution tree.

RSVP allows the sender to expedite the process of path tear-down using the `PathTear` message. Hence, a path can be torn down independently of the cleanup timeout. `PathTear` messages are generated whenever a path is deleted in order to inform the other routers on the distribution tree.

The rest of the messages in Fig. 2 concern resource reservation by the receivers. As indicated earlier, the `Path` message may carry an `Adspec` which provides information that can be used by the receivers to calculate the amount of resources that they need in order to receive the flow. The receiver sends a `Resv` message to open a session with the sender. This message carries:

- an indication of the reservation style (see below)
- a filter specification, `Filterspec`, which identifies the sender; its format is the same as that of the `Sender Template`
- a flow specification, `Flowspec`, composed of the reservation characteristics `Rspec` and the traffic specification `Tspec`; the latter is usually set to the `Sender Tspec`, except for the maximum policed unit which is updated according to the value of the path's maximum transmission unit as supplied by the `Adspec`
- an optional confirm object, `ResvConf`, supplied by the receiver to require the confirmation of the end-to-end resource reservation across the network, in case the reservation succeeds.

Upon receipt of the `Resv` message, the router interface passes the `Flowspec` to the traffic control module[1], which applies both admission control and policy control in order to decide whether the new flow can be accepted. If the request is unsuccessful, then the router must send a `ResvErr` message downstream. Otherwise, the reservation state is set up according to the effective `Flowspec` and `Filterspec`. These two specifications are affected by the *reservation style* selected for the communication session. The reservation styles help save the router resources by merging the processing of the data streams to be sent to receivers sharing the same communication session. Three reservation styles are currently available with RSVP:

1. The traffic control module is composed of the packet scheduler, the classifier, the admission control routine, and a policy database.

- The Fixed Filter (FF) style, which is characterized by a distinct reservation for each sender, and explicit sender selection by the receiver. In this situation, the effective `Flowspec`, at any given router interface, is the maximum of all FF reservation requests received about the sender of interest. Moreover, each `Filterspec` must correspond to one and only one sender.
- The Wildcard Filter (WF) style, which is characterized by a shared reservation among the senders and wildcard sender selection. No specific sender is selected by the receiver (no `Filterspec!`), and all the senders share the `Flowspec` specified by the receiver. The effective `Flowspec`, at any given router interface, is the maximum of all WF reservation requests received.
- The Shared Explicit (SE) style, which is characterized by shared reservation among senders explicitly selected by the receiver. With this style, the receiver requests a single `Flowspec` for a number of senders explicitly specified in the `Filterspec`.

The receivers may tear down their reservations by issuing the `ResvTear` message.

Weaknesses. RSVP has some weaknesses that considerably undermine its wide deployment. Here is a highlight list gathered from [15,21]:

- drawbacks of soft-state reservations, i.e., reservations with periodic refreshments: the whole paradigm is intolerant to faults (i.e., loss of refresh messages may entail the disruption of the session), bandwidth is wasted in carrying refresh messages. Moreover, the signaling messages and the data may follow different paths, since the routing protocol is independent of RSVP, it may well happen that, between two refresh messages, the shortest route between two points has changed. Since resource reservation between these points has been made along a different path, the next refresh messages will follow the new shortest path. This situation renders RSVP unable to always guarantee good network performance even if no error occurs and the reservation previously succeeded
- exponential growth of the reservation state table. Each router, along the paths between the sender and the receivers, maintains the state of each and every flow. This poses scalability issues; RSVP is not able to cope with a higher and higher number of simultaneous users
- reluctance of Internet Service Providers (ISP) to assure QoS across their domains. There is no coordination among them so far. As an ISP, I will not implement RSVP if all the other ISPs do not want to do so, since resource reservation is a waste of capacity if it is not coordinated all the way between the sender and the receiver.

2.5 Conclusion

The IntServ architecture is one of the first elaborate attempts to provide the Internet with a paradigm that considers the requirements of real-time services. The main strength of this architecture lies in its comprehensive, systematic, and studious approach: the services of interest are first identified, the commitments required from the network in order to support them are derived, and then a reference implementation framework is proposed. Commitments from the network that have been unambiguously identified so far are the guaranteed service and the controlled-load service.

The resource reservation protocol, RSVP, proposed by the IntServ group however presents a number of weaknesses that bring some complexity in its implementation. Several alternatives have been proposed in the last few years, many of which fall under the umbrella of differentiated services (presented next). There is one scheme, however, that is somehow in the line of RSVP (both schemes use the notion of flow).

It is a proposal by Pan and Schulzrinne [21], who suggest a reservation protocol based on the Real-time Transport Protocol (RTP) [23]. The proposal is called YESSIR (YEt another Sender Session Internet Reservation). The motivation behind it, came from the notice that most of the applications that need resource reservation today use RTP. It then becomes interesting to provide the network with resource reservation mechanisms that are based on this protocol. One important difference between YESSIR and RSVP is that the former is sender-oriented while the latter is (fundamentally) receiver-oriented. Pan and Schulzrinne argue that, for most services, the receiver will just accept the full quality provided by the sender, so having a receiver-oriented protocol makes less sense.

3. Differentiated Services Architectures: A State of the Art

In order to get round the weaknesses of the solutions proposed by the IntServ group, a new group, called Differentiated Services (DiffServ) group, suggested investigating another direction: instead of maintaining the state of each and every *flow*, why not look at discriminating the *packets* according to their precedence? This idea led to the concept of differentiated services which also has the advantage of being "easily" implementable even in existing networks. The IntServ architecture can be regarded as a differentiated services architecture built around the concept of flow as explained in Section 2. Nevertheless, the concept of differentiated services mostly refers to the packet-based scheme.

In this Section, we present a chronological review of the state of the art in the field of differentiated services. Although non-exhaustive, this review gathers the main trends in the area. In sections 3.1 through 3.6, we describe these main trends which are headed by:

- the Service Allocation Profile Scheme (SAPS) [10] (section 3.1)
- the QoS Services of the Cisco Internetwork Operating System (IOS) Software [7] (section 3.2)
- the Two-bit Differentiated Services Architecture (TDSA) [20] (section 3.3)
- the Scalable resource Reservation Protocol (SRP) [1,2] (section 3.4)
- the Simple Differential Services Model (SDSM) [12] (section 3.5)
- the Provider Architecture for Differentiated Services and Traffic Engineering (PASTE) [18] (section 3.6).

In section 3.7, we present an attempt to unify the existing differentiated services architectures. Finally, we compare the architectures with one another (section 3.8).

3.1 The Service Allocation Profile Scheme (SAPS)

Clark and Wroclawski suggest an approach that aims at allocating the bandwidth to the users in a controlled manner during periods of congestion [10]. Their proposal is called the Service Allocation Profile Scheme (SAPS). The core idea is to monitor the traffic generated by each user, and tag packets as being "in" or "out" of the service profile, i.e., the agreed-upon quality obtained by the customer from a provider. In the occurrence of congestion, the routers preferentially drop the traffic tagged as "out" of profile. "In" and "out" packets share the same queue, so nothing fundamental changes from the current situation. By not separating traffic into different flows or queues, the SAPS model becomes easier to implement than, say, RSVP. The network

commits to deliver an assured service to the “in” packets, provided they conform to the service profile negotiated.

The SAPS architecture. We summarize the SAPS model in Fig. 3. A policy meter, located at each traffic source, implements the tagging rules specified by the user and shapes the traffic according to the bandwidth negotiated in advance from the next ISP in the communication path. It determines the packets whose in-profile bit has to be set. At the opposite side of the link, a checking meter inspects the incoming traffic and marks packets as “out” of profile if this traffic exceeds the negotiated profile. In Fig. 3, it might happen that the aggregate traffic at the ISP domain 1 exceeds the bandwidth negotiated between this domain and the ISP domain 2. Then, a policy meter is needed in ISP domain 1, to shape the traffic that has to cross Domain 2. The two kinds of meter are examples of profile meters, broadly discussed in [9].

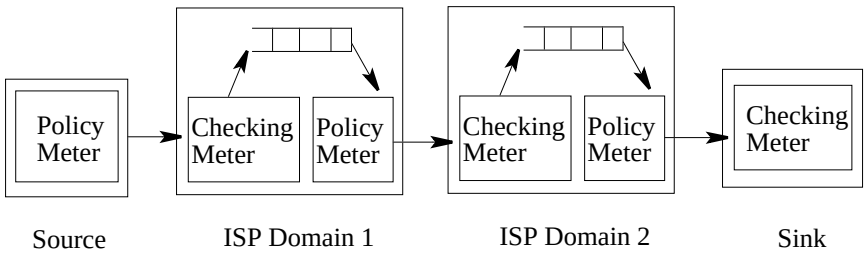

FIG. 3. Simple illustration of the SAPS architecture.

Clark and Wroclawski suggest the use of a single queue for both “in” and “out” traffic, instead of employing priority queues which may have the drawback of separating packets belonging to the same “flow” (e.g., when the same “flow” is composed of a base layer and an enhancement layer which are transported with different priorities). The result is out-of-order arrival of these packets at their destination. Such an impediment can be avoided when only one queue is used. In this situation, preferential packet treatment can only be achieved by using an appropriate dropping scheme in case of congestion. Clark and Wroclawski suggest the use of a variant of the Random Early Detection (RED) [14] which randomly drops packets to control congestion. The variant suggested is called RED with In and Out (RIO). RED is characterized by two thresholds and a drop probability distribution. When the queue size is below the lower threshold, no packet is dropped. When this size is in between the two thresholds, packets are dropped randomly according to a defined drop probability profile. Beyond the upper threshold, any incoming packet is dropped. The mechanism proposed with the SAPS (i.e., RIO) differentiates among the packets to be dropped. Hence, if absolute need be, the dropping of “in” packets is done as a function of the number of such packets in the queue (the thresholds depend only on the number of “in” packets in the queue). On the other hand, “out” packets are dropped on the basis of the overall queue size.

Clark and Wroclawski propose using one bit of the IP precedence field to indicate the packet’s service. This bit is called the In Profile Indicator (IPI). Set to 1, it denotes an “in” packet. However, Clark and Wroclawski do not specify which specific bit to use.

In conjunction with the indication given by this bit, Clark and Wroclawski suggest the use of the TOS field to discriminate among different levels of assurance (i.e., guaranteed service or statistical service).

Remarks and pending issues. Clark and Wroclawski posed the problem of providing differentiated services in a clear, understandable and pertinent way. They have not, however, addressed a number of important issues yet, most notably:

- the design of the policy meter, i.e, the way that the traffic policies should be implemented: how does the meter know which packets to mark “in” or “out”? Clark and Wroclawski proposed two alternatives: either (1) the user defines a level of quality which is coded within each packet using the IPI (but, how does the user know the level of quality he desires?), or (2) a management application runs in the background and reports to the edges of the network what the current level of congestion is; the edges could then modify the user service profile accordingly
- the design of an appropriate admission control routine: this will depend on the target services concerned with the differentiation
- inter-domain management: if two adjacent domains crossed by a flow implement different congestion management policies (e.g., RED and RIO), then the level of assurance requested by the user may never get fulfilled; such a situation is not considered by Clark and Wroclawski
- a signaling support to be used for negotiating, maintaining, and controlling the user communication session. Clark and Wroclawski suggest the use of a lighter-weight RSVP, which consists in making the reservation decision only at the network edges; the routers in the network backbone would simply forward the message without making any reservation decision. This implies using RSVP on explicit routes previously set up by the network manager, and reduces the state to be stored in the backbone routers
- the setting of the RIO parameters
- charging.

3.2 The Cisco IOS™ Software QoS Services

In 1997, Cisco introduced advanced QoS services into its IOS Software suite [7]. This was a result of the consideration of new critical network requirements essentially due to the massive increases in demand for Internet bandwidth, performance and flexibility. We present these requirements below. Then, we report on their fulfillment by the Cisco IOS Software both in the network backbone and at the network edges.

Critical Network Requirements. The major key technological and business requirements that Cisco considers for the design of the QoS enhancement to its routers are:

- *Services scalability*: an increasing number of services will be furnished by service providers using network capabilities. To this end, the network should embed a comprehensive set of features to be used by the service providers to implement their own resource allocation policies
- *Intelligent Congestion Control*: the network must actively seek to anticipate congestion, recover gracefully from congestion situations, and distinguish between temporary traffic bursts and long term traffic overload conditions. In the event of congestion, higher priority traffic must receive preferential treatment
- *Investment protection*: network providers should not be required to change their infrastructure fundamentally before new services could actually be introduced

- *Traffic classification and prioritization*: the network must sort out and map packets into traffic classes or service levels for appropriate handling
- *Granular, lightweight metering*: the network must make available highly detailed and accurate measurements for billing/accounting as well as further management and planning purposes
- *Policy and service flexibility*: service providers must be enabled to specify resource allocation policies at fine-grained levels, e.g., policies might be defined at the physical port, address and application level.

These critical requirements led to many design options implemented at the network edges and within the backbone. We present these options below.

Design at the Network Edge. We can summarize the influence of the above-mentioned requirements on the design of the network edge as follows: the service provider must be able:

- to specify the policies that define the traffic classes and service levels, as well as the resource allocation and control schemes to apply to each class or level
- to map packets to the traffic classes or service levels
- to collect detailed measurements about the resources consumed by the traffic and the network services invoked.

To fulfill these requirements, the Cisco IOS QoS Services equipped the service provider with:

- the use of the IP precedence field to specify the provider's traffic classes (up to 6 classes)
- Extended Access Control Lists (ACLs) to define network policies in terms of congestion handling and bandwidth allocation for each class
- the Committed Access Rate (CAR) to implement bandwidth commitments and guarantee that traffic sources and destinations fulfill their contract. CAR is part of the Extended ACLs and may be used to specify policies to apply in case of excess to the allocated bandwidth. CAR thresholds may be applied per access port, IP address or application. It uses token bucket filters to measure the traffic load and force the sources to comply with the allocated bandwidth. It allows several categories of service: (1) the firm CAR policy option is chosen to express that packets in excess to the allocated bandwidth must be discarded, (2) combined with the Premium option, the CAR feature indicates the "recoloring" of the exceeding traffic with either higher or lower precedence levels, (3) used with the Best-effort option, CAR "recolors" the exceeding traffic up to the burst threshold after which any excess is simply dropped, and (4) the Per-application CAR specifies different policies for different applications
- the NetFlow package to provide a satisfying level of control and management on each flow[1]; for each flow established, NetFlow instantiates a task that simultaneously performs the processing activities, switches packets and collects data on a connection-oriented basis. The data collected includes the source and destination

1. Even though the Cisco solution implements differentiated services, we can still talk about flows in the network edges, since the state of the connection requested by the user is kept at the edges, while the state is recorded on a per-link (between ISP domains) basis in the network. The concept of differentiated services essentially means that the notion of individual end-user flow does not exist in the network.

IP address, start and end of flow timestamps, packet and byte counts, next hop router address, input and output physical port interfaces, source and destination TCP/UDP port numbers, and Type-of-Service field.

Design of the Backbone. The backbone carries data from one edge to another. In order to fulfill the critical network requirements outlined above, the backbone must offer ultra-high throughput, capacity, and reliability, as well as policy administration and enforcement mechanisms. Two such mechanisms are considered: *congestion management* and *queuing*.

Two schemes are offered to *manage congestion*: RED and Weighted RED (WRED). The latter is a variant of the RED, taking into account the service level requested by each individual packet. As mentioned earlier, RED uses two thresholds and a drop probability distribution for all packets in transit on the backbone. The originality of the WRED lies in the definition of these parameters for each service level. Thus, packets with higher priority may have a dropping profile different from that of lower priority packets.

The *queuing* mechanism adopted by the Cisco solution is Weighted Fair Queuing (WFQ) based on the Class of Service (CoS) required by the packets. WFQ divides the link traffic into high priority and low priority flows based mainly on the information carried by the IP precedence field and the traffic volume. High priority flows receive immediate treatment, while low priority packets are interleaved and receive proportionate shares of the remaining bandwidth.

An overall picture of the Cisco solution is depicted in Fig. 4.

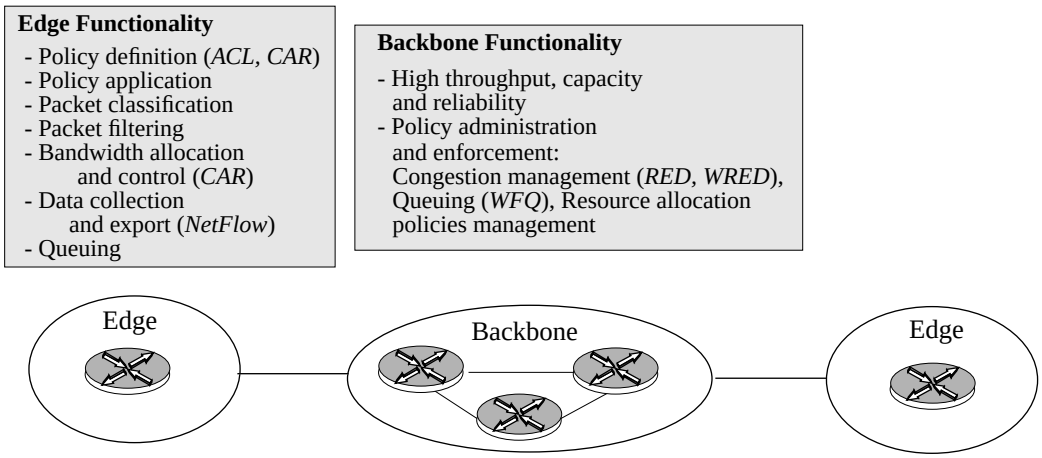

FIG. 4. Cisco's Architecture for Differentiated Services.

Remarks and Pending issues. The Cisco IOS Software QoS Services offer a wide range of mechanisms to deal with QoS enforcement in an "Intelligent Internet". While the mechanisms deployed within the backbone are based on per-packet service differentiation, the edges may use RSVP to perform QoS negotiation and control. This gives the heterogeneous picture that is likely to characterize the future of the Internet. Moreover, the range of options provided with the CAR feature may be

used to offer a better choice of network commitments than is currently considered by most differentiated services architectures.

The WFQ scheme used within the backbone implies that packets of the same flow must be of the same service level; otherwise, packets would arrive out of sequence at their destination. Nevertheless, the WFQ scheme enables the discrimination of packets on the basis of their sensitivity to the delay.

3.3 The Two-bit Differentiated Services Architecture (TDSA)

Nichols *et al.* propose a Two-bit Differentiated Services Architecture (TDSA) which currently incorporates three services: Premium, Assured, and Best-effort service [20]. The Premium service is provisioned according to its peak capacity profile. It denotes packets that are enqueued in a higher priority queue than the ordinary best-effort traffic. The Assured service follows "expected capacity" profiles which are statistically provisioned [9]; this service is the one considered by Clark and Wroclawski in [10]. It denotes packets that are treated preferentially according to the dropping probability applied to the best-effort queue. The assurance of the service comes from the expectation that the traffic is unlikely to be dropped as long as it stays within the negotiated capacity profile. The dropping scheme adopted is RIO.

The TDSA is an enhancement (with an additional mechanism) to the work done by Clark and Wroclawski in their SAPS architecture (section 3.1). Below, we present the way service differentiation, traffic marking and shaping, and profile metering are achieved within the TDSA design. Then, we present this architecture's basic functions (or primitives). Inter-domain bandwidth allocation is treated next. Finally, we provide some remarks and point out some pending issues.

Service Differentiation. Nichols *et al.* propose using two bits of the IP precedence field to indicate the packet priority (or service level). One bit, the A-bit, maps to the Assured service, while the second, the P-bit identifies the Premium service. These bits are called *service bits*.

Traffic marking and shaping. The traffic is marked (as of A or P type) and shaped using a token bucket mechanism. Tokens are generated at a rate that maps somehow to the packet rate negotiated. The token depth corresponds to the burst parameter negotiated. It is set to 1 or 2 (tokens) for the Premium service, which is expected to conform to the peak bandwidth allocated without any (or with very little) burst.

Upon receipt of a packet, the marker checks whether there is enough token in the bucket. If so, then the packet's either A- or P-bit is set to 1. Otherwise, the packet is not marked in the case of the Assured service. The marker that deals with the Premium service waits for a token to be emitted. Thereafter, the packets are forwarded to an output queue.

Profile Metering. The profile meter (actually a checking meter, Fig. 3) suggested by Nichols *et al.* is based on the token bucket scheme as in the marking process. When a packet arrives at the router, its IP precedence field is checked. If the P- (resp. A-) bit is set, then the profile meter checks whether a sufficient token is available. If so, then the packet is passed to the forwarding engine. Otherwise, it is dropped in the case of the Premium service, and unmarked (its A-bit is cleared) in the case of the Assured service. In the latter situation, the packet is considered as a best-effort packet.

Forwarding path primitives. In the light of the description above, Nichols *et al.* identified the following forwarding path primitives:

- `General Classifier`. Leaf or first-hop routers must perform a transport-level pattern matching based on a tuple of the packet header. Packets whose tuples match one of the configured flows, have the appropriate service bit set.
- `Bit-pattern classifier`. This primitive basically makes a decision based on the value of a particular bit in the IP header (e.g., profile meters check whether the service bits are set, and make decisions accordingly).
- `Bit setter`. The service bits need to be set or cleared at many places.
- Priority queues. Nichols *et al.* propose the use of two priority queues: one for the Premium service and the second for the Assured and Best-effort services.
- `Shaping token bucket`. This primitive shapes the traffic in order to make it comply with the negotiated profile.
- `Policing token bucket`. This primitive checks whether the incoming traffic conforms to the profile negotiated.

Inter-region traffic allocation: Bandwidth Brokers. Two main functions are still missing in the proposal by Nichols *et al.* as described until now: how to parcel out an ISP's marked traffic allocations and configure the leaf routers accordingly? And how to manage the messages that are sent across the boundaries of two adjacent ISP's domains?

Nichols *et al.* suggest that these two important tasks be taken by intelligent agents called Bandwidth Brokers (BB). A BB is associated with a particular trust region. A trust region can be thought of as a domain wherein the differentiated services traffic is fulfilled as long as it conforms to the resource allocation policies negotiated. A trust domain has a policy token bucket and a shaping token bucket implemented at its boundaries.

Each BB keeps a database that contains the policies to be applied to the trust region. It is the only entity authorized - as far as differentiated services are concerned - to configure the leaf routers such that they deliver a particular service to the packets. The BB interacts across the trust region with other BBs in order to provide customers with end-to-end QoS. All the requests for resource allocation are issued to the respective BBs. The yet-to-come protocol between peer BBs will play the role of an advance resource reservation protocol for differentiated services. A new real-time signaling is not needed across the trust regions for setting up a session between two end-systems.

We give an overview of the TDSA architecture in Fig. 5. The implementation of the profile meters using the primitives outlined above is also shown.

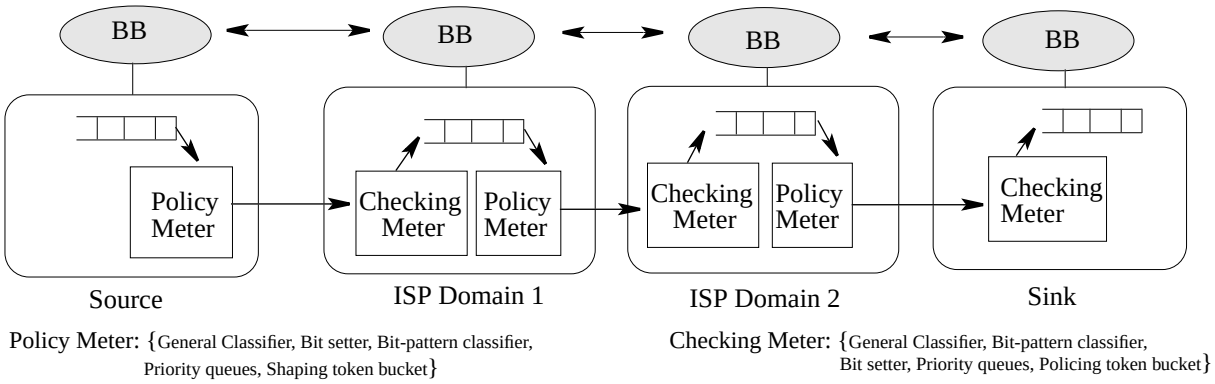

FIG. 5. An overview of the TDSA architecture.

Remarks and Pending Issues. The TDSA scheme describes most of the primitives needed in a differentiated services architecture. Moreover, it gives relevant insight on inter-domain interactions for setting up and maintaining the policies that govern the fulfillment of the agreements among the domain administrators. This fulfillment is achieved by the BBs. However, a number of issues are still pending, especially:

- the way policies (configuration information) are specified and passed to the routers
- the way policies are enforced
- the thorough implementation of the primitives outlined, as well as an accurate admission control routine
- the support for dynamic resource reservation: BBs can take the initiative of augmenting the capacity of a given link so that the requirements of a new flow can be met. In this case, BBs later inform their respective human managers
- inter-broker communications: the conditions, under which the communication among BBs is allowed, have to be defined by the service managers.

3.4 The Scalable resource Reservation Protocol (SRP)

SRP [1,2] is a reservation protocol which achieves scaling for a large number of flows by aggregation on each link in the network (Fig. 6). Users do not explicitly signal connection parameters. Instead, senders mark packets not covered by an existing reservation as REQUESTs. A router's decision to accept a REQUEST packet is based on an estimate from measurements of already reserved traffic. A receiver sends the number of successful REQUESTs as feedback to the sender, which then marks packets as RESERVED at the appropriate rate (or if the reservation failed, it may cease sending). The feedback consists of traffic specification (`trafSpec`) parameters.

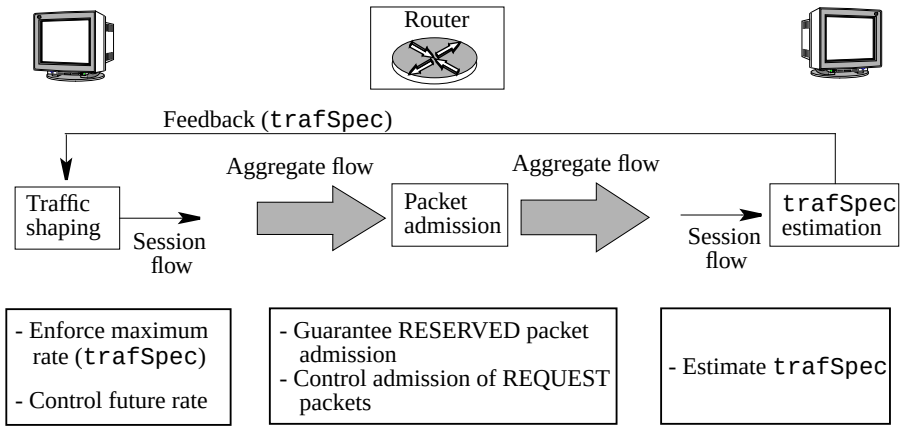

FIG. 6. A service architecture based on SRP [2].

In order to control abuse by sources of the aggregation nature of SRP, a policing element may be placed at each router. This element provides any abuser with a high probability of receiving worse service than if it were well-behaving, and thus proves a controller and a disuader.

Remarks and pending issues. Almesberger *et al.* [1,2], the authors of SRP, proposed a simple and complete (i.e., end-to-end) solution for differentiated services. Unlike the architectures mentioned above, the SRP-based model assumes that packets are marked by the applications. Therefore, existing applications should be reimplemented in order to take advantage of SRP. It might be more convenient to have some of the SRP mechanisms implemented at the leaf routers as suggested by Nichols *et al.* within the TDSA.

3.5 The Simple Differential Services Model (SDSM)

Ferguson proposes a Simple Differential Services Model (SDSM) [12] which is based on delay indication and drop preference. Delay requirements are indicated in IP packets via the TOS field specified in [3] (Table 1). Ferguson makes an intelligent interpretation of the existing semantics, and does not change anything to the existing IP paradigm. He suggests the Class Based Queuing (CBQ) as the queuing discipline to be used with his model. With the CBQ, a queue is allocated to each service level; packets are forwarded to one of the queues according to the delay indication expressed by their TOS field. The CBQ should be implemented in either the first-hop ingress router or each intermediate hop on the transit path. The latter alternative is more questionable, since the transit path may span many different administrative domains.

TABLE 1. Delay indication using the IP TOS field as specified in [3]

Bit Value	Existing Semantics	Delay Indication
1000	minimize delay	Highest delay sensitivity
0100	maximize throughput	.
0010	maximize reliability	.

TABLE 1. Delay indication using the IP TOS field as specified in [3]

Bit Value	Existing Semantics	Delay Indication
0001	minimize monetary cost	Lowest delay sensitivity
0000	normal service	No delay sensitivity

Ferguson proposes the use of the IP precedence field [16] to express the drop preference (Table 2). Again, he makes an intelligent interpretation of the IP paradigm. Obviously, network control information has precedence on any other data. Ferguson argues that the most effective method of mitigating congestion is to use RED. To implement the preferential packet servicing, Ferguson suggests that an "enhanced" RED be designed, along the lines of WRED or RIO.

TABLE 2. Drop preference expressed using the IP precedence field

Bit Value	Existing Semantics	Drop Preference Semantics
111	Network Control	Lowest
110	Internetwork Control	.
101	CRITIC/ECP	.
100	Flash Override	.
011	Flash	.
010	Immediate	.
001	Priority	.
000	Routine	Highest

Remarks and pending issues. Ferguson proposes an intelligent use of the IP Type-of-Service field rather than a thorough differentiated services architecture. Hence, all the issues related to the design of the mechanisms that will support these services are not addressed. Likewise, there is no way to consider services other than those already present in the existing IP precedence semantics.

3.6 The Provider Architecture for Differentiated Services and Traffic Engineering (PASTE)

Most of the architectures presented above addressed the mechanisms to be deployed at the boundaries of the ISP domain. What about the internal organization of this domain? Li and Rekhter [18] propose an architecture that answers this question. The architecture is constructed around the concept of *trunk*. A trunk carries traffic of a single traffic class that is aggregated into a single Label Switched Path (LSP). An LSP is a path made up of Label Switching Routers (LSR) whose particularity is to forward packets on the basis of a label, not according to the destination address of the packet. When entering the ISP domain, the packet is given a label on the basis of its header. As long as it is still in the same domain, the packet is routed, along the LSP, with respect to this label. Flows from different sources can therefore be aggregated under a single label. The main requirement upon this aggregation is that, the aggregate flow must be composed of flows that share a forwarding state and a single resource reservation within the domain. Li and Rekhter suggest the use of the Multi-Protocol Label Switching (MPLS) [22] as the mechanism for implementing flow aggregation.

The issue of traffic engineering and service differentiation is hence reduced to the handling of trunks (Fig. 7). A trunk is associated with a Class of Service (CoS), and

characterized by the direction of the traffic. Three basic CoS are considered: Best-effort, Priority, and Network control service. There might be further multiple levels in the same CoS, depending on the actual needs. Since the number of CoS is small, the scheme scales well with increasing user traffic. PASTE thus avoids the state explosion that might occur with solutions such as RSVP. In Fig. 7, we illustrate the provision of trunks by the LSRs in the ISP domain.

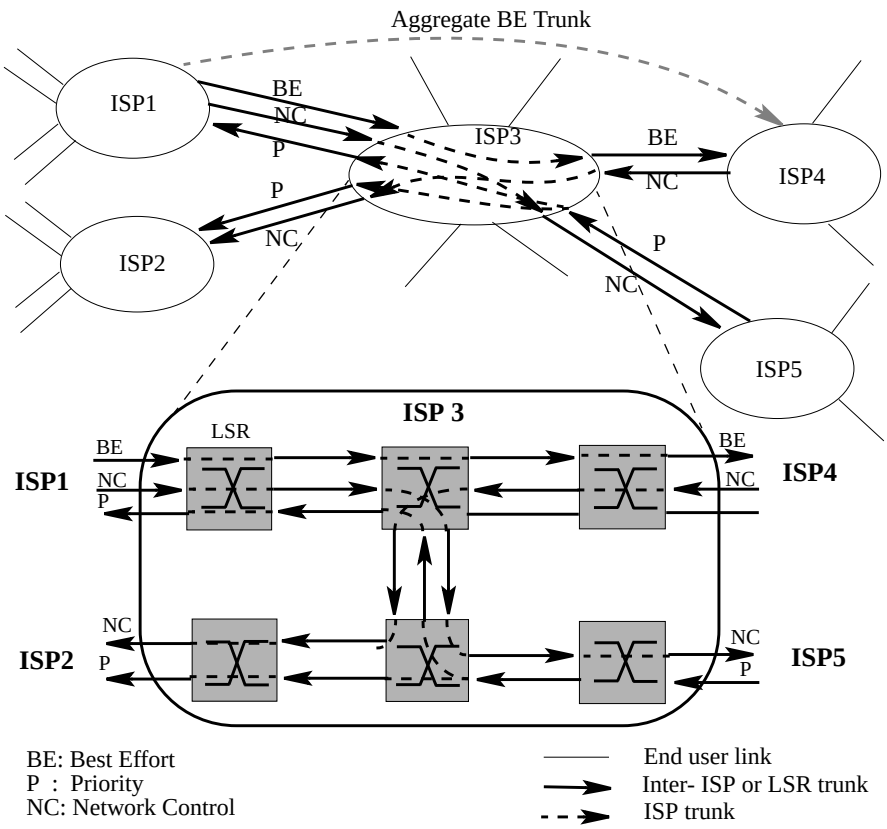

FIG. 7. An overview of the PASTE architecture.

Li and Rekhter elaborate on some of the rules that will govern the handling of the trunks. They suggest establishing and releasing trunks by means of RSVP. This protocol however is not employed by Li and Rekhter in the same manner as the IntServ group. There are three main differences between the two uses:

- PASTE uses RSVP to set up a collection of flows, not individual flows
- RSVP is employed in PASTE not only to make resource reservations, but also to install and keep state related to traffic forwarding, including label switching information
- destination-based routing, which might undermine correct operation of RSVP (cf. section 2.4), is no longer used. PASTE uses the Explicit route option of RSVP (this route being the trunk).

Trunks are established after agreements have been made among the customer and the ISPs. The ISP might merge trunks that share some common internal path. Li and Rekhter left the trunk merging policies for further study.

Remarks and Pending Issues. Li and Rekhter provide a thorough architecture for ISP domains, and clarify many issues of interest to the ISPs. They also provide a new useful application for multiprotocol label swicthing.

The mains contributions that we retain from the PASTE architecture are the concept of trunk, the use of MPLS, and the adaptation of RSVP for aggregate flows. The other mechanisms that need to be deployed in interior routers can be borrowed from other differentiated services architectures.

After the review of the main architectures for differentiated services, we present next a unifying framework (i.e., set of principles, definitions and methods to be used when designing an architecture) proposed by Clark.

3.7 A Unifying Framework?

Clark [8] proposes a framework in the goal of:

- identifying, and agreeing on, some mechanisms that will be implemented in the Internet; to be "standardized", these mechanisms must be general and implementable
- illustrating the way that the mechanisms can be used to provide a wide range of services
- defining how bits should be used to select the mechanisms.

Below, we define the main terms introduced above, and describe Clark's framework. Lastly, we establish the relation between this framework and the architectures presented above.

Definition 1. A mechanism is a component of a specific network element (e.g., router, profile meter).

Definition 2. A service is a specific use of the mechanisms. It might correspond to the user perception of the work achieved by a set of mechanisms.

Definition 3. Rules define limitations on the usage of the mechanisms to build services. They are enforced by profile meters.

Mechanisms are implemented at three important places:

- the end node that generates or consumes traffic
- the boundaries between networks: mechanisms here must verify the usage patterns, tag packets, shape flows, log usage, etc.
- the router: mechanisms here essentially anticipate, prevent, and handle congestion.

Clark outlines a number of drawbacks with using the header bits to select mechanisms. One is heterogeneity in the way services might be specified in different parts of the network. Imagine that the bits used for selecting mechanisms are not the same across domains! There is no standards that specify the way in which services should be built out of the mechanisms. To solve this problem, Clark proposed that either the meters in the interior of the network should be able to know what the overall desired service is, or a reservation setup or management tools should be called upon to provide this information.

The use of header bits serves two goals: mechanism selection and per-packet control over what the mechanism does. Two mechanisms are identified by Clark: priority

queuing and RIO-based dropping. Per-packet treatment is intended to check the packet priority and forwards the latter to the appropriate queue; it is also concerned with implementing the drop preference scheme. Clark suggests the use of a TOS value to select mechanisms, and one or more precedence fields to select the per-packet treatment.

Relation to the architectures reviewed. Most of the works described above fit into the global framework laid out by Clark in [8]. This framework considers the three places where mechanisms should be implemented: the end node, the network boundaries (which are called edges in Cisco parlance) and the router (which is located in the backbone as termed in Cisco parlance). The Cisco package implements a comprehensive set of mechanisms in both the edges and the backbone. These mechanisms have the same functionality as those in the SAPS and the TDSA. In this respect, the three architectures are very close to one another. However, they differ in the number of services offered. The CAR feature of the Cisco package enables a much higher number of services than the others. Moreover, the Cisco solution has already integrated policy specification and management, which is not the case with other solutions.

The PASTE architecture complements Clark's framework by addressing the coordination of the mechanisms deployed by the leaf and intermediate routers. The concept of trunk introduced by the PASTE architecture may be used by the bandwidth brokers (of the TDSA model) to enforce the policies set up by the ISP.

3.8 Joint Assessment of the Architectures

We summarize in Table 3 some of the main features of the architectures reviewed in the preceding sections. A coarse look at this table reveals that no row is empty, i.e., any of the issues is addressed by one architecture or another. We use the *italic* font (Table 3) to denote that the solution provided needs some more work.

Globally, the solutions are close in their approach to designing the backbone: congestion is managed using variants of RED, and priority queuing is employed to differentiate service levels (although the Assured service and the Best-effort service share the same queue in the TDSA). The main difference among the solutions reviewed lies in the inter-domain resource allocation which is *conceptually* solved by the TDSA and PASTE with the use, respectively of bandwidth brokers and trunk management policies.

Many differences appear however among the architectures in the design of the edges. Specifically, the Cisco solution offers more

- service categories,
- policy specification and management features, and
- data collection and export features

than the other architectures.

TABLE 3. Comparison of the differentiated services architectures.

		SAPS	Cisco QoS Services	TDSA	SRP[a]	SDSM	PASTE
Services considered		Best-effort and Assured	6 classes including many variants of the Premium and Assured services	Premium, Assured and Best-effort	Best-effort and controlled-load	Current IP precedence semantics	Best effort, Priority, Network Control
Service differentiation		1 bit from the IP precedence field	IP precedence field	2 bits from the IP precedence field	1 bit for service differentiation (RESERVED flag) and 1 bit for reservation request	3 bits from the IP precedence field	Current IP precedence semantics
Edge Functionality	Policy definition, enforcement and management		ACL, CAR	*To be developed with the bandwidth brokers.*	Done by the end-application on the basis of traffic estimation (derived from the feedback provided by the receiver)		
	Packet marking, shaping and policing	Token bucket	Token bucket	Token bucket (with different treatments for Premium and Assured service)			Token bucket with MPLS
	Bandwidth allocation and control	"Expected capacity"	CAR	"Expected capacity" and *Bandwidth broker*			Trunk establishment
	Data collection and export		NetFlow		Real-time Control Protocol (RTCP) [23]		
	Queuing	A single queue for both services considered	WFQ	Priority queuing: one queue for the Premium service, and another one for the rest	Priority queuing	CBQ	Priority queuing
Backbone Functionality	Congestion Management	RIO	RED, WRED	RIO			
	Queuing	Same as for the edge.					
	Inter-domain bandwith allocation			*Bandwidth Broker.*			*Trunk establishment: RSVP with MPLS*
	Policy specification and enforcement	Token bucket.	Token bucket, Policy-based routing [6]	Token bucket, Bandwidth Brokers	Token bucket, traffic estimator, measurement-based admission control		Token bucket, *trunk management*
Edge/Edge signaling		RSVP	RSVP + flow aggregation	*Bandwidth Broker* with RSVP	RTCP/SRP		RSVP + MPLS
Edge/Backbone signaling		IP	IP		SRP		

a. The Edge in the case of SRP is the end-system, while it is rather the first-hop router (or network) in the case of the other architectures.

We can also notice (from Table 3) that the work contributed by the IntServ group - especially RSVP - can be re-used for differentiated services. Therefore, the admission control mechanisms [17] being proposed for the network services offered by RSVP (i.e., controlled-load and guaranteed service) can be re-used as well. It is worth noting that RSVP as appearing in Table 3 is not employed as originally intended. RSVP is used here for aggregate flows, and not for individual ones.

4. Conclusion

In this article, we presented the main motivation behind packet-based service differentiation: per-flow differentiation, symbolized by the work under way within the IETF IntServ group, does not scale well as the number of simultaneous users grows. Then, we described some of the main differentiated services architectures, namely the SAPS (by Clark and Wroclawski), the Cisco IOS Software QoS Services, the TDSA (by Nichols *et al.*), SRP (by Almesberger *et al.*), the SDSM (by Ferguson) and PASTE (by Li and Rekhter).

We summarized some of the main features of these architectures in table form which reveals three interesting facts. First, there is some overlap among the architectures, i.e., they often propose similar mechanisms. Second, solutions to unsolved issues in one architecture might be found in other proposals. Lastly, some results from the IntServ group - specifically RSVP and admission control routines - might well fit the needs of differentiated services. These three observations may open the road to a unified differentiated services architecture. We think that this will be possible only after all the solutions at play have been implemented. Only then may we make a sufficiently thorough comparison. A unified architecture, at this point, may prove cumbersome and inconsistent.

There are however some important issues that are still pending, such as inter-domain bandwidth allocation, admission control, policy specification, enforcement and management, multicasting, and charging for the traffic. Policy specification, enforcement and management include the handling of misbehaving traffic (aggressive flows), which may use the network resources at the expense of well-behaving traffic. A careful look should be taken at defining policies about traffic violations, in order to protect well-behaving traffic. Charging is another important issue that will drive the user toward selecting the level of quality for which she can actually pay. Despite its importance, charging, in the context of differentiated services, has been ignored until now. The new services will need to be charged differently from the best-effort service, which is typically charged today with a flat rate, based on factors such as time of day and volume. The question remains as to whether differentiated services should only be charged by using the current parameters plus the service class, or whether additional parameters such as distance and destination [11], should be factored in.

References

[1] W. Almesberger, T. Ferrari and J. Y. Le Boudec, Scalable Resource Reservation for the Internet, Internet Draft <draft-almesberger-srp-00.txt>, Nov. 1997.

[2] W. Almesberger, T. Ferrari and J. Y. Le Boudec, SRP: A Scalable Resource Reservation Protocol for the Internet, Technical Report, March 1998, available from http://lrcwww.epfl.ch/srp/.

[3] P. Almquist, Type of Service in the Internet Protocol Suite, IETF RFC 1349, July 1992.

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

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

[6] Cisco Systems, Inc., Policy-based Routing, White Paper, 1996.

[7] Cisco Systems, Inc., Advanced QoS Services for the Intelligent Internet, White Paper, 1997.

[8] D. Clark, A Combined Approach to Differential Service in the Internet, Dec. 1997.

[9] D. Clark and W. Fang, Explicit Allocation of Best Effort Packet Delivery Service, Nov. 1997.

[10] D. Clark and J. Wroclawski, An Approach to Service Allocation in the Internet, Internet Draft <draft-clark-diff-svc-alloc-00.txt>, July 1997.

[11] H. J. Einsiedler and P. Hurley, Link Weighting: An Important Basis for Charging in the Internet, Technical Report SSC/1998/017, Communication Systems Division, EPFL, March 1998, http://lrcwww.epfl.ch/PS_files/WeightTTL.ps.

[12] P. Ferguson, Simple Differential Services: IP TOS and Precedence, Delay Indication, and Drop Preference, Internet Draft <draft-ferguson-delay-drop-00.txt>, 7 Nov. 1997.

[13] S. Floyd and V. Jacobson, Link-sharing and resource management models for Packet Networks, *IEEE/ACM Transactions on Networking*, 3(4), pp. 365-386, Aug. 1995.

[14] S. Floyd and V. Jacobson, Random Early Detection Gateways for Congestion Avoidance, *IEEE/ACM Transactions on Networking*, Aug. 1993.

[15] D. Hutchison, R. El-Marakhy and L. Mathy, A Critique of Modern Internet Protocols: The Issue of Support for Multimedia, ECMAST ‘97, May 1997, Milano, Italy.

[16] IETF, Internet Protocol, RFC 0791, Sept. 1981.

[17] S. Jamin, C. Lin and L. Breslau, A Measurement Based Admission Control Algorithm for Controlled-Load Service with a Reference Implementation Framework, Internet Draft <draft-ietf-intserv-control-flow-01.txt>, Oct. 1997.

[18] T. Li and Y. Rekhter, Provider Architecture for Differential Services and Traffic Engineering (PASTE), Internet Draft, <draft-li-paste-00.txt>, Jan. 1998.

[19] M.R. Macedonia, D.P. Brutzman, MBone Provides Audio and Video Across the Internet, *IEEE Computer*, April 1994.

[20] K. Nichols, V. Jacobson and L. Zhang, A Two-bit Differentiated Services Architecture for the Internet, Internet Draft <draft-nichols-diff-svc-arch-00.txt>, Nov. 1997.

[21] P. Pan and H. Schulzrinne, YESSIR: A Simple Reservation Mechanism for the Internet, IBM Research Report, RC 20967, T. J. Watson Research Center, Sept. 1997.

[22] E. C. Rosen, A. Viswanathan and R. Callon, A Proposed Architecture for MPLS, Internet Draft <draft-ietf-mpls-arch-00.txt>, Aug. 1997.

[23] H. Schulzrinne, S. Casner, R. Frederik and Van Jacobson, RTP: a transport protocol for real-time applications, IETF RFC 1889, Jan. 1996.

[24] S. Shenker, C. Partridge and R. Guérin, Specification of guaranteed quality of service, IETF RFC 2212, Sept. 1997.

[25] P. P. White, RSVP and Integrated Services in the Internet: A Tutorial, IEEE Comm. Mag. 35(5), May 1997, pp. 100-6.

[26] J. Wroclawski, The Use of RSVP with IETF Integrated Services, IETF Standards Track RFC 2210, Sept. 1997.

[27] J. Wroclawski, Specification of the Controlled-Load Network Element Service, IETF RFC 2211, Sept. 1997.

[28] L. Zhang, S. Deering, D. Estrin, S. Shenker and D. Zappala, RSVP: A New Resource ReSerVation Protocol, *IEEE Network*, 1993.