The Role of Signaling in Quality of Service Enabled Networks

K. K. Ramakrishnan, Gisli Hjalmtysson and Jacobus E. Van der Merwe
Networking and Distributed Systems Research
AT&T Labs - Research
Florham Park, NJ 07932

Abstract

In this paper we identify the common building blocks that enable some networks to provide better than best-effort transfer guarantees to the traffic they carry. We consider the role signaling plays in such a network and argue in favor of:

- pinned routes, with a highly efficient pinning process, to improve network stability and to ease the task of maintaining Quality of Service (QoS) guarantees in the face of changing network characteristics, including failures.
- the use of broad QoS classes to determine the path that a particular flow should follow through the network.
- providing the flexibility of specifying the detailed QoS for the flow, if needed, at any arbitrary time during the life of the flow.

We conclude that a flexible signaling architecture is an essential enabling component of any QoS aware network. We present an overview of the design and implementation of UNITE, as an example of a signaling architecture that embodies these attributes. More generally, we consider the relationship between QoS related signaling and other protocols and mechanisms that may form part of a an overall QoS enabled network and service infrastructure.

1 Introduction

Data communication networks have seen tremendous growth and are becoming ubiquitous. As a part of this, there is a great desire to support all forms of real-time and interactive communication on a common infrastructure. Significant effort has been expended on adaptive applications that can dynamically adjust to the changing conditions in a best-effort network. There appears to be growing consensus that there is a need for explicit support for Quality of Service (QoS) in the networking technology used, particularly at the network layer.

A number of networking technologies have emerged that are capable of providing a variety of different service guarantees to the traffic they carry. The evolution of each of these technologies to support QoS has been different. Networks based on the internet protocol (IP) suite start by having the minimum support at the network layer for QoS, and to then evolve by adding functionality for QoS support. The premise for asynchronous transfer mode (ATM) networks has been to design in the flexibility for QoS support, so that it is capable of supporting a wide range of future applications.

Thus, the different networking technologies differ in the exact nature of the service guarantees they provide, and indeed in the direction to achieve QoS enabled networks. However, we recognize that in all of this evolution, a common set of components are needed to achieve QoS and the associated service guarantees in a network. Some of the important functions include:

- Forwarding: In the data path, there is the need for differentiated treatment of the different groupings of traffic. These groupings may vary from broad traffic classes to individual flows between applications.
- Routing: QoS support in routing involves two sub-components:
  - Route dissemination that may be based on link capacities and the actual load on the links in the network.
  - A decision process that determines which subset of links are suitable to be part of a route that has adequate residual capacity.
- Signaling: Signaling is involved in choosing a path that is capable of providing the necessary guarantees; setting up a classifier in the intermediate systems to treat different classes of traffic in distinct ways (from the perspective of scheduling and buffer management); and finally setting up boundaries for resource usage across the classes of traffic, so that limits are placed on how much resources each class may use. As part of signaling, some of the sub-components include:
  - Admission Control. When an individual flow or an aggregation of flows requests service, a set of algorithms are executed to determine whether or not the flow can be accommodated. This is the role of admission control.
  - QoS negotiation. This involves further customization of the data path beyond the coarse grained definition of a traffic class. For example, a connection that supports a voice conversation may require a reservation of a specific bandwidth end-to-end.

One distinction between the approach IP has taken to QoS support (in particular in the Differentiated Services approaches) and the approach of ATM involves the need for methods to further enhance and customize the data path for the treatment of a flow. In ATM, this is embodied in the process of QoS negotiation. The question that is currently in front of the industry is whether QoS negotiation is needed for individual (or aggregations of) flows or can we live with differentiation between classes?

We argue in favor of a solution that is rooted in differentiated treatment based on coarse grained traffic classes, yet is flexible enough to allow QoS negotiation for individual flows when that is required.

The use of QoS routing protocols does not imply that detailed QoS specifications are used (or required) to route a flow requiring QoS through the network. Instead, a coarse grained QoS class, possibly signifying the dominant QoS constraint of the application, could be used to determine the best route for a particular flow. For example, a telephony application might be mainly concerned with latency, while the dominant requirement for the delivery of a high quality video stream might be bandwidth. If required, the resources used by a route that is pinned based on this broad QoS class can subsequently be refined. For example, detailed resource requirements for a flow might need to be specified in a resource restricted access part of the network, while in the backbone the broad QoS class may suffice. However, in both the access and the backbone, the initial route chosen can be based on the QoS class.

The two functions previously identified for routing, namely information dissemination and decision making, is present in conventional best effort networks, such as current IP networks. There are, however, some critical differences in the way these functions are performed in a QoS enabled network. In a best-effort network the routing information distributed is limited to topology information, while in a QoS enabled network it would likely also contain information regarding the current utilization of the network. In most networks that primarily support best-effort traffic, topology information is fairly static, and changes in the topology are often associated with failure events such as node or link failure.

In contrast, in a QoS enabled network, the utilization might change much more dynamically, but such information might be "less critical" than in the case of pure topology change information. In QoS enabled networks, the routing decision making process should naturally be influenced by the utilization information distributed. This calls for an enhancement of the decision process found in best-effort networks. The best route to a specific destination for a specific QoS might change without any topological change. In order to prevent network instability because of traffic continually switching between two alternative routes, it is crucial that a newly advertised "better" route only be used for new traffic destined for that destination, while existing traffic continue using the earlier best route. This implies the need for connections or some form of route pinning to be used in QoS enabled networks. (This appears to be the underlying assumption in QoS routing work although it is not always stated explicitly [1, 2].)

When routes for flows are not "pinned", assurances of QoS are more difficult to provide. When one path for a flow changes to another, due to changes in the load, not only do we need to ensure that the new path can provide the QoS needed for this flow, but the flows that already exist on that path also see the impact of the change in the route. Without route pinning, advanced planning and provisioning of paths is needed to support the potential of flows using alternative paths. Concomitant with pinning routes, the network needs to support admission control functions that ensure that the path is not over-committed. Maintaining the same route, unless there is a compelling reason to modify it, minimizes the frequency of changes in the path of flows, and helps in the provisioning and admission control functions.

We recognize the need to have the capability to recover from link failures. Pinned routes will also lessen the impact of topology change on flows that are not directly influenced by the topology change. Exercising the admission control function when a flow is re-routed on a topology change allows us to continue to maintain the QoS of those flows that should, in principle, not be impacted by the topology change, while allowing re-routing and recovery from failures for the impacted flows in a graceful manner.

We believe an explicit signaling protocol used in conjunction with a QoS-aware routing protocol is the appropriate means for both determining and "pinning" the route for a flow.

1.1 Signaling Functionality

In order to forward a stream or flow of packets through a QoS enabled network a number of functions have to be performed:

- Determine an appropriate path on which to forward the flow.
- Establish mechanisms in the nodes along the path, such as scheduling and queueing, so that packets belonging to this flow will receive appropriate treatment.
- Establish appropriate state in the network to identify packets belonging to this flow, so that only such packets will be subject to differentiated treatment.

When we consider the conventional ATM call establishment [2, 3], using source routing, all of the functions listed above are done together - at the time when the call is established through signaling. With hop-by-hop routing and forwarding in IP, the functions of path selection and the choice of forwarding mechanisms are combined into one, all operating on the time-scale of forwarding a packet.

We believe that a QoS enabled network will require a more flexible approach than what is offered by either of the above (conventional) extremes. Such functionality is provided by the UNITE signaling protocol. The details of UNITE are placed in the context of a QoS-capable framework, which is the well-understood connection-oriented ATM paradigm. In Section 2, we use UNITE as an example to better explain and demonstrate the feasibility of our ideas. However, we believe that there is a broad applicability of these principles in developing any QoS-enabled communication network.

The way UNITE chooses the path is closer to the way IP chooses a path for a flow - based on hop-by-hop routing. However, the way UNITE uses the path is closer to the way ATM uses a path - in a connection-oriented manner. Further, UNITE avoids loops when determining the path. This allows us to take advantage of the nice scaling properties of hop-by-hop routing, while achieving the requirement for loop-free pinned routes.

UNITE offers additional flexibility by allowing applications to communicate requirements for scheduling and queueing at arbitrary times. With ATM, this is typically done only at the call-establishment time. We believe that the freedom offered by the approach taken by UNITE is, in fact, critical for QoS aware applications. It is often the case that the detailed QoS requirements are not needed or are unknown at the time the path is chosen. It may also be desirable for applications to change the treatment of the flow during the lifetime of the flow. For example, an application such as packet telephony requires an end-to-end path between the communicating end-systems for initial application-level negotiation before the QoS requirements are known. It is only after this point that the end-systems (whether source or destination) can request appropriate differentiated treatment from the network. Restricting the signaling protocol to the detailed QoS negotiation only at the beginning of the flow is likely to be inefficient because the end-points may have to be overly conservative in stating their bandwidth requirements.

In the next section, we present the UNITE protocol in more detail. Much of QoS research, and the details of the UNITE protocol in particular, is grounded in a connection oriented networking environment such as ATM. In recognition of a broad industry move to connectionless IP networks, the final section of this paper attempts to look at QoS issues for this different context. In particular the evolving role of RSVP in fulfilling many diverse signaling needs is identified. We also consider the role and interaction between QoS enabled networks and services (and related protocols) that might make use of such networks. The paper ends with a conclusion.

2 UNITE

UNITE is an example of a signaling protocol that achieves the objectives stated above within the context of ATM as the QoS enabled network. (More in depth coverage of the protocol details can be found in [4, 5].) The fundamental philosophy of UNITE is the separation of connectivity from QoS control. This eliminates the round-trip connection setup delay, before initiating data transmission. Using a single cell with proper encoding, we avoid the overhead of reassembly and segmentation on the signaling channel, and enable hardware implementation of the signaling protocol. Performing QoS negotiation in-band allows switches in the path to process QoS-requests in parallel, facilitates connection specific control policies, supports both sender and receiver initiated QoS, and allows for uniform treatment of unicast and multicast connections.

2.1 Unicast call setup

The principal idea behind UNITE is a complete separation of connectivity from quality-of-service, or more generally, service attributes. Each message involved with connectivity establishment is reduced to a single ATM cell, with fixed field sizes and positions, avoiding the overhead of reassembly and segmentation on the signaling channel, as well as type-length-value (TLV) encoding of the signaling messages, allowing for hardware processing. UNITE eliminates the round-trip delay which is required in conventional ATM signaling before data transmission can be initiated. Exploiting per-virtual connection (VC) queueing, data can be forwarded immediately after a one-hop exchange, rather than suffering a full round-trip latency. For switches without per-VC queuing support, or if desired to perform the setup processing in software, an additional message is used. All QoS and service attribute messages in UNITE are sent in-band. This allows us to build on the extensive work done for QoS elsewhere both in ATM

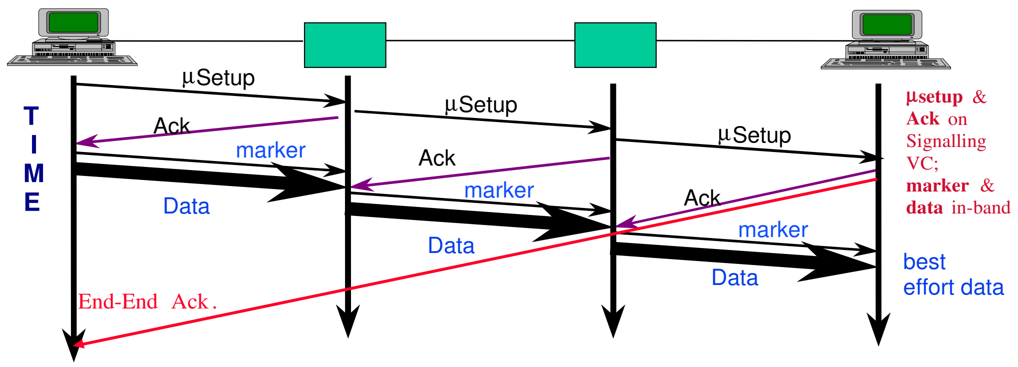

Figure 1: UNITE Connection Setup

and IP networks. Because the QoS messages are sent on the established VC, we can exploit parallelism in QoS (attribute) processing, improving throughput and latency for QoS establishment. UNITE allows for uniform treatment of unicast and multicast connections. In part due to its simplicity, UNITE supports both source and destination initiated QoS, supports multipoint-to-multipoint connections and recognizes the possible need for variable QoS to different participants (variegated multicast trees).

To establish connectivity, the connection originator (the calling station) issues a “micro-setup”, a single cell, on the signaling channel. The micro-setup includes all the information necessary to establish a best-effort connection to the remote destination (called station), most notably the destination address, a QoS class, and a globally unique flow identifier. A switch receiving a micro-setup determines the route based on the destination address and the QoS class and forwards the micro-setup. After allocating a VC and forwarding the micro-setup, the switch returns a single cell acknowledgment (“micro-ack”). In the case of downstream VC allocation, the micro-ack will contain the VC allocated by the downstream switch. On receiving the micro-ack, the upstream node opens the channel by transmitting a single cell “marker” on the established VC (in-band). This marker serves as the 3rd step of a three-way handshake, and opens the channel for use on a best-effort basis. It also acknowledges the micro-ACK message and demarcates the onset of the new data flow. The marker includes the flow identifier and the source address. The marker is the first cell sent in-band by the upstream node. Subsequent cells are valid data cells of the new flow. Virtual circuits are bi-directional. The latency to establish a best-effort flow is therefore only a single hop round-trip propagation delay, plus nodal processing.

To ensure that no persistent loops form, UNITE uses a combination of the flow identifier and an end-to-end acknowledgement. When the destination receives the micro-setup, it sends a reply message on the established VC to the source. This message is repeated periodically to reassert end-to-end connectivity and facilitate state cleanup. The first keep-alive indicates that a loop-free path has been established. The source will only release the connection after receiving an end-to-end acknowledgement, or after giving up on waiting for it (timeout). The above description applies for switches that support per-VC queueing. In addition, cells immediately following (too close to) the marker may be lost unless the marker is processed online in the (hardware) data-path. To cope with these cases an additional message, marker acknowledgement, is needed to confirm (upstream) the processing of the marker and thereby the opening of the channel. This extra message, which introduces an additional hop round-trip, is sufficient to accommodate software implementations and FIFO switches. The UNITE QoS class (1 byte) allows for coarse grained class based service differentiation and is a hint to improve the likelihood of success for subsequent QoS requests. We envisage the need to identify a small number of classes as being sufficient, for example classifying into: i) best effort, ii) delay sensitive, and iii) high bandwidth. The UNITE QoS class can also be used to encode the Differentiated Services field (DS field) being defined within the Diffserv working group of the Internet Engineering Task Force (IETF). UNITE uses hop-by-hop routing of the micro-setup, in contrast to the traditional ATM source-routing.

2.2 Soft state and garbage collection

The destination periodically sends a end-to-end state message in-band on the established VC. This serves two purposes. The first end-to-end message confirms that a loop free connection has been successfully established. Subsequent refreshes assert the liveliness of the connection. Each refresh message is marked as an operations and maintenance (OAM) cell that may is also delivered to every switch in the path. The messages contain the flow identifier and an expiration time (a function of the frequency with which the end-to-end refresh message is sent). A switch that does not receive a refresh message for some time (based on the expiration time) should release the connection in both directions and follow normal release procedures to recover local resources.

The end-to-end state refresh provides two major benefits. The first benefit is that the network may clear up state to recover from failures. State is cleaned up simply by a local decision at a switch in the path. This allows us to have a simple means for handling exception cases such as failures, implementation errors and lost protocol messages (see below on release). A second benefit is that the initial end-to-end confirm message avoids the possibility of an infinitely looping connection request. Such looping would be possible, for example during a routing transient, if and only if the originating station sends a release before an end-to-end connectivity is established. In this case, the looping micro-setup may not be detected if the release clears the flow state before the micro-setup completes the loop. By waiting for an end-to-end confirmation before issuing a release this exception case is avoided. The overhead of the state refresh is also negligible, because an end-system that transmits the end-to-end “keep alive” may control the frequency of the transmission possibly through some form of an exponential backoff (with a maximum interval), based upon knowledge of the behavior of the channel. The specification of the expiration time enables the end-systems to communicate the appropriate interval for switches to use for timing out state.

2.3 Restoration

One of the attractive features of IP is its ability to recover from node and link failures. This is simply performed by the routing system providing nodes the connectivity information, and re-routing packets around failures. Currently, the state of the art in ATM signaling allows for recovery during the process of setting up a connection. The means for recovering from a failure to set up an end-to-end virtual circuit is to crankback and attempt to set up a VC from an upstream node. It would be preferable if a connection-oriented network could also recover from failures, without impacting the end-systems. The multiprotocol label switching (MPLS) framework allows for restoration by pre-provisioning a label switched path to accommodate failures. This allows very rapid switch over to a backup path. However, the difficulty with the pre-provisioned approach is that resources may need to be committed on the back up path, even when it is unused. We believe that the separation between connectivity and resource reservation in UNITE can be very useful to manage restoration of paths, especially when this restoration has to be performed dynamically. While many details have to be worked out, our initial approach to UNITE recovering from link failures is to have a switch that identifies the failure to regenerate a micro-setup towards the destination. A switch already retains the information required for generating this micro-setup once the end-to-end connection has been setup. The primary identifier of interest is the

Flow-ID. When the micro-setup reaches a switch that has context associated with this Flow-ID, the micro-setup is not forwarded any further, and thus the connection is restored around a failure. As we described earlier, the Flow-ID is used at a switch to recognize loops. To enable the restoration around a failure, it is important to detect the switch generated micro-setup as being distinct from the original micro-setup using a flag.

There are several issues of concern that we do not address here in detail due to limited space. For example we need to ensure that the cells that were still in flight beyond the failed link do not arrive at the downstream switch and get re-ordered. This requires policy at the downstream switch to ensure that cells arriving on the old input port are not forwarded. We realize that some data may inevitably be lost, and believe that it is a function of a higher layer protocol to recover lost data. Quality of service will not be assured during the transition to the new path. To reserve resources on the new path, it may be desirable for the end-systems to refresh their reservation requests to enable the network to re-establish the QoS over the new segment of the connection that was established. This is similar to the approach used by resource reservation setup protocol (RSVP) in using a refresh mechanism for recovery from failures. However, this refresh mechanism does not need to be tied strictly to the soft-state approach where reservations are timed out on an existing path. We believe that this approach for restoration holds promise. By exploiting the separation between connectivity and resource reservation, we can route around failures effectively, without committing resources on alternate paths unduly for the purposes of restoration.

2.4 Call setup for multicast

UNITE incorporates the functionality of having multipoint-to-multipoint communication as an integral part of the signaling architecture. The simpler cases of point-to-multipoint multicast calls are simple sub-cases of this overall multicast architecture. The simple difference between a unicast call and a multicast call is that the micro-setup issued indicates that the call is potentially a multicast call. For the purposes of this discussion we assume that the underlying switch forwarding mechanism can manage issues such as cell interleaving. Therefore, we describe procedures that are applicable for core-initiated joins (for core based trees), which are similar for source-initiated join for a source-based tree. We then describe leaf-initiated joins for other participants that join subsequent to the call being setup.

2.4.1 Core/sourc

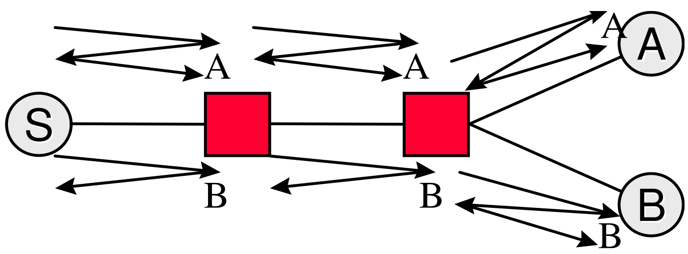

Figure 2: Core Initiated Join. Observe, that the Marker on the last hop to B is generated by the controller at the branch point.

Core initiated joins (or source initiated joins) are relevant when the set of participants is known initially. The core issues a micro-setup knowing the address of each individual participant. Since there is no way to package more than one address in the micro-setup, an individual micro-setup is issued for each of the participants. We think this is not as important, because: (a) the micro-setup is relatively cheap and (b) the number of participants that subsequently join using the leaf-initiated joins may dominate. The first micro-setup issued to a participant includes a label (in the Type field) to indicate that it is a multicast-capable call setup. The rest of the micro-setup is similar to that described for a unicast call. The Flow-id is determined by the originator (the core or sender). The Flow-id acts as a call-reference identifier for the multicast call. The micro-setup issued for joining subsequent participants uses the same Flow-id, again labeled as a multicast. The micro-ACK that comes back from the downstream hop returns a VC id as with unicast calls. The Marker transmitted by the core (or source) is sent in-band, on the VC id returned in the ACK. The Flow-id used in the micro-setup is retained at the switch, as a means of identifying the multicast call. During joins, the switch sending the micro-setup maintains state in the switch, which includes the Flow-id and the destination address to which the setup was issued (the new leaf). This way, ACKs that return for the individual setups issued may be matched up by the switch, for purposes of managing their retransmission.

The initiator of the micro-setup (core or source) sends the Marker when it receives the first micro-ACK. Upon receiving subsequent micro-ACKs, the source/core knows that the VC is already open (operational) and therefore, doesn't generate a further Marker. At a new branch point on the multicast tree, however, a Marker is required to the new destination: this is because that branch of the tree needs to be flushed of any old data that is in existence for that VC identifier. The controller is responsible for generating and sending this in-band Marker. Subsequently, data may be forwarded on that VC id, as a result of a proper 3-way handshake. Figure 2 illustrates this scenario.

2.4.2 Leaf initiated joins

The mechanisms for Leaf Initiated Joins (LIJ) are similar to those suggested in the conventional ATM Forum UNI 4.0 [3]. However, instead of having a separate LIJ and Add-Party mechanism, UNITE uses the same mechanisms of the micro-setup for performing a LIJ. Consider Figure 3, where two participants A and B wish to join the multicast tree, that currently ends at Switch 4. The LIJ is a micro-setup (the Type indicator indicates that this is a LIJ for a multicast call) from one of the participants, that is directed towards the core/source, using the address corresponding to the core/source. The Flow ID used in the micro-setup is the multicast call reference identifier, and is stored at the switches as the micro-setup is forwarded upstream towards the core. We assume that the underlying call routing mechanisms direct the micro-setup towards the source/core in accordance with the appropriate criterion (e.g., shortest-path or least cost). When a LIJ arrives at a switch from another participant, such as B, the Flow ID is recognized as already existing at the switch, and the forwarding of B's micro-setup is suppressed. This may be done only if the core does not wish to be notified of the address of an individual leaf joining. Note that this happens even though the LIJ of the first participant added on this branch has not yet reached the tree at Switch 4. When the micro-setup from A is issued, the 3-way handshake results in the Marker being forwarded by the switches upstream. This effectively opens up the VC from the node A up to the branching point, at Switch 4. Along with the suppression of the micro-setups, subsequent Markers are also suppressed at the switches.

2.5 Coexisting with UNI

There are at least two ways for UNITE can coexist with UNI signaling, one by deploying UNITE parallel to UNI, the other by translating the signaling messages at the UNITE to UNI boundaries. In the former case, UNITE will use a different default VC for signaling messages. This leaves the UNI signaling is unaware and unaffected by the presence of UNITE. In addition the switch must dispatch the QoS and attribute messages sent on UNITE VC's to the UNITE control processor. This is depicted

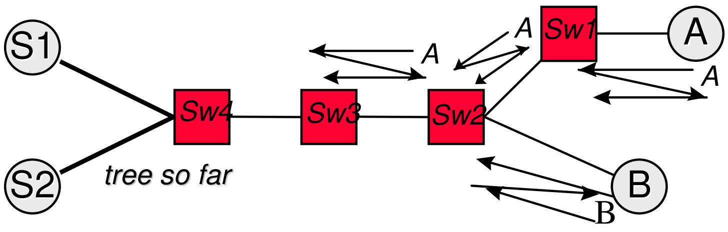

Figure 3: Leaf Initiated Join. A's LIJ is progressed to switch four. B's LIJ is suppressed at switch two.

in Figure 4. The latter case is achieved using a conventional signaling gateway. A switch going from the UNITE to UNI domain maps the micro-setup to a UBR connection request. In the reverse direction, when a UNI setup is received, that is appropriate for a UBR connection, the border switch can return the connect back to the UNI switch while forwarding a micro-setup into the UNITE cloud. For other connections that require assurances, the border switch creates both the micro-setup as well as a QoS request. The connect is returned only when the commit for the QoS request is received back from the UNITE end-system.

2.6 QoS

UNITE uses in-band messages for QoS establishment. This way UNITE can exploit existing QoS signaling, including an ATM UNI type of QoS service model and processing, an RSVP type of QoS model, and a coarse grained QoS mechanism as defined by differentiated services. QoS messages are sent on the established VC exploiting parallelism to improve the throughput and latency for QoS establishment. In part due to its simplicity, UNITE supports both source and destination initiated QoS, supports multipoint-to-multipoint connections and recognizes the possible need for variable QoS to different participants [6], (variegated multicast trees). Performing QoS negotiation in-band allows switches in the path to process QoS-requests in parallel, facilitates connection-specific control policies, supports both sender and receiver initiated QoS, and allows for uniform treatment of unicast and multicast connections. However, UNITE's framework provides a more flexible and efficient QoS management in the following dimensions:

- UNITE QoS requests may be initiated by the source or the destination of the original best effort connection setup. In the more general case of multicast connections, QoS requests may be

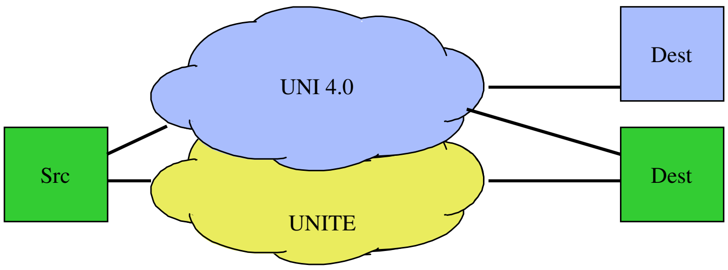

Figure 4: Parallel control planes

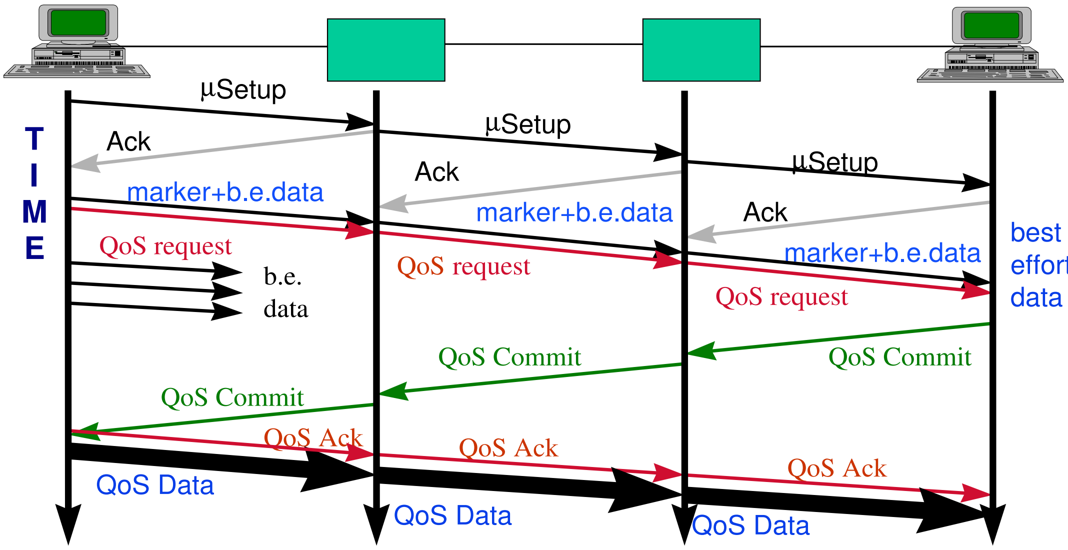

Figure 5: Establishing QoS in UNITE

source/core initiated or leaf initiated.

- UNITE QoS in-band signaling allows QoS renegotiation originating from any of the connection end points.
- UNITE QoS in-band signaling enables potentially different QoS negotiation modalities and implementations taking advantage of parallelism in the processing of the QoS setup across multiple switches in the end-to-end path.

For those flows that require a detailed QoS negotiation, we use the process of QoS setup described in Figure 5. The QoS request may immediately follow the marker, as shown in Figure 5, or may be submitted after the call is established. The receiver, after processing the request, sends a QoS Commit, that commits the reservation. To adjust over-committed reservations, and to confirm the QoS reservation to the receiver, the originator sends a QoS Ack. The delay until a QoS flow begins on the forward path is an end-to-end round-trip plus the processing at the destination. On the reverse path, a confirmed QoS flow begins one round-trip after the QoS Commit is issued from the destination. For compatibility with existing ATM, we anticipate that the QoS request, Commit and Ack, would be encoded as in the UNI connection setup and connect messages, as far as the QoS information is concerned. For our purposes in this section, we treat the end-system that initiates the QoS setup request as the QoS source. The end-system that responds to the QoS setup request at the other end is the QoS destination. During the QoS negotiation, data may still flow on the connection on a best-effort basis. Cells that belong to the QoS negotiation message are marked with a Payload-Type Indicator (PTI), possibly as resource management (RM) cells, so that it may flow to the controller on the switch. Thus, in fact, QoS signaling and data cells (or messages) may be interleaved because of the PTI value being distinct from one another.

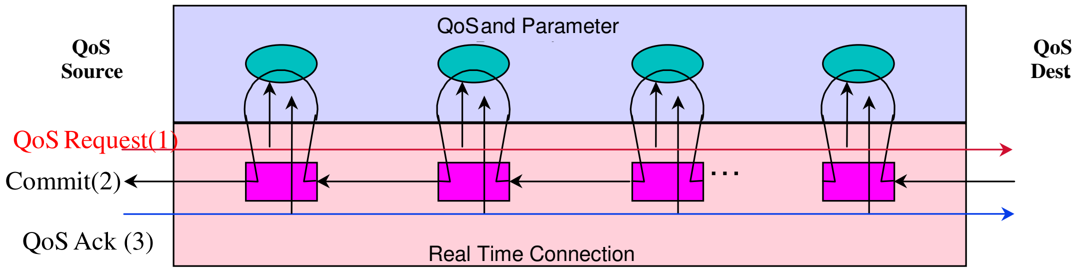

Figure 6: Three way QoS setup

Various alternatives for detailed QoS negotiation can be considered here, including the conventional three way setup depicted in Figure 6, and one which is consistent with the RSVP-like signaling proposed for IP networks. With reference to Figure 6, the QoS request is multicast (locally within the switch) to all switch controllers in the path and to the next link at each switch , facilitating parallel processing in the controllers (1). The Commit message traverses the reverse path, slaloming to every controller, collecting the commitments (2). The QoS Ack. multicasts (again locally) the commitment to all controllers (3). In UNITE a QoS request may be initiated by any participant of a multicast, the core (if present), source or a leaf. Moreover, unless otherwise dictated by higher level policies, core/source and leaf initiated QoS may all be used at different times for the same multicast. As an illustration of the potential of UNITE, we describe the case of Leaf Initiated QoS request by referring to Figure 7.

The leaf initiated QoS requests carry the demand from the receivers upstream. When the QoS request arrives at a switch, the demand is noted at the switch. The switch conveys upstream, the maximum of all the demands observed from the different branches (a leaf node or a switch may be at the end of the branch). Note that different leaves may issue their QoS requests at different times. The switch examines each QoS request and transmits a request upstream only if the QoS request is higher than the current maximum. When the demands arrive at the core/sender, the permit returned is the minimum of the offered capacity, the demands received from the leaves and the available link capacity.

2.7 Dealing with aggregation

One of the major issues that has been raised with state in a connection-oriented network is that of scalability. Since there is per-flow state (both in terms of connectivity and quality of service) in the intermediate switches for every conversation between end-systems, the concern is the ability of network switches to support a large number of flows simultaneously. This is particularly relevant when we consider the growth of IP traffic in the Internet currently, where the number of flows that may have to be supported in a provider's backbone may range in the several tens or hundreds of thousands.

This has caused the Internet to seek other means of supporting QoS than with the simple per-flow state approach of RSVP, or with individual VCs in an ATM network. Furthermore, with the very short-lived flows we observe in the current traffic, supporting even best-effort traffic may require us to support aggregation of simple connection state. In the ATM framework, one step in the aggregation process is to consider the use of virtual paths.

One of the ways that signaling load on the network is reduced is through the use of Virtual Paths.

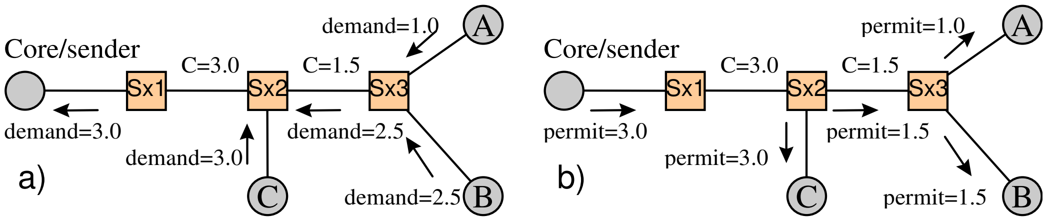

Figure 7: Multicast QoS. Leaf initiated QoS (a) demand phase, (b) permit phase

A substantial number of these flows (in the worst case up to a full 64K of the flows between the ingress and egress of a VP) may be aggregated into a virtual path (VP). The signaling load on the switches internal to the VP is dramatically reduced, since these are carried transparently through the VP. UNITE mechanisms for signaling will also operate in a similar fashion: the micro setup sent on the signaling VC arrives at the entry point of a VP, and is processed at this ingress switch. But, the micro-setup is forwarded to the exit point of the VP, transparent to the switches internal to the VP. After the exit point of the VP, the micro setup is now sent on a hop-by-hop basis to the destination, just as before. For all intents and purposes, the VP appears as a single link for UNITE signaling between the entry and exit switches. The benefits of UNITE overcoming the end-end (source to destination) propagation delay is however still derived. The source may start transmitting as soon as the very first hop processing is completed, rather than a request going from the source going through a set of switches, the VP and then finally another set of switches before reaching the destination and back. Also, the lightweight nature of UNITE's micro setup is still beneficial to the set of switches that are not part of the VP (and the number of these may in fact be larger than the number of switches in the VP). Further, the QoS processing at the switches within the VP are avoided with UNITE, just as it is in the UNI environment. In terms of details, let us assume that the VP is set up a priori between the ingress and egress switches, either by configuration or through a provider specific policy to set up the VC when the switches come up. Let there be a specific signaling VCI on the VP. When the micro-setup from an end-system arrives at the ingress switch, based on the destination address and QoS class, the VPI is determined. The micro-setup, since it is sent on the signaling VC for the end-systems, is not looked at by the VP switches. Note that the particular VP on which the micro-setup is transmitted is a local policy decision at the ingress switch. When the micro-setup arrives at the egress switch, the VCI for the micro-setup indicates that this is a signaling cell, and causes a connection to then be set up from the egress switch to the destination. The OAM and RM cells on the VCs transmitted by the end-systems would not be processed by the switches within the VP.

3 Networking evolution and related work

The perceived needs of applications for fine grained guarantees has led to the development of fine grained QoS models such as those defined by for ATM networks. Part of the motivation to provide strict QoS has been to support real-time interactive communication even with relatively simple endpoints. More inteligent endpoints that are capable of adjusting to a variety of network conditions have less stringent needs for guarantees (especially in terms of jitter). However, application demand for QoS support from the network remains, ranging from the fundamental need to support real-time communication to provide compatability with existing equipment and protocols. A further demand for networks capable of providing QoS comes from the network providers, because of the need to provide differentiation and to avoid the "tragedy of the commons". It is also a means for network providers to introduce a variety of revenue models to support the investment in better infrastructure. The model for providing QoS support considered in this paper is in part based on the experience gained from the protocol development for connection oriented networks such as ATM. In this model, utilization information is distributed together with topology information by a QoS aware routing protocol. This utilization information is used to set up connections along paths through the network. Some of these solutions might not be directly applicable within a connectionless IP network. However, we see many of the ideas developed in connection oriented frameworks, including some first introduced by UNITE, being adopted in various ways by connectionless IP networks seeking to provide QoS guarantees.

Consider for example recent developments in RSVP. While originally conceived as a resource reservation protocol [6], RSVP has now evolved to be a generic signaling protocol for the Internet. Recent additions and developments to RSVP include the following:

- An explicit route object allows RSVP messages to following a particular route through the network. Together with an extention to carry label information these extentions allow RSVP to be used in an MPLS network [7]. A major application of this combination of technologies appears to be traffic engineering. Allowing label switched paths (LSPs) to be established based on information received from a QoS enabled routing protocol, offers an equivalent solution to ATM.

- Aggregating per flow RSVP requests in both the data and the control path is dealt with in [8]. In the data path, aggregation is done by mapping individual flows into coarse QoS classes either by means of tunneling or by appropriate marking of the type of service (TOS) field in the packet header. The means for aggregating control path information depends on the data path strategy used. In the case of tunnels, individual per flow RSVP requests are invisible while being tunneled. The tunnel endpoints are responsible for maintaining an RSVP session between them which reflects the QoS requirements of the individual flows being carried in the tunnel. One scheme for aggregating control messages in the case of TOS field aggregation in the data path involves per-flow RSVP messages being sent through the backbone, in the traditional manner, until the ingress and the egress routers are aware of each other. At this point, RSVP messages for aggregated flows can be sent directly between the backbone edge routers and various mechanisms are proposed to allow per flow RSVP messages to go through the backbone without being processed. An alternative method which completely eliminates the processing of per flow RSVP messages in the backbone is also proposed.

- In [9] a framework for providing end-to-end QoS in a network with Integrated Services in the access or stub network and Differentiated Services in the backbone or transit network is proposed. While not proposing any specific enhancements to RSVP, the proposed scheme also assumes that per flow RSVP messages in the stub network will be carried transparently through the backbone network. The assumption is that the Diffserv network will be capable of providing some "well-understood" service to ensure end-to-end QoS. The proposal assumes that per flow RSVP messages will trigger admission control functions in the Diffserv network, on some aggregate basis. While not explored in any detail the possibility of using RSVP to dynamically set Diffserv codepoints in the network is mentioned. A broad indication of tolerance to delay is proposed to be added to to Intserv RSVP messages to aid the mapping of intserv RSVP requests to Diffserv per-hop-behaviours (PHBs).

The exact nature of the QoS guarantees that a Diffserv network will be capable of providing has not yet been fully defined. Indeed, initial Diffserv networks are expected to make use of static provisioning and to set up PHBs at fairly slow timescales [10]. In such an environment any QoS commitment that the network can provide will apply as long as there is enough capacity to carry a specific class and all classes of higher priority. The use of a QoS aware routing protocol would appear to add little if any value in such an environment. If however additional capacity is required to carry traffic of a certain class and require determination of a suitable route for that class, it might involve the use of a QoS routing protocol in addition to a signaling protocol to dynamically change capacity allocated (e.g., by changing weights or thresholds) for a class in the network.

Early work on QoS enabled networks treated admission control mainly as a resource availability issue. There is an increasing realization though that admission control in QoS enabled networks will also heavily depend on policy issues. Such policy issues deal with the question of whether a client should be allowed to make use of resources given that such resources are available. Policy based admission control is normally coupled to the revenue model espoused by the service provider. In general we expect there to be a (signaling) protocol selecting and pinning a QoS aware route through the network and performing the reservation for a flow or its aggregate (the subject matter of this paper), as well as a service specific protocol to deal with policy decisions. The coupling between these different signaling functions requires careful consideration:

- First, the coupling should be loose as there is expected to be many more ways to decide policy than to decide resource availability. Different service providers might use different policies. In particular when different service providers use the same physical network infrastructure, they might use different policies and use different (service specific) protocols between policy servers in the network. Although a policy decision may in part be based on coarse grained information about resource availability, a policy decision to authorize the use of a resource does not imply that the (subsequent) request for resources will be admitted.

- Care should be taken to ensure that protocols that invoke service specific policies can make use of QoS information without requiring them to duplicate information about the current state of the network themselves. (Such information is maintained by QoS routing protocols.) Furthermore, it is desirable, that service specific protocols remain unaware of the specifics of the current network layer topology. This separation of information and function is desirable to enable the network infrastructure to be able to both operate at a different time scale than service specific protocols as well as support multiple services.

These issues are probably best explained by means of an example. In [11] a signaling protocol, called DOSA (for Distributed Open Signaling Architecture), is presented for telephony in an IP network. The underlying assumption in DOSA is that telephony is an application that will require QoS guarantees from the network. Resources to provide such guarantees are expected to be limited, at least in some segments of the network, thus requiring QoS aware protocols for routing and signaling. The DOSA architecture deals with the policy issues concerning access to these resources as well as functions specific to the telephony service. In particular DOSA can operate in a fashion reminiscent of today's telephone service where charging only commences when a called party picks up the phone. At the same time, having picked up a ringing phone, a user should be able to depend on the fact that resources have already been assured to sustain the established call.

The DOSA architecture represents what has been called a "service specific control architecture" in [12]. Different services can be expected to provide different control architectures, each of which will be responsible for managing some resources in the underlying network infrastructure. The switchlet abstraction [13], showed that this is a safe way of achieving the needed flexibility without jeopardizing the intergrity of the network as a whole. Also, service specific architectures such as DOSA, should be able to interact with QoS management protocols to enable efficient use of QoS enabled networks, without requiring them to maintain QoS information themselves.

The switchlet ideas are complementary to those of UNITE [14, 13]. The putting together of the network is provided by the connectivity abstraction provided by UNITE. The connectivity of UNITE could be provided among a set of switchlets that is a subset of the switches in the network. Layered on top is the QOS abstractions and protocols of UNITE and other signaling frameworks including service specific protocols. These could be local to each switchlet at a switch. We could customize the processing of a particular VPN's connection by using the QoS protocol of UNITE to communicate to the switchlets in the path, establish customized state and processing and thus achieve the customization of the path that a particular VPN desires.

4 Summary

Supporting quality of qervice in networks has been an area of active research for a long time. However, there has been a renewed interest in QoS for networks because of the introduction of the Differentiated Services framework for IP networks, the active use of packet networks for carrying multimedia information and for providing telephony services. Although not all of the frameworks for QoS currently incorporate a signaling protocol, there is acceptance of the need for such a protocol. In this paper we identified the common underlying components of a signaling protocol that are needed for effective support of QoS in networks. These include:

- Stitching together an end-to-end path for communication between end-systems. This involves the determination of the path and requires some information (e.g., utilization) about the path.
- In some cases, we may need to pin the route followed by a sequence of packets, and may want to achieve this very efficiently.
- Customization of the characteristics of this path. This requires the end-systems to convey information to the network about the requirements of the flow (in some cases, this is implicit).

The primary differences between the various approaches to QoS support are in terms of the timescales at which the path selection is done: in connection oriented networks finding a path is done at connection setup timescales, whereas in a connectionless Differentiated services type network this is done on a packets by packet basis. Customization of the QoS of the path is performed through explicit signaling, dynamically, on a per-flow basis in a connection-oriented network (especially when using the UNITE protocol) or somewhat more statically (provisioning timescales), on an aggregate basis with a connectionless Diffserv network.

We discussed a signaling protocol, UNITE. The primary characteristic of UNITE is to recognize and exploit the separation of connectivity from Quality of Service for a connection. Thus, UNITE achieves much of the fundamental functions needed in a signaling protocol in an efficient yet flexible manner. UNITE has been prototyped on two switch platforms. Performance of a UNITE prototype implementation is described in [5].

We finally made the observation that QoS at the network layer by itself is not enough. There is a need to enable its use, by means of service specific protocols. The design of service specific protocols needs to recognize and take advantage of the underlying QoS mechanisms of the network. This is exemplified by DOSA, an architecture for packet telephony. We also identified the role of switchlets in supporting the overall approach to QoS in networks.

References

[1] E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, "A framework for qos-based routing in the internet." RFC2386, August 1998.

[2] ATM-Forum, "Interim Inter-Switch Signalling Protocol Specification (PNNI 0)," *The ATM Forum: Approved Technical Specification*, December 1994. af-pnni-0026.000.

[3] ATM-Forum, "ATM User-Network Interface Specification - Version 4.0, (UNI 4.0)," *The ATM-Forum: Approved Technical Specification*, July 1995. af-sig-0061.000.

[4] K. K. Ramakrishnan, G. Hjalmtysson, K. van der Merwe, F. Bonomi, S. Kumar, and M. Wong, "Unite - an architecture for lightweight signaling in atm networks." ATM Forum contribution 98-0520, 1998 July.

[5] K. K. Ramakrishnan, G. Hjalmtysson, K. van der Merwe, F. Bonomi, S. Kumar, M. Wong, and V. K. Samalam, "Unite: Establishing best-effort connectivity." ATM Forum contribution 98-0782, October 1998.

[6] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource reservation protocol (rsvp) – version 1 functional specification." RFC2205, September 1997.

[7] D. O. Awduche, L. Berger, D.-H. Gan, T. Li, G. Swallow, and V. Srinivasan, "Extentions to rsvp for lsp tunnels." draft-ietf-mpls-rsvp-lsp-tunnel-00.txt, November 1998. Work in progress.

[8] R. Guerin, S. Blake, and S. Herzog, "Aggregating rsvp-based qos requests." draft-guerin-aggreg-rsvp-00.txt, Nov 1997. Work in progress.

[9] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, K. Nichols, and M. Speer, "A framework for use of rsvp with diff-serv networks." draft-ietf-diffserv-rsvp-01.txt, November 1998. Work in progress.

[10] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services." RFC 2475, December 1998.

[11] P. Goyal, A. Greenberg, C. Kalmanek, B. Marshall, P. Mishra, D. Nortz, and K.K.Ramakrishnan, "Integration of call signaling and resource management for ip telephony," *IEEE Network*, 1999.

[12] J. van der Merwe and I. Leslie, "Service Specific Control Architectures for ATM," *IEEE JSAC*, vol. 16, pp. 424–436, April 1998.

[13] S. Rooney, J. E. van der Merwe, S. Crosby, and I. Leslie, "The tempest, a framework for safe, resource assured, programmable networks," *IEEE Communications Magazine*, vol. 36, pp. 42–53, October 1998.

[14] J. van der Merwe and I. Leslie, "Switchlets and Dynamic Virtual ATM Networks," *Integrated Network Management V*, pp. 355—368, May 1997.