On the Feasibility of RSVP as General Signalling Interface
Martin Karsten, Jens Schmitt, Nicole Beriér, and Ralf Steinmetz

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

On the Feasibility of RSVP as General Signalling Interface

Martin Karsten[1], Jens Schmitt[1], Nicole Beriér[2], and Ralf Steinmetz[1,2] *

1: Darmstadt University of Technology
Merckstr. 25 • 64283 Darmstadt • Germany

2: GMD IPSI
Dolivostr. 15 • 64293 Darmstadt • Germany

{Martin.Karsten,Jens.Schmitt,Nicole.Berier,Ralf.Steinmetz}@KOM.tu-darmstadt.de
http://www.kom.e-technik.tu-darmstadt.de/

Abstract. Much debate exists whether explicit signalling is eventually required to create a reliable and integrated multi-service Internet. If yes, further disagreement exists, how such signalling has to be carried out. In this paper, we adopt the point of view that signalling of Quality of Service (QoS) requests must not be abandoned, given the high level of uncertainty about the future traffic mix in an integrated communication network. We present a flexible architecture, based on an extended version of RSVP, for signalling QoS requests. We approach the question of RSVP's suitability for this purpose from two directions. First, we present the design of a QoS signalling architecture describing flexible, yet efficient interfaces between participating entities. Second, we report practical experience from our ongoing effort to implement key components of this architecture.

1 Introduction

The invention of RSVP [1] and the Integrated Services (IntServ) architecture [2] has created significant expectations about the migration of the Internet towards an integrated multi-service network. Afterwards, objections against the resulting signalling and data forwarding complexity have led to the establishment of a new working area, called Differentiated Services (DiffServ) [3], in which much simpler solutions are sought. However, recent results [4,5,6] have shown that only by installing static service level agreements (SLA), the theoretical worst-case performance guarantees for providing per-flow services might exhibit a larger conflict with the objective to utilize resources as efficient as possible, than often assumed. We conclude that building end-to-end services out of DiffServ Per-Hop-Behaviour (PHB) forwarding classes will not be fully sufficient to satisfy the diverse end-to-end requirements for a future Internet. Instead, we favour a combination of signalling service requests with a variety of topological scopes. On the other hand, we also question the usefulness of precipitous standardization of new signalling mechanisms, before the full potential of existing (yet maybe extended) proposals has been investigated and exploited.

In this paper, we try to show how stringent decoupling of service interfaces from service creation (as e.g. initially intended for RSVP and IntServ) can create a new point of view on service signalling. The main goal for our work is to design and realize a flexible QoS signalling architecture, which is composed out of a few basic building blocks. At the same time, we try to adhere to existing standardization proposals as much as possible. This work is intended to be aligned with the recent IAB draft on QoS for IP [7].

*. This work is partially funded by the European Commission, 5th FW, IST, Project M3I (11429).

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

The paper is organized as follows. We present an overall signalling architecture in Section 2 as well as certain extensions to the current RSVP specification in Section 3. In Section 4, we present a simple use case analysis to demonstrate the flexibility of our proposed architecture. Afterwards, in Section 5, we present experiences and quantitative results of our RSVP implementation to illustrate the point of view that, although its theoretical complexity, RSVP is not as inefficient as often assumed. We relate our work to other approaches in Section 6, as far as possible, and conclude this paper in Section 7 with a summary and an outlook to future work items.

2 Proposed Architecture

One must clearly distinguish two roles of a signalling protocol like, e.g. RSVP. It has been initially designed as a distributed algorithm to enable multiple entities to cooperatively deliver a certain service, i.e., multiple routers creating a reservation-based, end-to-end transmission service. On the other hand, it can be considered as an interface specification to request services, regardless of how the service is technically constructed. The most important requirement to consider when assessing the basic architectural alternatives, is to consider interfaces (especially interfaces to end-users) as stable and hard to change. Therefore, service interfaces must be chosen carefully to be very flexible, robust and compatible with future developments. On the other hand, a certain service interface must not inhibit the performant realization of services. The best way to accommodate these goals is to make interfaces as lean yet expressive as possible.

2.1 Concept

Our proposal for an overall QoS signalling architecture conceptually consists of three layers as depicted in Figure 1. It is assumed that a basic connectivity mechanism exists, which is given by a routing protocol and packet forwarding nodes called *router*. This is described as *packet layer* in the picture. The actual QoS technology is represented by an intermediate *QoS layer*. An entity that, besides carrying out router functionality, also performs packet-based load management by policing, shaping, scheduling, or marking packets for a certain scheduling objective is called *QoS enabler*. A pure QoS enabler, however, does not participate in end-to-end signalling. Advanced end-to-end services that allow to dynamically specify performance characteristics are realized using a complementary interface on the *service layer*. The entities of this layer, which handle service signalling and potentially flow-based load control (admission control) are denoted as *service enabler*. A service enabler can also perform the role of a QoS enabler. Of course, in a future QoS-enabled Internet, further open issues, such as QoS routing have to be addressed, as well. However, their eventual precise definition is currently beyond the scope of a QoS signalling architecture.

The focus of this work is to flexibly realize a service layer that allows to integrate a variety of QoS layers. In the conceptual architecture, the layers can be considered as roles. Compared to previous work, the role or functionality of each layer is not bound to certain nodes in the network topology. Instead, it depends on a network operator's particular choice of QoS technology and furthermore, on the service class, which node carries out the role of a certain layer. Detailed use cases are presented in Section 4.

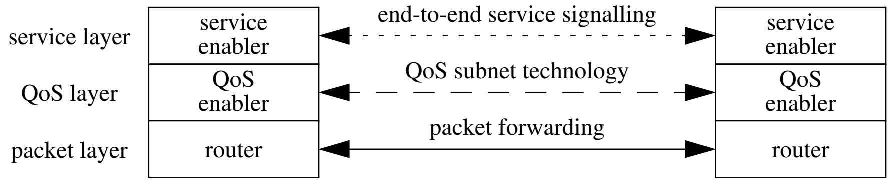

Figure 1: QoS Signalling Architecture - Conceptual View

2.2 Topological View

When considering the topological view on this signalling architecture, intermediate nodes have to be distinguished between edge routers and interior routers. Service signalling takes place between at least edge routers. Depending on the service class and the particular QoS technology, intermediate routers might participate in the signalling, as well. Furthermore, subnets might employ bandwidth brokers to carry out resource allocation for the complete subnet for certain service classes. In this case, service requests can be forwarded from edge routers to the bandwidth broker. All nodes are classified as either *service-aware*, *partially service-aware* or *service-unaware* as depicted in Table 1. Note that the term service-unaware does only denote that a node does not participate in service signalling. It might nevertheless carry out the role of a QoS enabler and thus, perform packet-based QoS-enabling mechanisms. In case of partially service-aware nodes, these nodes have to distinguish whether to process or just forward a service request. The main criterion for this distinction is very likely to be the service class. This is further discussed in Section 4.

Table 1: Service Awareness of Network Nodes

Service Awareness	Description
service-aware	RSVP-capable, support for all service classes
partially service-aware	RSVP-capable, support for some service classes
service-unaware	not RSVP-capable

2.3 RSVP as General Mechanism

In order to satisfy both goals of flexibility and optimization for highly demanding services when realizing a service layer, a solution is given by a uniform extended RSVP interface for advanced services. Using such an interface as service layer entity at each traffic exchange is both sufficient and effective to realize the conceptual architecture for multiple topological and QoS technology alternatives and to create meaningful end-to-end services. This design represents the choice to carry on with the Internet service architecture and employ RSVP (including the extensions presented in Section 3) as the primary signalling mechanism, especially for inter-domain signalling. Initially, it can then be used as a service interface between bandwidth brokers (particularly for dynamic DiffServ SLAs).

However, the main motivation is given by the advantage that a future migration to employ RSVP in its initially intended style as distributed algorithm to request and provide per-flow and potentially per-node service guarantees will be alleviated, if the basic

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

mechanisms are already in place. In such a future scenario, RSVP then acts as a signalling mechanism between each node, as well. Consequently, it is intended that a router employing this extended version of RSVP can efficiently handle both per-flow and aggregated service invocations of multiple service classes. The alternatives to invent different signalling mechanisms for per-flow and aggregated service requests or different mechanisms for end-to-end and backbone signalling seem clearly inferior, especially if RSVP can be applied beyond its initial scope without introducing a large overhead.

3 RSVP Extensions

There are mainly two shortcomings in the currently specified version of RSVP, which aggravate its application as a general service interface:

- Traffic flows are either identified by host or a multicast addresses, e.g., the specification of subnets as source or destination address is not possible.
- Path state information has to be stored for each service advertisement in order to ensure correct reverse routing of service requests.

In order to appropriately extend RSVP's functionality, existing ideas [8,9] have been taken up for this work and augmented to design a general processing engine for a lean and flexible service interface. The major goal is to achieve a high expressiveness for service interfaces. The extensions are mainly dedicated for, but not restricted to, unicast communication (including communication between subnets) and cover cases where the per-flow model of traditional RSVP signalling, which eventually exhibits quadratic state complexity [9], seems inefficient, because the requested transmission performance characteristics do not require flow isolation at each intermediate node. In that sense, the extensions are targeted to aggregated service requests on the control path. This has to be distinguished from the issue of aggregating flows on the data path. For the latter, careful network and traffic engineering, e.g. using MPLS [10], is required or alternatively, strict performance guarantees might be given by applying network calculus to multiple flows [11]. For both multicast in general and non-aggregated performance-sensitive (i.e. inelastic) unicast communication, the current version of RSVP can be considered as very well-suited, especially if recent proposals to increase the overall efficiency of RSVP operation [12] are realized. Note that the following extensions can be implemented without increasing the complexity of an RSVP engine. However, they do extend the application scenarios to cover a variety of new alternatives. This is demonstrated by a use case analysis in Section 4.

3.1 Compound Prefix Addressing

The current specification of RSVP supports only host and multicast addresses. In order to specify service requests for traffic aggregates between subnets, the notion of addresses has to be extended to cover CIDR prefixes for network addresses. A respective proposal has been made in [8]. In the following, the term *generalized address* is used to refer to either an end-system's address or a network's address expressed as CIDR prefix, extended by class A network addresses and the special address prefix 0.0.0.0/0 denoting complete wildcarding. Additionally, it might be necessary to specify several of such addresses within a single session or sender description, thus the notion of a *compound ad-*

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

dress is introduced, which consists of a set of generalized addresses. Of course, a dedicated node must exist within an end-subnet to receive and respond to such service requests. In principle, any node can emit such requests as long as they are authorized.

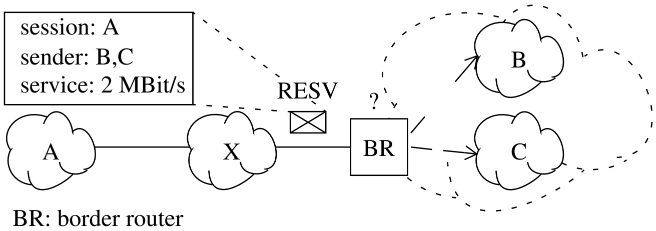

Figure 2: Compound Addresses and Scoping Style

In order to employ the full flexibility of compound addresses, it is inevitable to introduce a further generalization to specify their handling at certain nodes. During the transmission of RSVP messages, targeted to a compound address, the border router towards the specified subnet(s) will be hit. In that case, it has to be decided whether the message is forwarded towards multiple destinations or not. If the message is not forwarded, then the resulting service essentially covers only a portion of the end-to-end path. If however, the message is forwarded into multiple subnets, it is not immediately clear how to interpret any quantitative expression of performance characteristics. The term *scoping style* is used to describe the alternatives that such a message is forwarded to multiple next hops (*open scope*) or not (*closed scope*). To this end, it is an open issue whether the scoping style should be chosen by the node issuing a request or whether it is determined by the network provider depending on its local policy how to provide certain services. As this is a matter of strategy and not mechanism, it is beyond the scope of this work to extensively investigate this question. Nevertheless, some use case examples are given in Section 4. In Figure 2, an example RESV message is shown to illustrate the choice between both alternatives.

If RSVP's addressing scheme is extended to include compound addresses, new challenges are presented to the data forwarding engine of a router. In order to support flows targeted to or sent from an end-system at the same time as a session involving the subnet of this end-system, a longest-prefix match on both destination and source address might be necessary to distinguish which packets belong to which session. However, it can be expected that any service establishing performant communication for traffic aggregates between subnets is going to be built using a packet marking scheme, as e.g. the DiffServ model. In the DiffServ architecture, such a case is already considered and alleviated by the fact that only edge-routers are expected to do the full classification to isolate aggregate service contracts from individual flows. In the core of the network, traffic belonging to aggregates is forwarded according to its DiffServ marking and individual flows requiring total isolation can be appropriately serviced using a dedicated DiffServ mark and full packet classification. The same marking scheme can be applied to RSVP messages themselves, such that per-flow request messages are transmitted to the appropriate end-subnet, but not processed by nodes along a trunk flow. This allows for transparent end-to-end signalling, even in case of intermediate flow mapping.

A somewhat different treatment of port numbers is necessary to incorporate compound addresses into RSVP. It might be useful to specify a port number, if e.g., the resulting service is used for a single application which can be identified through the port number. In any other case, the port number should be set to zero and effectively denote wildcarding. Analogous to the description in the previous paragraph, a classification challenge exists, which will be alleviated by employing a DiffServ-like marking scheme.

A scheme of compound addresses in combination with the choice of scoping style is more appropriate for service requests between subnets than the initial approach to CIDR addressing of RSVP messages [8], because it overcomes the limitations induced by restricting source and destination to a single address prefix each. Furthermore, the scoping style provides a controllable way to deal with the resulting flexibility. Thereby, it is well-suited to especially provide a signalling mechanism and interface between bandwidth brokers which control the establishment of SLAs that are eventually provided to traffic aggregates by means of DiffServ code points.

3.2 Hop Stacking

To reduce the quadratic amount of state that has to be kept by routers in case of traditional RSVP signalling, it is quite trivial to extend its specification similar to [9]. Usually, PATH messages are sent along the same path as the data flow and state containing reverse routing information is kept at each node to allow forwarding of a RESV message along the reverse path towards the sender. In order to alleviate this effect for intermediate nodes, a mechanisms termed *hop stacking* can be incorporated into RSVP. Each router has the option to replace the RSVP_HOP object by its own address and store appropriate state information in PATH messages (traditional operation). Alternatively, the address of the outgoing interface is stored as additional RSVP_HOP object in front of existing ones. During the service request phase, the full stack of such hop addresses is incorporated into RESV messages and used at respective nodes to forward the service request to previous hops, if no PATH state has been stored. On the way upstream, such a node removes its RSVP_HOP object and forwards the message to the next address found in the stack. This mechanism allows to install state information for service requests without the necessity to keep PATH state for each service announcement. This specification introduces even further flexibility as compared to other approaches in that stacking of hop addresses is optional and can be mixed with traditional processing within a single session. A node might even remove the full stack, store it locally together with the PATH state, and insert it into upstream RESV messages, such that the next downstream node does not have to deal with hop stacking at all. Figure 3 illustrates the flexibility of hop stacking. In this picture, nodes C and D perform hop stacking instead of storing local state whereas node E removes the full stack and stores it locally, such that node F does not realize the existence of stacked hops at all. An according RESV message travelling along the reverse path, can find its way back to the sender by local state or stacked hop information.

From a node's point of view, hop stacking provides a transparent method to employ other approaches for QoS provision without per-flow state at intermediate nodes, e.g., RSVP over DiffServ-capable networks [13]. However, from an overall system's point of view, hop stacking defines a generic mechanism to carry out RSVP signalling without PATH state at each node. It can be used for trunk signalling, tunnelling and provides

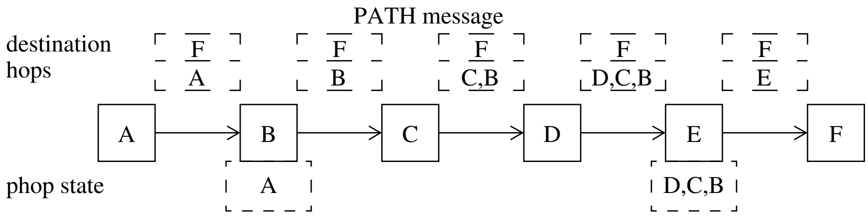

Figure 3: Hop Stacking for RSVP Messages

for an open interaction with traffic and network engineering. In that sense, slightly more freedom is taken to extend the existing RSVP specification than other approaches.

3.3 Interface Semantics

While the extensions presented above form the procedural part of this proposal, it is important to define coherent semantics at a service interface. The inherent meaning of accepting a traditional RSVP message is to appropriately process and forward the request, establishing an end-to-end resource reservation. In our architecture, the semantics are changed such that the meaning of accepting a service request is a (legal) commitment to deliver this service, regardless of its actual realization. For example, compound addressing provides an interface to transparently incorporate IP tunnels as presented in [14]. Similarly to the notion of *edge pricing* [15], this creates a notion of *edge responsibility* for the end-to-end service invocation. Effectively, an application's data flow might be mapped onto several consecutive network flows in the notion of traditional RSVP. In that sense, intermediate nodes carrying out that mapping might actually be considered as “RSVP gateways” or “service gateways".

4 Use Case Analysis

In this section, a collection of use cases is described to conceptually show the flexibility of the RSVP-based signalling architecture to integrate diverse QoS technologies and create a variety of service scenarios besides RSVP's initial designation for IntServ. The use cases focus on the mechanisms of service layer signalling between service enablers. In case of multiple alternatives, it is left open to further work to determine the optimal strategies to map service requests onto the underlying QoS technology.

4.1 Supporting Diverse Subnets

In the following it is briefly presented, how various QoS subnet technologies can be integrated by this QoS signalling architecture. There has been a lot of work to support diverse link-layer mechanisms, DiffServ clouds and ATM subnets. Most of these are well-known and treated (together with link layer technologies) by the IETF ISSLL working group (see [16] for a list of documents) and covered by a number of other publications, as well. However, there's an additional scenario explained below.

Service Signalling across ECN-priced Subnet. A somewhat speculative proposal to provide QoS has been made in [17]. It is based on intermediate nodes carrying out statistical ECN-marking, which are interpreted as small charges at edge systems. It is

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

claimed that the resulting economic system provides a stable resource allocation which then could be considered to resemble a certain QoS. In order to mimic the all-or-nothing characteristic of regular admission control, the ingress of the subnet acts like a risk broker and decides whether to accept or reject a service invocation. This risk broker subsequently undertakes the economic risk of guaranteeing the accepted service even in the presence of rising congestion and thus, charges. Another option is for the ingress node to adapt the sending rate to the current congestion situation. Since the ECN mechanism is an end-to-end mechanism and usually requires a transport protocol to carry the feedback from the receiver to the sender, it is not immediately obvious how such an approach should be realized for a partial path in the network. However, if RSVP signalling is employed between the end nodes of such a partial path, the periodic exchange of RSVP messages can be used by the egress node to provide at least a some kind of feedback to the ingress node.

4.2 Flexible Service Signalling Techniques

The following scenarios present a variety of service invocations that can be supported using the RSVP-based QoS signalling architecture. Note that all the scenarios presented below can be carried out at the same time in the same infrastructure.

Reduced State Service Signalling in Backbone Networks. In this scenario, a backbone network is assumed, which allows to establish trunk reservations between edge nodes, which are dynamic in size and routing path. Because of a potentially large number of edge nodes that advertise services to each other, it may be inappropriate to potentially keep state for each pair of edge nodes at routers. Furthermore, the service class does not provide precise service guarantees, but rather loosely defined bandwidth objectives. RSVP signalling can be carried out between each pair of nodes including the hop stacking extension. Path state is not stored at intermediate nodes and reservations towards a common sender are aggregate at each node. Consequently, the worst-case amount of state that has to be kept at each router is linear to the number of nodes, instead of quadratic. This example resembles the basic state reduction technique of BGRP [9].

Service Mapping of Flow Service to Trunk Service. The notion of compound prefix addresses allows to express service mappings of individual flows into aggregated trunk services. Individual flow requests that arrive at the ingress end of the trunk service are incorporated into a single service request, which is described by a compound prefix address and transmitted to the other end of the trunk. In Section 3.1, it is discussed, how to distinguish trunk traffic from other packets which might be exchanged between the corresponding end systems. Alternatively, a tunnel might established for the aggregation part of the data path [14] and eligible packets are encapsulated into the tunnel. Nevertheless, it is useful to have a notion to describe the aggregate traffic flow, such that signalling can be carried out across multiple autonomous systems.

Lightweight Service Signalling. One might even go one step further and consider an RSVP PATH message as service request, while RESV messages only confirm the currently available resources. In that case, the end-systems keep track of the network state along the data path and no state information is stored at intermediate nodes. Such a sce-

QofIS'2000: http://www.fokus.gmd.de/events/qofis2000/

nario can be realized by a specific service class instructing each intermediate node to report its current load situation and service commitments, but without carrying out any particular activity for this request. PATH messages record their way through the network by hop stacking and RESV messages are initiated by receivers including the amount of service that this receiver requests. On their way back to the sender, the RESV message is used to collect the information whether this service is currently possible. Intermediate nodes are free to store as much state information as needed and feasible to report best-effort estimates of the current load situation.

4.3 Application Scenarios

In addition to the simple techniques described in the previous section, the following examples describe more complete application scenarios which employ these techniques.

Service Signalling for Dynamic Virtual Private Networks. Consider a corporate Internet user wishing to establish a virtual private network (VPN) between multiple locations. Each of these locations operates an IP network with a different subnet address prefix. Furthermore, it is deemed important to dynamically adapt the requested VPN capacity according to each locations current demand. In this example, it is examined how the resulting service requests are handled by a backbone network B, which is crossed by traffic from multiple locations. The scenario is illustrated in Figure 4. The corporate subnets are denoted with S_1, S_2, S_3 and S_4. The edge routers are depicted as E_1, E_2 and E_3. Each corporate subnet emits service advertisements (e.g. from a bandwidth broker or dedicated gateway) towards the other subnets, either separately or bundled with a compound destination address. The corresponding service requests might be treated separately or also be aggregated at certain nodes and targeted towards a compound sender address.

As an example, S_1 advertises a certain total amount of traffic towards the other subnets, hence there is no specific description for each subnet. The advertisement is processed by E_1 and forwarded to the other edge devices. If the backbone QoS technology is given by a combina-tion of static SLAs and a bandwidth broker, E_1 obtains the information about multiple egress edge devices from the bandwidth broker and splits up the request accordingly. If intermediate nodes also act as service enablers, the advertisement is forwarded as a bundle, until an intermediate node contains two routing entries for the different destination subnets. This is similar to multicast distribution and applies the service mapping technique described in the previous section. The correspondent service requests from S_2, S_3 and S_4 traverse back to S_1 establishing the subnet-to-subnet service. Because of the dynamic nature of RSVP signalling, the dimensioning of the VPN service can be adapted over the time.

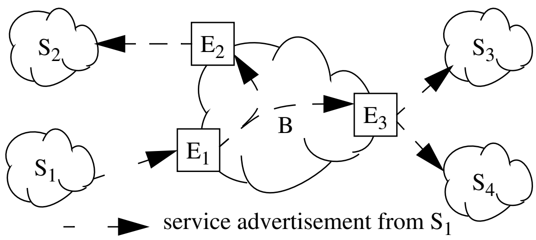

Figure 4: Virtual Private Network Scenario

Inter-Domain Service Signalling. A scenario of inter-domain trunk reservation signalling has been described and carefully analysed in [9]. The same advantages as reported for BGRP can be obtained by employing the reduced state signalling technique described in the previous section. If combined with a recent proposal to bundle and reliably transmit refresh messages [12], RSVP provides a functionally equivalent solution having the same complexity as described there. However, there's no completely new protocol needed.

5 Experiences from Implementing RSVP

As a main building block for our architecture we have realized a new implementation of RSVP, which is designated to clearly express RSVP message processing concepts in the code, be highly flexible and extensible. Furthermore, we have used an object-oriented design and implementation, e.g., to separate container implementations from the rest of the code. This approach allows to experiment with different data structures and algorithms for those containers that can become large and/or crucial for efficient execution. Details of the implementation are described in [18] and [19]. The full source code can be downloaded at http://www.kom.e-technik.tu-darmstadt.de/rsvp/.

We have done some initial performance evaluations, which we consider quite promising with respect to RSVP's ability to deal with a large number of flows. The implementation has not been subject to detailed code-level optimization, so far. However, on a FreeBSD workstation, equipped with a single Pentium III 450 MHz processor, our implementation is able to handle the signalling for at least 50,000 unicast flows under almost realistic conditions (see [20] for details). From these numbers, we deduce that the applicability of RSVP as a general purpose signalling interface and protocol to handle both aggregated and per-flow service requests, is much better than generally assumed.

Besides its complexity of operation, RSVP is often objected to as being overly complex for implementation. Our own experience shows that RSVP indeed exhibits a certain complexity. However, we have been able to realize an almost complete and even multi-threaded implementation of RSVP investing less than 18 person-months of development effort. Given the large applicability and the inherent complexity of the underlying problem of providing performant end-to-end services, we believe that this experience somewhat contradicts those objections.

6 Related Work

Because of the fairly broad scope of this paper, almost all research in the area of QoS for packet-switched networks can be considered as related work. Here, we have to restrict ourselves to only a few relevant examples.

Very interesting work has been carried out in the area of open signalling [21]. However, the focus of this work goes much beyond our understanding of signalling in both effort and goals. It is targeted towards creating programmable interfaces employing active networking nodes. In that sense it can be considered more heavy-weight and less evolutionary as compared to a simple protocol-based approach.

Many other proposal have been made for so-called "lightweight" signalling protocols, e.g. in [12,22,23]. While all these proposals contain interesting properties, we believe it is advantageous to approach the overall problem with a single homogeneous protocol as compared to using multiple protocols for different services and scopes, because a single protocol eliminates functional redundancy.

In comparison to proposals how to carry out the inter-operation of multiple QoS mechanisms, we concentrate on the interface role of a signalling protocol and take more freedom to extend the current RSVP specification. Work as described in [10,13,14] can be considered as complementary, in that low-level detailed aspects of inter-operation are examined and solved.

7 Conclusions and Future Work

In this paper we have discussed and illustrated the feasibility of an extended version of RSVP to serve as general signalling interface for multi-service networks. We have presented a flexible, role-based QoS signalling architecture, based on an extended version of RSVP. This architecture utilizes the observation that a signalling protocol can be considered as carrying out two roles, as distributed algorithm and interface mechanism. Afterwards, we have presented a use case analysis to demonstrate that such a system architecture can enable general service signalling for a large variety of service classes, including aggregate and per-flow services. Experiences and performance numbers from creating the basic building block, a new implementation of RSVP, have been included in this paper to argue against common prejudices in this area. Finally, we have briefly discussed the relation of this work to other approaches.

We intend to realize the full system described in this paper, partially in the framework of a cooperative European research project. If time permits, further examination and tuning of the core RSVP engine will be carried out in the future. A particular focus of our research agenda will be the generic yet efficient realization of inter-operation between RSVP and actual QoS technologies, such as DiffServ. Of course, the discussion about the best way to provide quantitative and reliable QoS assurances in the Internet, to eventually create a truly multi-service network, is still open and further work is needed on all aspects.

References

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

[2] J. Wroclawski. RFC 2210 - The Use of RSVP with IETF Integrated Services. Informational RFC, September 1997.

[3] S. Blake, D. L. Black, M. A. Carlson, E. Davies, Z. Wang, and W. Weiss. RFC 2475 - An Architecture for Differentiated Services. Experimental RFC, December 1998.

[4] A. Charny. Delay Bounds in a Network with Aggregate Scheduling, February 2000. Available from ftp://ftpeng.cisco.com/ftp/acharny/aggregate-delay_v3.ps.

[5] J.-Y. L. Boudec. A Proven Delay Bound for a Network with Aggregate Scheduling. Technical Report DSC2000/002, EPFL-DSC, Lausanne, Switzerland, January 2000.

[6] I. Stoica and H. Zhang. Providing Guaranteed Services Without Per-Flow Management. Technical Report CMU-CS-99-133, Carnegie-Mellon Univ., Pittsburgh, USA, May 1999.

[7] G. Huston. Next Steps for the IP QoS Architecture. IAB Internet Draft draft-iab-qos-00.txt, March 2000. Work in Progress.

[8] J. Boyle. RSVP Extensions for CIDR Aggregated Data Flows. Internet Draft draft-ietf-rsvp-cidr-ext-01.txt, December 1997. Work in Progress.

[9] P. Pan, E. Hahne, and H. Schulzrinne. BGRP: A Tree-Based Aggregation Protocol for Inter-Domain Reservations. Technical Report CUCS-029-99, Columbia University, New York, NY, USA, December 1999.

[10] T. Li and Y. Rekhter. RFC 2430 - A Provider Architecture for Differentiated Services and Traffic Engineering (PASTE). Informational RFC, October 1998.

[11] J. Schmitt, M. Karsten, L. Wolf, and R. Steinmetz. Aggregation of Guaranteed Service Flows. In *Proceedings of the Seventh International Workshop on Quality of Service (IWQoS'99), London, UK*, pages 147–155. IEEE/IFIP, June 1999.

[12] L. Berger, D.-H. Gan, G. Swallow, P. Pan, F. Tommasi, and S. Molendini. RSVP Refresh Overhead Reduction Extensions. Internet Draft draft-ietf-rsvp-refresh-reduct-04.txt, April 2000. Work in Progress.

[13] Y. Bernet, R. Yavatkar, P. Ford, F. Baker, L. Zhang, M. Speer, R. Braden, B. Davie, J. Wroclawski, and E. Felstaine. A Framework For Integrated Services Operation Over Diffserv Networks. Internet Draft draft-ietf-issll-diffserv-rsvp-04.txt, March 2000. Work in Progress.

[14] A. Terzis, J. Krawczyk, J. Wroclawski, and L. Zhang. RFC 2746 - RSVP Operation Over IP Tunnels. Standards Track RFC, January 2000.

[15] S. Shenker, D. Clark, D. Estrin, and S. Herzog. Pricing in Computer Networks: Reshaping the Research Agenda. *ACM Computer Communication Review*, 26(2):19–43, April 1996.

[16] Integrated Services over Specific Link Layers (issll), 2000. IETF Working Group. http://www.ietf.org/html.charters/issll-charter.html.

[17] R. J. Gibbens and F. P. Kelly. Distributed Connection Acceptance Control for a Connectionless Network. In *Proceedings of 16th International Teletraffic Congress*, June 1999.

[18] M. Karsten. Design and Implementation of RSVP based on Object-Relationships. In *Proceedings of Networking 2000, Paris, France*, pages 325–336. Springer LNCS 1815, May 2000. Extended version appears as TR-KOM-2000-01.

[19] M. Karsten, J. Schmitt, and R. Steinmetz. Generalizing RSVP's Traffic and Policy Control Interface. In *Proceedings of the 7th International Conference on Parallel and Distributed Systems (Workshops)*. IEEE, July 2000. Accepted for publication.

[20] M. Karsten. KOM RSVP Protocol Engine: Performance Results, 2000. http://www.kom.e-technik.tu-darmstadt.de/rsvp/performance.html.

[21] A. T. Campbell, A. A. Lazar, H. Schulzrinne, and R. Stadler. Building Open Programmable Multimedia Networks. *Computer Communications Journal*, 21(8):758–770, June 1998.

[22] P. Pan and H. Schulzrinne. YESSIR: A Simple Reservation Mechanism for the Internet. *ACM Computer Communication Review*, 29(2):89–101, April 1999.

[23] G. Feher, K. Nemeth, M. Maliosz, I. Cselenyi, J. Bergkvist, D. Ahlard, and T. Engborg. Boomerang - A Simple Protocol for Resource Reservation in IP Networks. In *Proceedings of IEEE Workshop on QoS Support for Real-Time Internet Applications, Vancouver, Canada*. IEEE Computer Society Press, June 1999.