A Dynamic Sender-Initiated Reservation Protocol for the Internet

P. P. White, J. Crowcroft
Department of Computer Science
University College London
Gower Street
London WC1E 6BT
England
phone: +44 171 419 3701, +44 171 380 7296
fax: +44 171 387 1397
email: p.white@cs.ucl.ac.uk, j.crowcroft@cs.ucl.ac.uk

Abstract

In this paper we discuss the need for resource reservation in the Internet and examine some of the strengths and weaknesses of RSVP, which is currently the most popular of Internet reservation protocols that have been developed. The deficiencies of RSVP motivate our design of a new resource reservation protocol which uses dynamic sender-initiated reservations to achieve a highly bandwidth-efficient reservation mechanism with excellent scalability with regards to round trip time, data rate and number of hosts.

Keywords

Resource Reservation, Quality of Service

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35388-3_42

H. R. van As (ed.), *High Performance Networking*

1 INTRODUCTION

It is clear that the current Internet which was founded upon the concept of 'best-effort' datagram delivery must be enhanced in some way in order to accommodate the changing communications environment. In particular there is a growing demand for real-time applications which have specific Quality of Service(QoS) requirements, especially with regard to end-to-end delay and minimum bandwidth, both of which cannot be guaranteed in the current Internet using traditional connectionless best-effort delivery. Furthermore as the World-Wide-Web is increasingly used for business there is a growing number of users for whom delay bounded access of information is important.

In response to the changing requirements of Internet users, much attention has focussed on the use of resource reservation as a means of providing selected data flows with special QoS commitments in accordance with their needs. Under such a framework it is likely that special QoS delivery would be the exception rather than the rule with the majority of Internet traffic continuing to receive the 'default' best-effort mode of delivery. The special QoS required by a specific data flow can be realised by reserving resources(bandwidth, buffer space) and installing appropriate scheduling behaviour in each router along the end-to-end path followed by the data flow. Such mechanisms require admission control at the individual intermediate nodes to ensure that the request for reservation is only accepted and installed provided sufficient resources are available. In addition, per-flow state[1] in the intermediate nodes will usually be required in order to identify the flows to receive special QoS as well as the QoS to be received.

In order to allow users to invoke special QoS delivery on demand for a data flow several protocols have been developed to enable users to communicate their QoS needs to the intermediate routers along the data path in an IP internetwork. The majority of these protocols initiate the set up of flow-specific reservation state in intermediate routers, a notable exception being the approach described in (Almesberger, 1997) whereby no per-flow reservation state is set up in routers which instead record their reservation commitments as a whole per output port. While this approach potentially offers very good scalability characteristics for a large number of flows, it is dependent upon a certain degree of trust among end hosts not to exceed their indicated traffic levels unless per-flow policing is applied at the network access point. In addition, the approach is only able to offer end applications an approximate minimum bandwidth without any quantitative guarantees on loss or delay and so may not be suitable for applications with stringent QoS requirements such as Distributed Interactive Simulation (Seidensticker, 1997).

Of the reservation protocols that set up flow-specific reservation state, an early example in the Internet is the Stream Protocol, ST (Forgie, 1979) which was

---

[1] The introduction of per-flow state is a significant departure from the initial Internet design philosophy of a pure connectionless network with no per-flow state in the intermediate routers.

limited to unicast reservations. Although its successors, ST-II (Topolcic, 1990) and the more recent ST2+ (Delgrossi, 1995) can handle both multicast and unicast reservations as well as possessing many improvements over ST, the ST group of protocols has attracted little commercial interest. By contrast, another reservation protocol, RSVP (Braden, 1996) has received significant industry support and with good reason. Unlike the ST protocols, RSVP reservation state is soft-state and will time-out in the absence of any refresh reservation requests within a certain time period. This so-called soft-state nature of RSVP provides a very simple failure recovery mechanism over a wide range of fault scenarios and helps to retain much of the robustness that has helped to make IP so successful. The soft-state approach where the end applications are responsible for maintaining the flow-specific router state leads to a significant reduction in complexity compared to a hard-state approach where the network is responsible for maintaining the flow-specific router-state. RSVP has other notable architectural differences compared to the ST protocols such as receiver-initiated rather than sender-initiated reservations[2]. The initial design of RSVP was to a large extent influenced by the needs of multicast conferencing applications although its intended use is now much broader.

While RSVP is concerned merely with signalling the end application's reservation requests to the intermediate nodes, it is the special QoS delivery models[3] that define the node behaviour required to meet the signalled special QoS objectives. The Integrated Services Working Group(intserv) of the IETF(intserv 1998) has standardised several special QoS delivery models while the Integrated Services over Specific Lower Layers(issl) Working Group of the IETF(issl 1998) has developed ways of mapping this network layer QoS onto specific link layer technologies such as ATM, IEEE 802 and Ethernet.

In parallel with the recent Internet growth much interest has been generated by Asynchronous Transfer Mode(ATM), a technology designed from the outset with end-to-end QoS in mind. A necessary component in ATM networks for achieving QoS on demand is a signalling protocol in order to request resource reservations in the intermediate nodes of the end-to-end path. More traditional ATM signalling protcols such as ITU's Q.2931 standard for public networks (ITU-T, 1995) or ATM Forum's UNI standards for private networks (ATM Forum, 1996) use end-to-end handshaking to set up an end-to-end reservation before data transfer can take place. A more dynamic and flexible approach is that provided by the ATM Block Transfer/Immediate Transfer(ABT/IT) (ITU-T, 1996) signalling protocol which sends reservations in-line with data and as such is more conducive to efficient bandwidth utilisation than the more static end-to-end handshaking approach.

---

[2] ST-II+ permits both sender and receiver-initiated reservations, ST-II and ST permit sender-initiated reservations only.

[3] At present two delivery models have been standardised, both of which offer applications an end-to-end minimum bandwidth albeit with different assurances. First, Guaranteed Service which offers applications a loss-free service with an end-to-end delay bound. Second, Controlled-Load Service which does not provide any quantitative guarantees on delay or loss, although qualitatively these parameters can be expected to be the same as for best-effort delivery under low network load.

In the next few sections we present a new QoS signalling protocol known as Dynamic Reservation Protocol(DRP) which could be used to set up the IETF's integrated services models 'on-the-fly' in IP internetworks. We outline the benefits of DRP compared to RSVP before presenting details of packet formats and processing rules and our conclusions.

2 DYNAMIC RESERVATION PROTOCOL (DRP) OVERVIEW

Our protocol, known as Dynamic Reservation Protocol(DRP) incorporates many principles of RSVP along with the dynamic sender-initiated reservation concept of ABT/IT to achieve the following goals:

- High control dynamics to achieve efficient bandwidth usage for both sender-specific and shared reservations.
- Scalability of router-state with regard to number of senders and receivers.
- Scalable and simple approach to One Pass With Advertising (OPWA)[4].
- Minimal receiver complexity.
- Minimal number of messages to implement session-wide reservation changes in large-scale multicast sessions.
- Heterogeneity of reservation QoS classes among receivers of the same session.

DRP allows reservations to be set up 'on-the-fly' by sending Reservation packets, RES in-line with the data flow. In this respect, DRP is similar to ABT/IT although, unlike ABT/IT, DRP does not need to make an end-to-end connection before sending its first in-line reservation packet. Also, unlike ABT/IT, DRP does not support the concept of a sustainable cell rate for the data transfer and consequently the probability of acceptance of a reservation request is determined purely by the available resources at that moment in time. As with RSVP, all flow-specific router state that is set up using DRP is soft-state as we believe that this is a key strength of RSVP that can also be used to good effect in DRP.

The scheme also uses Return (RTN) packets that are reverse-routed up the tree to provide the intermediate routers/switches and sender with certain feedback and end-to-end path information. DRP is applicable to both unicast and multicast scenarios but in the following sections we concentrate on the more complicated multicast case.

3 DRP DESIGN PRINCIPLES

3.1 Sender initiated reservations

The use of in-line reservation packets allows the sender to set up new reservations, or alter existing reservations, on demand at any point in the data transfer. This

[4] This is a term introduced by RSVP, to describe a mode whereby all necessary information is made available(advertised) in advance of making a reservation request so that the correct level of reservations necessary to achieve the target end-to-end QoS can be determined and installed in 'one pass' of the reservation message.

makes it possible to achieve a very close match between the instantaneous service provided by the network and the instantaneous requirements of the data flow. As a result, network resource usage can be minimised. These benefits are particularly prominent for stop/start data flows since the resources can be freed during the quiet periods and re-installed on a just-in-time basis at the start of each activity burst. Such action is precluded with both RSVP and traditional ATM signalling[5] approaches, both of which incur a time lag in excess of the round-trip time when modifying end-to-end QoS to reflect a change in the sender's traffic stream characteristics.

Another advantage of using sender-based reservations rather than receiver-based reservations is a reduction in the volume of processing required at intermediate nodes of a multicast tree each time a sender changes its traffic stream characteristics. With RSVP, each time this occurs and the receivers consequently modify their reservations, it is possible for a node to install a reservation due to a request from a particular receiver, only for it to increase the reservation a moment later when a larger request from a different receiver arrives at the same interface. In fact when the multicast tree serves a large number of receivers it is possible that some reservations may be updated several times before settling down to their steady state values. This effect will be particularly prominent for a Guaranteed Service session since each receiver will probably need to request a different reservation bandwidth even if they require the same end-to-end delay bound. By contrast with DRP, a single pass of the RES packet down the multicast tree will typically[6] achieve the new steady state reservations in the on-tree nodes.

3.2 Heterogeneity of QoS reservation classes between receivers of the same session

DRP allows the sender to request, 'on-the fly', intserv's Controlled-Load Service (Wroclawski, 1997) and Guaranteed Service (Schenker, 1997) by sending a RES packet in-line with data. The sender designates a 'Ceiling' Reservation class (or Type), CRTs to each data flow block[7] as well as an associated end-to-end QoS level. In addition each receiver specifies a 'ceiling' reservation class, CRTr which represents the highest quality reservation class it is willing to receive. The Guaranteed Service reservation class is taken to be the highest quality reservation class, followed by Controlled-Load Service with 'best-effort'(no reservation) being the lowest. Assuming that sufficient end-to-end resources exist, the effective end-to-end reservation class received by a receiver will then be given by MIN(CRTs, CRTr). Each receiver is free to change its value of CRTr at any time by sending a

---

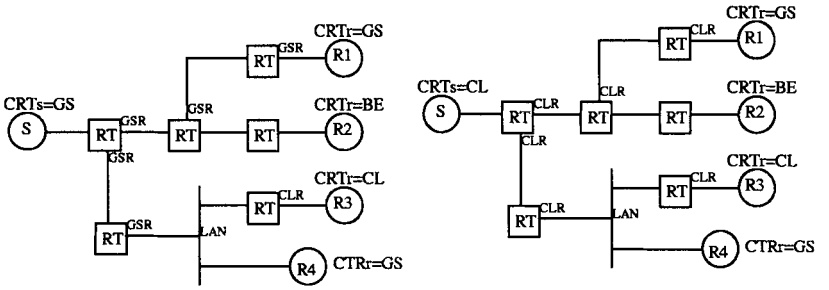
[5] Traditional ATM signalling(e,g, Q.2931 and UNI) requires end-to-end handshaking.

[6] In the case of Guaranteed Service, DRP may sometimes use feedback to alter certain reservations after the first pass in an attempt to achieve a target end to end delay bound that was not satisfied on the first pass as described in section 3.6.

[7] A data flow defined by the combination of (sender IP address, sender port, destination IP address, destination port, transport layer protocol) can be considered as a series of data flow blocks, each of which may have its own specific QoS requirements.

RTN packet upstream containing the new value of CRTr. In addition, the reservation class installed at each on-tree outgoing interface will be the lowest quality reservation class that is necessary to guarantee each receiver their effective end-to-end reservation class as determined by the above rules. This is exemplified in Figure 1.

BE	Best-effort	CRTs	Ceiling reservation type(sender)
CL	Controlled-load	CRTr	Ceiling reservation type(receiver)
CLR	Controlled-load Reservation	LAN	Local Area Network
GS	Guaranteed Service	S	Sender
GSR	Guaranteed service Reservation	Rn	Receiver
		RT	Router



Example a: CRTs=GS **Example b: CRTs=CL**

Figure 1: Heterogeneity of reservation classes between receivers of same session.

Table 1 compares the DRP approach to providing end-to-end delay bounds with those of RSVP and ABT/IT. DRP is similar to ABT/IT in the sense that for a given sender to a multicast session the target end-to-end delay bounds will be identical for each receiver[8]. The difference is that with DRP the sender can control what this delay bound will be whereas with ABT/IT the delay bound is a feature of the QoS class provided by the network and cannot be controlled by end nodes. Like DRP, RSVP facilitates end node control of end-to-end delay albeit by receivers rather that senders. RSVP allows receivers finely grained control within a reservation class at the expense of added receiver complexity together with lack of support for reservation class heterogeneity among receivers of a session. By contrast, DRP supports such reservation class heterogeneity in that the sender suggests a reservation class and QoS level for all receivers who then have the option of downgrading QoS class. Although DRP does not offer receivers any control over QoS level within a class, we do not believe such a feature is necessary anyway and certainly not with regard to end-to-end delay bound. We argue that any such end-to-end delay bound is determined by the nature of the sender's traffic stream and as

[8] In the case of DRP we are only referring to those receivers that have actually requested an end-to-end delay bound, i.e. those with CRTr=GS.

such the sending application is the node most qualified to specify what it should be. The receivers simply need to be told what this end-to-end delay bound is so that they can set their playout buffers accordingly. Furthermore, removing receiver-control of end-to-end delay in DRP enables merging of RTN messages and RTN state in routers to ensure scalability to large multicast sessions as described in section 3.4.

	RSVP	ABT/IT	DRP
Sender control of delay bound	**No**	**No**	**Yes**
Receiver control of delay bound	**Yes**	**No**	**No**

Table 1: Comparison of different schemes with regard to provision of end-to-end delay bounds

3.3 Reservation request admission control.

Apart from the initiator of QoS requests(sender vs receiver) there are two other notable differences between the reservation mechanisms used by RSVP and DRP.

1. Explicit vs implicit reservation requests.

With RSVP, for both Controlled-Load and Guaranteed Service reservations, the request explicitly informs the node of the level of resources to reserve. The same can also be said of a Controlled-Load Service reservation request using DRP. However with a DRP Guaranteed Service request the RES packet requests the level of resources to reserve implicitly by informing the router of the accumulated delay bound thus far, together with the target delay bound and the sender traffic characteristics. Using this information along with path information obtained from RTN packets each router is able to estimate the local reservation required and update the accumulated delay bound in the RES packet accordingly. Each router calculates a local reservation bandwidth which, if also reserved in each subsequent router, will lead to an overall delay bound equal to the target delay bound. However, should any router have insufficient resources to install the calculated local reservation bandwidth then it reserves the most that it can and the attempt is only referred to as a 'reservation failure' if the resultant accumulated delay thus far exceeds the target delay bound. If the attempt is not a so-called 'reservation failure' then the RES message is treated the same regardless of whether the level of local reservation initially calculated could be reserved or it couldn't. This is because even in the latter case the target end-to-end delay bound may still be met since each subsequent router will automatically attempt to reserve more in order to compensate. The action taken in the event of a so-called 'reservation-failure' is discussed next.

2. Action in event of reservation failure.

With RSVP, any request that fails admission control at a router is not propagated any further along its path towards the sender(s) and a ResvErr message is sent to

affected receiver(s). By contrast, whenever a DRP node cannot satisfy the calculated local reservation, and the maximum level of resources that it can reserve is so low that it prevents the target end-to-end QoS from being satisfied, the request is not rejected. Instead, the node reserves as much resources as possible and sets specific QoS violation bits in the RES header while updating the other header fields in the usual manner before propagating the RES message down the distribution tree.

In the event of a DRP Controlled-Load Service 'reservation-failure', the node sets a bit, known as the QoSvoid bit, to 1. The RES packet is handled in the usual way by all subsequent routers encountered, although the presence of the non-zero QoSvoid bit will be an indication to receivers that the end-to-end QoS could not be achieved.

In the event of a Guaranteed Service request where the node could not reserve more than the mean rate of the sender's traffic, it becomes impossible to guarantee either lossless transmission or conformance to the target delay bound, or even the Controlled-Load Service. Consequently three flags should be set in the RES packet, namely the delayvoid, lossvoid and QoSvoid bits. When downstream routers see a RES packet with (CRTs=GS, delayvoid=lossvoid=1) then they take the 'effective CRTs' to be Controlled-Load Service(CL) and attempt to install a Controlled Load Service Reservation.

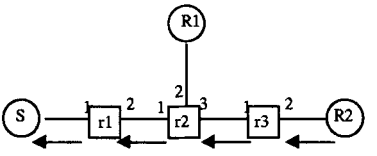
In the event of a Guaranteed Service request where lossless transmission has not yet been precluded [9] but the accumulated bound at a node exceeds the target delay bound for the first time, the action taken is as described in section 3.6.

In the case of Guaranteed Service reservations in a large multicast tree, there are some interesting differences between DRP's sender-based reservations and RSVP's receiver based reservations. To illustrate these differences we refer to the example topology of Figure 2. This shows the logical connectivity of a multicast session between a sender, S and two receivers, R1 and R2. These end nodes are interconnected via routers, r1-r3 and all links are 10Mbps Ethernet. The exported C and D error terms (Schenker, 1997) from the routers are shown together with the token bucket parameters of the Sender Tspec. We will assume that both receivers, R1 and R2 require a queuing delay bound of 300ms to sender S. With RSVP , each receiver calculates an Rspec that to be reserved in each router along the end-to-end path in order to achieve its delay bound. In this example, R1 calculates an Rspec of 325.3Kbytes/s while R2 calculates an Rspec of 490.89 Kbytes/s. At router r2 these two requests are merged so that the Rspec propagated to router r1 is 490.89Kbytes/s. Packets from S to receiver R1 will now experience a reservation bandwidth of 490.89kbyte/s in router r1, interface 2 rather than the requested 325.3 Kbytes/s. This will cause a reduction in R1's end-to-end delay meaning that theoretically the bandwidth reserved for R1 in r2, interface 2 could be decreased from the initially calculated value of 325.3 Kbytes/s while still achieving R1's end-to-end delay bound. However R1 does not facilitate such a mechanism and in this example R1's end-to-end delay bound will be less than, rather than equal to, that requested. By contrast, DRP keeps a running total of end-to-end delay bound

[9] That is, every node so far has been able to reserve in excess of the mean rate of the sender's traffic

which it updates at each hop and uses to calculate the local reservation required to stay on course for the desired end-to-end delay bound. As a result, in this example DRP automatically readjusts the reservation level in r2, interface 2 where 247.7 Kbytes/s is then reserved rather than 325.3 Kbyte/s as in RSVP.

For multicast examples such as this where the routers and links are homogeneous(same values of C, D error terms and link propagation delay) RSVP will never use fewer resources than DRP. However in environments with heterogeneous routers and links the matter is not as straightforward. With DRP, in a multicast environment the bandwidth to be reserved at each node is calculated based on, among other things, worst-case merged(see section 5.3)path characteristics received from RTN messages. The effect of this worst-case merging can be for DRP to make an over-estimation in the local reservation. Any such over-estimation will cause a reduction in the local node queuing delay. In turn this will mean that DRP allows an increase in the local queuing delay at nodes further downstream whose reservations will then not be as high. This 'skewing' of the bandwidth reservation pattern in multicast sessions whereby nodes closer to the sender are more likely to over-estimate their local reservations can theoretically cause an increase in the overall reservation bandwidth required in the multicast tree. This is an area for further study.



Token bucket parameters	
Peak rate(p) Bytes/s	200K
Token bucket rate(r) Bytes/s	50K
Token bucket depth(b) bytes	1K
Router error terms	
C(bytes)	48,000
D(μs)	1200
Link propagation delay	0

interface	Rspec reserved(Kbytes/s)	
	RSVP	DRP
r1 interface 2	490.89	490.89
r2 interface 2	325.3	247.7
r2 interface 3	490.89	490.89
r3 interface 2	490.89	490.89

Figure 2: Guaranteed Service Reservations using RSVP and DRP

3.4 Merging of RTN messages

DRP uses RTN messages which are reverse-routed up the distribution tree from receiver(s) to sender(s) for the following purposes:

1. To accumulate certain path characteristics information which is used by a node when calculating the level of resources to reserve.
2. To allow a receiver to downgrade its received reservation class below that suggested by the sender.
3. Optional feedback information that may be used to convey information to intermediate routers in cases where the end-to-end delay bound was not satisfied in the first pass of a RES message.

With respect to 1., RTN messages fulfil a similar role to Path messages in RSVP.

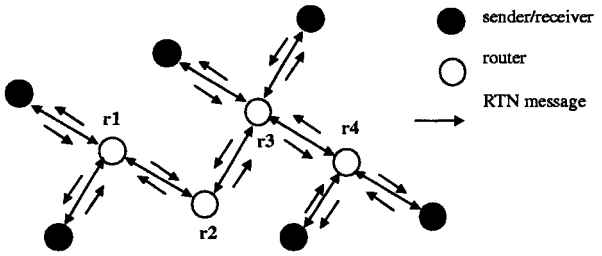


Figure 3:DRP RTN messages on shared tree per refresh period in the steady state.

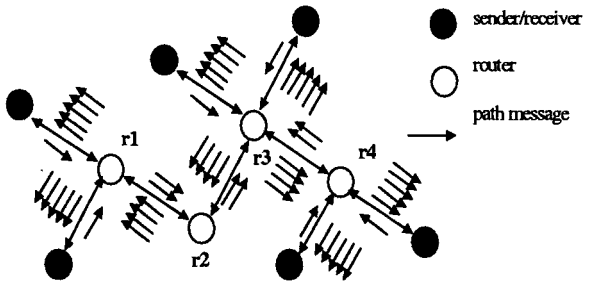


Figure 4: RSVP Path messages on shared tree per refresh period in the steady state.

For large-scale multipoint-to-multipoint applications the use of a single shared tree, such as a Core Based Tree(CBT), for all senders to a multicast group will consume far less resources (Billhartz, 1997) than a separate source-based tree for each sender to the group. Because of this, a single shared tree is likely to be preferable to a mesh of source-based trees in such scenarios. In such cases DRP displays much more favourable scalability characteristics than RSVP. With DRP, full merging of RTN messages is possible and ensures that the number of RTN messages on each link of a shared tree in the steady state is never more than two(one in each direction) every refresh interval as shown in Figure 3. By contrast, with RSVP the total number of Path messages on each link of a shared tree per refresh period in the steady state is equal to the number of senders as shown in Figure 4. However perhaps a more important benefit of the DRP approach is the fact that the number of RTN state entries in each on-tree router is equal to the number of on-tree logical interfaces and so never becomes an issue no matter how many hosts are sending to the group. By contrast, with RSVP the number of Path state entries in each router and end-host of a multicast shared-tree is equal to the number of senders to the group and consequently may become excessive for large-scale multipoint-multipoint applications.

3.5 Shared-Session Reservations and Intra-Session Reservation Style Heterogeneity

RSVP supports shared-style reservations which match on multiple senders and are usually used on the understanding that only one sender will be active at once. Although this will yield resource savings compared to a number of sender-specific reservations, shared-style reservations that are set up using RSVP can still be sub-optimal for 2 reasons. First, the reservation stays in place during quiet periods. Second, during active periods the reservation may sometimes or always be larger than necessary to meet the agreed QoS for the data flow currently using it (White, 1998). Both of these inefficiencies are obviated with DRP if the shared-session[10] is handled using sender-specific reservations that are installed and torn down on-the-fly at the start and end of each activity burst. However, in such cases it may be possible for end users to detect a degradation in QoS at the start of each activity burst due to the finite time required to install the 'just-in-time' sender-specific reservation. One way in which such QoS disruption could be minimised is to use what we refer to as a 'simple shared reservation' which would apply to all senders to the session and would be left in place during quiet periods but modified at the start of an activity burst each time the sender to the session changed. With this approach QoS disruption would only occur at the start of an activity burst for a new sender whose reservation requirement was greater than that of the previous sender, but even in this case the QoS disruption would be minimised because of the presence of the now free reservation from the previous sender that can be used as a starting point for the 'just-in-time' reservation request from the new sender to build upon.

While the 'simple shared reservation' mechanism just described would work well in the true 'shared-session' case where there is never more than a single active sender at any one time, it would suffer from under or over-reservations[11] in cases where it is possible for multiple senders to be simultaneously active which might occur in the absence of appropriate conference control mechanisms. This deficiency of a simple shared reservation approach is highlighted in Figure 6 for the example traffic pattern of Figure 5. Bearing these potential hazards in mind, DRP provides an alternative reservation mode to the standard sender-specific(SS) mode known as Sender-Specific with Residue(SSR). In SSR mode each sender makes a reservation at the start of each activity burst and sends a teardown request at the end of the activity burst. When a sender's teardown request[12] reaches an outgoing interface of a router the SSR reservation of the sender will only be removed if at least one other SSR reservation for the session is in place in the

---

[10] where only one sender to the session transmits at once.

[11] That is, over-reservations in addition to the 'over-reservations' present when the reservation stays in place during quiet periods.

[12] A teardown request is simply a RES packet with the reservation level set to 0. It indicates to the intermediate routers that the sender no longer requires a reservation for its data packets.

router at that outgoing interface. Otherwise the sender's SSR reservation is left in place, but a status flag associated with the reservation is set from 'active' to 'passive' state.

Figure 6 illustrates the operation of SSR mode for the traffic pattern of Figure 5. When only one sender is active at once, the operation of SSR mode is essentially the same as with a 'simple shared reservation' and so will suffer from resource wasteage when all of the senders go simultaneously quiet[13]. However should the senders go simultaneously quiet for extended periods of time the soft-state nature of the reservation will cause it to eventually timeout and be removed. In cases where more than one sender is simultaneously active, the operation of SSR mode is essentially the same as SS mode and so cannot suffer from the under-reservation problem that exists with the 'simple shared reservation'.

A notable advantage of DRP compared to RSVP is that DRP allows co-existence of both modes of reservations within the same multicast session while with RSVP each receiver within a given multicast session must choose the same reservation style. The way in which co-existence of reservation modes within the same multicast session is accommodated in DRP is summarised as follows.

When a reservation request arrives at an on-tree incoming router interface it is copied to each on-tree outgoing interface where the following steps are applied:

If reservation for that sender already exists

- Set reservations's mode flag(0=SS, 1=SSR) according to mode field in RES packet.
- Adjust reservation level to value indicated in RES packet.
- Set reservation's status flag to 1 (active).

Else if mode field in RES packet indicates SS

- Create a new reservation according to filter spec and value indicated in RES packet.
- Set reservation's mode flag to indicate SS.
- Set reservation's status flag to 1 (active).

Else if a SSR reservation exists with state = 'passive'

- Set filter spec of that reservation to the sender of the RES packet.
- Adjust reservation level to value indicated in RES packet..
- Set reservation's status flag to 1 (active).

Else

- Create a new reservation according to filter spec and value indicated in RES packet.
- Set reservation's mode flag to indicate SSR.
- Set reservation's status flag to 1 (active).

---

[13] For example in a multimedia conference if the audio channel used a different multicast group to the other multimedia traffic components there might be significant periods of time where the audio channel was quiet. By contrast in an audio-only conference the channel is unlikely to be quiet for any lengthy period of time.

When a reservation teardown arrives at an on-tree incoming router interface it is copied to each on-tree outgoing interface where the following steps are applied:

If reservation mode is SS

- Remove reservation

Else if total number of installed SSR reservations including this one is greater than one

- Remove reservation.

Else

- set reservation's status flag to 0(passive)

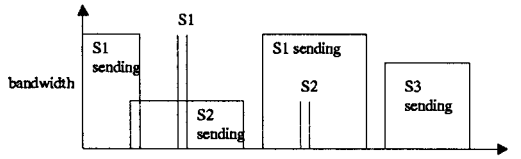


Figure 5: Traffic Pattern for a shared session with some sender transmission overlap

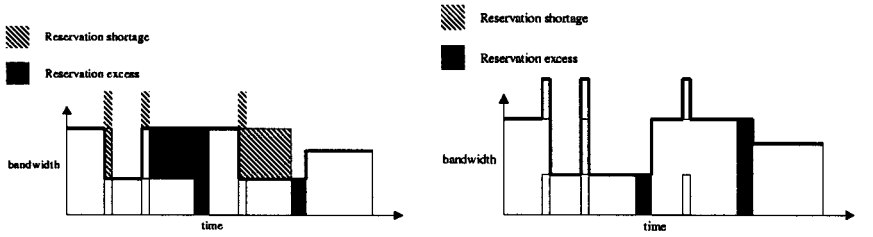


Figure 6: Bandwidth reserved for a shared session with some sender transmission overlap.

3.6 Using feedback to increase the probability of achieving end-to-end delay bound

In the case of Guaranteed Service reservations the adoption of a strategy whereby an end-to-end reservation is only permissible by installing an equal reservation in each router reduces the chances of meeting a target end-to-end delay bound. This characteristic has been noted by the designers of Guaranteed Service and exploited to a reasonable degree through the introduction of a slack term (White, 1997) into the reservation flow specification. Use of the slack term enables higher reservations to be made between the receiver and the bottleneck router to compensate for the increase in delay incurred by the lower reservation in all routers between, and including, the bottleneck router and the sender. However it does not permit the reservation to be increased once it has passed through the bottleneck

router on its way towards the sender. Such a restriction can sometimes prevent RSVP from achieving the target delay bound even on a path that actually contains enough resources to meet the target end-to-end delay bound.

Unlike RSVP, DRP employs cooperation and feedback between routers to ensure that if a given end-to-end path is capable of supporting a specific target delay bound then DRP will always meet the target delay bound. In the example of Figure 7 each router is able to reserve a bandwidth in excess of the mean rate of the sender's traffic but at router r3, accumulated delay for the first time is in excess of the target delay bound, Dt. Consequently, router r3 sets a bottleneck flag associated with its local reservation as well as setting bottleneck and delayvoid flags in the forwarded RES message. When r4 receives the RES message it notices that the delayvoid flag has been set to 1 and so as a result reserves the maximum reservation that it can(subject to any installed policy decisions) in order to minimise its contribution to the accumulated delay. When R receives the RES message it notices that the target delay bound has been exceeded and immediately issues a RTN packet containing the amount by which the target delay has been exceeded in a field in the packet known as the excess delay field. In addition, a bit in the packet known as the bottleneck bit, is set to 0. This RTN packet is reverse-routed up the tree but is ignored at each router until its bottleneck flag has been set to 1 which will occur when it reaches the interface of a router, r3 in this example, in which the bottleneck flag has been set to 1 for the installed reservation. The RTN packet will then travel hop by hop towards S with an attempt being made at each hop to eliminate the excess delay or at least reduce it as much as possible by increasing the level of the local reservation on the appropriate outgoing interface. If a router succeeds in reducing the excess delay to zero then the RTN packet will cause no further alterations in local reservations on the rest of its journey towards S. In this example, r2 is able to increase its local reservation and cause a reduction in its local queueuing delay of del which is then subtracted from the excess delay field before sending the RTN packet to r1 which manages to increase its local reservation sufficiently to completely eliminate the excess delay. The target end-to-end delay bound has now been achieved.

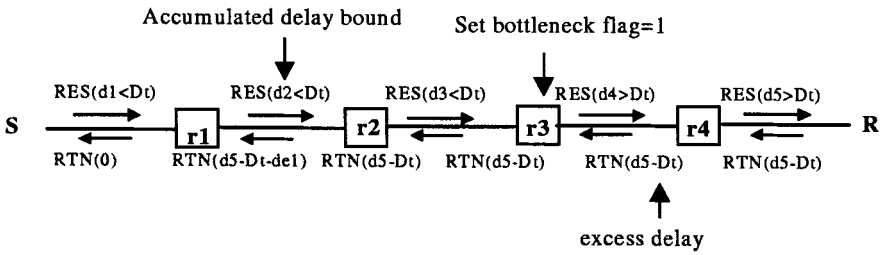


Figure 7: Use of DRP feedback mechanism to maximise chances of meeting target delay bound for Guaranteed Service.

In the next two sections we present details of the main fields required in RES and RTN packets together with the processing rules in order to provide the basic functionality of DRP described in the previous sections.

4 RESERVATION(RES) MESSAGE

The IP destination address of the IP datagram encapsulating a RES message is equal to the session destination address while the IP source address is equal to the initial sender of the RES packet. The IP router alert option is used to ensure that intermediate nodes intercept and process the RES packets.

4.1 RES message Common Part

- **Session** – (object defined in the RSVP protocol) – it contains the destination address, transport layer protocol identifier and transport layer destination port.
- **Phop** - (object defined in the RSVP protocol) – it is the identity of the last DRP-capable logical outgoing interface to forward this message. The Phop object consists of the pair (IP address, logical interface handle) and is required to install Phop state in the router to ensure correct reverse routing of RTN messages.
- **Sender Template** - (object defined in the RSVP protocol) – it is a filter specification identifying the sender. It contains the IP address of the sender and optionally the sender port(in the case of Ipv6 a flow label may be used in place of the sender port)
- **timestamp** field - this is stamped with the time of the local node clock just before being forwarded to next hop(s) down the distribution tree. It is used to calculate dnext as described in (White, 1998).
- **CRTs** field(2 bits) - this identifies the ceiling reservation class of the sender. 11 indicates Guaranteed Service, 10 indicates Controlled-Load Service, and 00 indicates best-effort. 01 is currently unspecified although may at some time be used for a new service with quality in between best-effort and Controlled-Load Service.
- **Tspec** describing sender's traffic characteristics using the following token bucket representation as described in (Schenker, 1997)
  p = peak rate of flow (bytes/second)
  b = bucket depth (bytes)
  r = token bucket rate (byes/second)
  m = minimum policed unit (bytes)
  M = maximum datagram size (bytes)
- **end2end delay** field - this gives the current delay from when a packet was transmitted by the initial sender until it is due to arrive at the incoming interface of the current next hop.
- **Mode** field(1 bit) – this identifies the reservation mode. A value of 0 indicates SS mode while a value of 1 indicates SSR mode.
- **QoSvoid** bit – if set to 1 this indicates that no QoS guarantees can be offered.

4.2 RES message Guaranteed Service object

If CRTs = 11(Guaranteed Service) the RES packet will also contain a Guaranteed Service object comprising the following:

- **CSum** - accumulation of C values since last upstream reshaping point (see (Schenker, 1997)).
- **DSum** - accumulation of D values since last upstream reshaping point (see (Schenker, 1997)).
- **target-bound** field which indicates the target end-to-end delay of the sending application.
- **accumulated-bound** field which indicates the installed delay bound between sender and the incoming interface of the current next hop.
- **Flags** field containing
  - **delayvoid bit** (If set, this bit is an indication to the receiver that the target delay bound cannot be guaranteed)
  - **lossvoid bit** (If set, this bit is an indication to the receiver that a loss-free service cannot be guaranteed)

4.3 Node Processing of RES messages

When a node receives a RES packet for an end-to-end reservation attempt at which QoS violation has already occurred or which occurs following processing of the RES packet, the behaviour of the node is as described in sections 3.3 and 3.6. Otherwise the processing of the packet is as described in the remainder of this section.

Upon receipt of the RES message the node passes it to admission control which then determines the reservation that needs to be made at each of the outgoing interfaces. The reservation class is given by MIN(CRTs, CRTr) where CRTr is obtained from the Merged RTN State Entry(MRTNSE) for the appropriate outgoing logical interface as described in the next section.

If the reservation request is for the Controlled-Load Service then the reservation is governed entirely by the sender Tspec contained within the RES message. By contrast if the reservation request is for Guaranteed Service then the reservation is described by the combination of the sender Tspec and a reservation bandwidth, R that the admission control mechanism needs to determine using the following equations as given in (Schenker, 1997).

$$Qdelay_{end2end} = \frac{(b-M)(p-R)}{R(p-r)} + \frac{(M+Ctot)}{R} + Dtot \quad (\text{case } p > R >= r). \quad (1)$$

$$Qdelay_{end2end} = \frac{(M+Ctot)}{R} + Dtot \qquad (\text{case } R >= p >= r). \qquad (2)$$

Admission control obtains the parameters M, p, b and r from the sender Tspec contained in the RES message. The value of Ctot is given by the sum of the router's local C value and the merged Ctot value as obtained from the MRTNSE(see next section) for the relevant outgoing interface. Likewise the value of Dtot in the above equations is given by the sum of the router's local D value and the Dtot value in the MRTNSE for the relevant outgoing interface. To obtain the value of Qdelay to insert into the above equations, admission control uses the relationship given in equation (3).

Qdelay = target-bound - accumulated-bound – dnext – propdelay. (3)

where the target-bound and accumulated-bound are obtained from the corresponding fields in the RES packet, and propdelay and dnext are obtained from the MRTNSE for the outgoing interface.

Once the resultant value of Qdelay has been substituted into equations (1) and (2) along with the other mentioned parameters, a value of R to be installed at the outgoing interface is obtained.

Regardless of the reservation class, that is Controlled-Load or Guaranteed Service, if processing of the RES does not result in either the installation of a new reservation or a modification of an existing reservation(i.e. the RES packet was simply a refresh) then the soft-state timer for the reservation is simply reset. Otherwise, the reservation request is propagated immediately down the distribution tree after updating the appropriate fields in the packets header as follows. The end2end delay field in the RES packet is increased by adding to it the following:

- The propagation delay, dnext for the next hop
- An estimate of the current local queuing delay(for the relevant outgoing interface) for data packets of the flow to which the RES packet refers.

In addition, if CRTs=11 the following updates must be made to the RES packet:

- Add the following to the accumulated-bound field of the copied packet.
  1. The propagation delay, dnext for the next hop
  2. The installed local queueing delay bound(for the relevant outgoing interface) for data packets of the flow to which the RES packet refers. This local queuing delay bound is obtained by inserting the reserved value of R into equation (4) along with the local values of C and D

$$Q_{local} = \frac{C}{R} + D. \qquad (4)$$

- If reshaping to the sender Tspec is being performed at the outgoing interface
  set Csum=Dsum=0.
  Else
  Add the local value of C to the CSum field
  Add the local value of D to the DSum field

Once updating of the fields is complete the timestamp field is now set equal to the local clock before forwarding the RES packet to each next hop down the routing tree.

5 RETURN(RTN) MESSAGE

The IP destination address of the IP datagram encapsulating an RTN message is equal to the IP address of a previous hop node, the identity of which is obtained from installed Phop state obtained from RES messages, while the IP source address is equal to the IP address of the node out of which the RTN message was sent.

5.1 Common part

- **Session** – as for RES message.
- **Nhop** - (object defined in the RSVP protocol) – the identity of the DRP-capable logical outgoing interface that sent this message. The Nhop object consists of the pair (IP address, logical interface handle)
- **Sender address** – the combination of this field and the session object identify a source-based tree. In the case of a shared tree this field is ignored and should be set to all 0's.
- **timestamp** - stamped with the time of the local node clock just before being sent to previous hop up the distribution tree. This is used in calculation of dnext as described in (White 1998).
- **CRTr** (2 bits) - indicates the receiver's ceiling reservation class.
- **timedelta** - used in calculation of dnext as described in (White 1998).
- **propdelay** - the data packet propagation delay along the maximum 'Total Rate-Independent Delay'(TRID) path[14] between the node incoming[15] interface out of which the RTN packet was sent and each receiver downstream.
- **pathMTU** - the minimum pathMTU value between the incoming interface out of which the RTN packet was sent and each receiver downstream of that incoming interface.
- **Ctot** - the maximum accumulated Ctot value along the paths between the incoming interface out of which the RTN packet was sent and each receiver downstream of that incoming interface. The C error term is defined in the Guaranteed Service specification (Schenker, 1997).

---

[14] Total Rate Independent Delay(TRID) is given by the sum of the link propagation delays and the D error terms.

[15] The term 'incoming' refers to the direction of data flow. RTN packets are reverse-routed up the distribution tree in the opposite direction to the data flow and so are always sent out of so-called incoming interfaces.

- **Dtot** – sum of D error terms along the maximum 'Total Rate-Independent Delay(TRID)' path[14] between the node incoming[16] interface out of which the RTN packet was sent and each receiver downstream. The D error term is defined in the Guaranteed Service specification (Schenker, 1997).
- **path bandwidth** - the maximum path bandwidth value along the paths between the incoming interface out of which the RTN packet was sent and each receiver downstream of that incoming interface.

5.2 RTN Guaranteed Service feedback object

The RTN packet may optionally contain a Guaranteed Service feedback object comprising:

- **excess delay field** – the amount by which the installed end-to-end delay bound currently exceeds the target end-to-end delay bound.
- **bottleneck flag** - if set to 1 this indicates that the RTN message has travelled at least as far as the the router where the accumulated delay-bound first exceeded the target delay-bound on the first pass of the RES message.
- **Sender Template** – same as that in RES packet whose end-to-end delay bound was exceeded.

5.3 RTN state and message merging rules

At an outgoing interface, i of a router on the distribution tree, reception of an RTN packet from a next hop, j results in the updating of any matching router state, known as an RTN state entry or $RTNSE_{ij}$, or the setup of new state if no match exists. There will be a separate $RTNSE_{ij}$ for each 4-tuple (Session, sender address, next hop, outgoing logical interface). The first three parameters of this 4-tuple are contained within the received RTN message while the outgoing logical interface(oif) is determined by the interface on which the RTN message arrived. In the case of a shared tree the sender address field will be omitted for the $RTNSE_{ij}$. The format of an $RTNSE_{ij}$(excluding any guaranteed service feedback parameters) is as shown in Table 2. In addition, for each outgoing logical interface, i a single Merged RTN State Entry ($MRTNSE_i$) is created from the set of entries $\{RTNSE_{ij}\}$ for that logical outgoing interface. There will be multiple $RTNSE_{ij}$s for a given logical outgoing interface if the logical outgoing interface has multiple next hops on the distribution tree which can occur if the logical outgoing interface connects to a shared medium LAN(e.g. Ethernet). The parameters of the $MRTNSE_i$ and how they are formed from $\{RTNSE_{ij}\}$ are also shown in Table 2. The 'merged values' of various parameters in each RTN message sent out of an incoming interface to a previous hop upstream are obtained from $\{MRTNSE_i\}$, the set of MRTNSE for the outgoing interfaces as shown in Table 2.

---

[16] The term 'incoming' refers to the direction of data flow. RTN packets are reverse-routed up the distribution tree in the opposite direction to the data flow and so are always sent out of so-called incoming interfaces.

$RTNSE_{ij}$	$MRTNSE_i$	*Merged RTN packet sent upstream out of interface k*
CRT_{ij}	CRT_i=MAX{CRT_{ij}}	CRT_k=MAX{CRT_i}
$Ctot_{ij}$	$Ctot_i$=MAX{$Ctot_{ij}$}	$Ctot_k$=MAX{$Ctot_i$+$Clocal_{ki}$}(footnote 17)
$Dtot_{ij}$	$Dtot_i$=$Dtot_{ij}$ Where j is such that $TRID_i$=$TRID_{ij}$	$Dtot_k$=$Dtot_i$+$Dlocal_{ki}$ such that i gives MAX{$Dlocal_{ki}$+$TRID_i$} for that interface k (footnote 17)
$Propdelay_{ij}$	$Propdelay_i$=$propdelay_{ij}$ Where j is such that $TRID_i$=$TRID_{ij}$	$Propdelay_k$=$propdelay_i$+$dnext_i$ such that i gives MAX{$Dlocal_{ki}$+$TRID_i$} for that interface k (footnote 17)
$PathBandwidth_{ij}$	$PathBandwidth_i$= MAX{$PathBandwidth_{ij}$}	$PathBandwidth_k$= MAX{MIN(pathbandwidth$_i$, link rate$_i$)}
$PathMTU_{ij}$	$PathMTU_i$= MIN{$pathMTU_{ij}$}	$PathMTU_k$= MIN{MIN(path MTU_i, linkMTU_i)}
$dnext_{ij}$	$dnext_i$=$dnext_{ij}$ where j is such that $TRID_i$=$TRID_{ij}$	
$TRID_{ij}$= $Dtot_{ij}$ +$propdelay_{ij}$ +$dnext_{ij}$	$TRID_i$=MAX{$TRID_{ij}$}	
sender template$_s$	*sender template$_s$*	*sender template$_s$*
excessDelay$_{sij}$	*excess delay$_{si}$=* *MAX{excessDelay$_{sij}$}*	*excess delay$_s$=* *MAX{excessDelay$_{si}$ – delayReduction$_{si}$}*
bottleneckFlag$_{sij}$	*bottleneckFlag$_{si}$=* *MAX{bottleneckFlag$_{sij}$}*	*bottleneck flag$_s$=MAX{bottleneck flag$_{si}$}*

Table 2: relationship between RTN state entries, MRTN state entries and merged RTN packets.

[17] $Clocal_{ki}$, $Dlocal_{ki}$ =router's value of C and D error terms between incoming interface k and outgoing interface i

The last three rows of Table 2 represent optional GS-feedback objects and are written in italics to differentiate them from the core entries shown in the table. Merging between GS-feedback object state only occurs if the objects relate to the same sender template, s. A merged GS-feedback object for sender template, s is only included in the merged RTN packet sent upstream if the RTN packet is addressed to Phop for sender template s as obtained from installed RES state. With regard to the excess delay entries shown in the table, $delayReduction_{si}$ refers to the local reservation queuing delay reduction achieved since the RES for sender s at interface i was installed.

If CRTr is not equal to GS in the propagated RTN message, the rules in Table 2 are overridden by setting Ctot=Dtot=propdelay=0 in order to ensure that only those links receiving Guaranteed Service are taken into account when conducting worst-case merging of GS-specific parameters.

Whenever the contents of a RTN message to be sent upstream differ from the preceding one, the RTN message is sent immediately. Otherwise, i.e. in the steady state, an RTN message is sent to a previous hop once per some refresh period.

6 SUMMARY

In this paper we have discussed the need for resource reservation in the Internet and examined the use of RSVP for this purpose while highlighting some of its favourable characteristics such as its use of 'soft-state' reservations. Consequently we acknowledge RSVP as a useful starting point in the design of alternative reservation protocols but we do not accept that it represents the ultimate solution because of certain deficiencies and restrictions that we demonstrated in the text. This has motivated our design of an alternative IP reservation protocol, DRP which incorporates many principles of RSVP together with the dynamic sender-initiated reservation concept of ABT/IT to achieve the following main goals:

1. High reservation control dynamics to achieve efficient bandwidth usage.
2. Scalability of router-state with regard to number of senders and receivers. The protocol is especially suited to large-scale-multicast applications where it can expect to achieve a router state saving of several orders of magnitude compared to RSVP.
3. Heterogeneity of QoS classes and reservation styles for nodes within a given multicast session.

Details of control messages were presented along with associated processing rules. Although in principle DRP offers considerable benefits over existing reservation protocols certain aspects of it are not well understood and further work is required especially in the following areas:

1. Reservation setup time for each of the different service classes.
2. Impact of SSR mode on reservation set up time compared to SS mode.
3. Effect of worst-case merging of OPWA data – For a large multicast tree this will tend to cause the nodes closest to the sender to over-estimate their local reservations which as a result causes a reduction in the local reservations downstream. Any implications of this phenomenom need to be clarified.

4. Investigation into alternative Guaranteed Service feedback techniques for the purpose of reducing the end-to-end delay bound when it is in excess of the target-delay bound after one pass of the RES packet. For example one alternative worth investigating is the generation of the RTN packet containing the Guaranteed Service feedback object as soon as the bottleneck node is encountered rather than waiting until the RES packet arrives at the receiver.

7 ACKNOWLEDGEMENTS

We would like to thank British Telecom Labs, Ipswich, England for supporting this work. In particular we are grateful to Alan O'Neill and Terry Hodgkinson of BT labs for many valuable discussions regarding the material presented in this paper.

8 REFERENCES

Almesberger, W, Boudec, J and Ferrari, T. (1997) Scalable Resource Reservation for the Internet, EPFL DI-LRC, CH-105 Lausanne, Switzerland.

ATM Forum (1996). ATM User Network Interface (UNI) Specification Version 4.0. AF-UNI-4.0.

Billhartz, T., Cain, J., Farrey-Goudreau, E., Fieg, D. and Batsell, S. (1997) Performance and Resource Cost Comparisons for the CBT and PIM Multicast Routing Protocols in DIS Environments, IEEE Journal of Selected Areas in Communications, April 1997. http://www.epm.ornl.gov/~sgb/pubs.html.

Braden, R., Zhang, L., Berson, S., Herzog, S. and Jamin, S. (1996) Resource Reservation Protocol (RSVP) - Version 1 Functional Specification, August 12, 1996.

Delgrossi, L. and Berger, L. (1995) Internet Stream Protocol Version 2 (ST2) Protocol Specification - Version ST2+, August 1995, RFC1819.

Forgie, J., (1979) "ST - A Proposed Internet Stream Protocol", IEN 119, M.I.T. Lincoln Laboratory, 7 September 1979.

intserv, IETF, Integrated Services Charter (1998) http://www.ietf.org/html.charters/intserv-charter.html

issl, IETF Integrated Services over Specific Link Layers Charter (1998) http://www.ietf.org/html.charters/issll-charter.html.

ITU-T (1995). Q.2931: Broadband Integrated Services Digital Network (B-ISDN); Digital Subscriber Signalling System No. 2 (DSS2); User-Network Interface (UNI) Layer 3 Specification for Basic Call/Connection Control.

ITU-T (1996) Recommendation I.371. Traffic Control and Congestion Control in B-ISDN, (08/96).

Schenker, S., C.Partridge, R.Guerin. (1997) Specification of Guaranteed Quality of Service, Request for Comments, September 1997, RFC2212.

Seidensticker, S., Smith, W. and Myjack, M. (1997) Scenarios and Appropriate Protocols for Distributed Interactive Simulation, Internet Draft, March 1997, draft-ietf-lsma-scenarios-01.txt.

Topolcic, C. (1990) Experimental Internet Stream Protocol, Version 2 (ST-II), October 1990, RFC1190.

White, P. (1997) RSVP and Integrated Services in the Internet: a tutorial, IEEE Communications magazine, May 1997.

White, P. (1998) A case for Dynamic Sender Based Reservations in the Internet. UCL report. ftp://cs.ucl.ac.uk/darpa/SenRes.ps.Z.

Wroclawski, J. (1997) Specification of the Controlled-Load Network Element Service, Request for Comments, September 1997, RFC2211.

9 BIOGRAPHY

Paul White was awarded a BEng degree(First Class Honours) in Electronic and Electrical Engineering at the University of Birmingham, England in 1989. From then until 1994 he worked in various fields of hardware/software technology including telecommunications and became a Chartered Electrical Engineer. In 1994 he returned to academia and was awarded an MSc degree(with Distinction) in Data Telecommunications and Networks at the University of Salford, England in 1995. He has been working towards a PhD at University College London since 1995 and is supported by British Telecom Labs, Ipswich, England.