Measurement-Based Admission Control for Dynamic Multicast Groups in Diff-Serv Networks*

Elena Pagani and Gian Paolo Rossi
Dip. di Scienze dell'Informazione, Università degli Studi di Milano
via Comelico 39, I-20135 Milano, Italy
email: {pagani,rossi}@dsi.unimi.it

Abstract

An appealing approach to the admission control problem for traffic with QoS requirements consists in evaluating the resource availability by means of *measurement-based* techniques. Those techniques allow to provide QoS with minimal changes to the current network devices. Two techniques have been proposed in the literature: either *passive* or *active* measurements. The latter technique, although more intrusive, guarantees more accurate estimates. Yet, even though many of the applications requiring QoS are multicast, in our knowledge no solution has been so far proposed to extend the active measurement approach to the multicast case. The intrinsic problems here are to perform the admission control to convey the traffic to recipients that dynamically join during the transmission, and to merge the results of the measurements for different recipients.

In this work, we propose a mechanism to perform (active) measurement-based admission control for multicast groups with dynamically joining receivers. The proposed mechanism has been implemented in the NS-2 simulation framework, to evaluate its performance. For a simple system setting, we also obtain a lower bound on the length of the measurement phase, that must be met to guarantee the measure accuracy.

Keywords: Quality-of-Service; multicast; bandwidth broker; active measurement; performance evaluation

1 Introduction

One of the main problems in providing Quality-of-Service (*QoS*) support to multimedia and real-time applications in the current Internet concerns the set-up of the data transmission service. The service set-up requires that traffic admission control is performed, by evaluating the resource availability along the path from the traffic source to the destination(s), and that the routers along such a path are configured to provide QoS guarantees as requested by the

*This work was supported by the MURST under Contract no.MM09265173 "Techniques for end-to-end Quality-of-Service control in multi-domain IP networks".

applications. The *Differentiated Service* model [3] has been proposed in the literature to provide QoS in a scalable manner. According to that model, *bandwidth broker* agents [17] exist that take in charge the traffic admission control functionalities. Yet, only a few practical implementations of the diff-serv model have been realized, and the approach is still far from being standardized. Moreover, although many multimedia applications are characterized by a multicast communication pattern, the diff-serv model has intrinsic problems in supporting multicast [3]. The main aspects that must be dealt with concern the dynamic changes in the multicast group membership and the heterogeneity among the QoS levels perceived by different receivers. For what concerns the former issue, it is difficult to perform the resource reservation needed by recipients that join the group when the data transmission is ongoing. The latter issue requires the capability of managing service acceptances and service refusals generated by different subsets of receivers.

In this paper we describe the end-to-end *Call Admission Multicast Protocol* (CAMP) [18], that can be used to ensure bandwidth guarantees to multicast sessions in IP networks, thus providing them with the Premium Service [17]. CAMP is scalable, operates on a per-call basis and supports the group membership dynamics. It represents a possible implementation of a distributed bandwidth broker. To perform the admission control, CAMP adopts an active-measurement approach. We have implemented CAMP in the frame of the NS-2 simulation package to verify its effectiveness under different system conditions. We have obtained a lower bound on the length of the measurement phase, that guarantees an accurate measure of the resource availability; the result applies to the unicast case as well.

The paper is organized as follows: in Section 2, we introduce the system model considered throughout the work and the adopted notation, and we discuss related works presented in the literature. In Section 3, we describe the CAMP protocol, focusing on the mechanisms adopted to deal with dynamically joining recipients. In Section 4, we discuss some issues related with the measurement-based approach, and we obtain a lower bound on the measurement phase length. In Section 5, we present the experimental results achieved by implementing CAMP in the NS-2 simulation framework. Section 6 concludes the work.

2 System model

A QoS-sensitive application specifies to the underlying service provider, the QoS communication requirements and the behaviors of the data flow it is going to generate (*traffic profile*).

QoS can be provided according to two service models: the *integrated services* (int-serv, IS) [4] and *differentiated services* (diff-serv, DS) [3]. Int-serv's offer QoS on a per-flow basis. The data packets are labeled with the identifier of the flow to which they belong. By contrast, diff-serv's offer QoS on a per-packet basis. According to this model, each DS domain offers a specified set of service classes. Each packet, through the use of the TOS/*DS* byte field of the IP header [16], is labeled with the identifier of the class (*DS codepoint*) to which it belongs. Resource reservation is performed on the base of the different service classes. All the packets which belong to a given service class C are subject to the same forwarding policy (PHB); they constitute a *behaviour aggregate*.

Packets are forwarded through an interconnected system, composed of a set of networks connected by routers. Routers allow the forwarding of messages through different networks

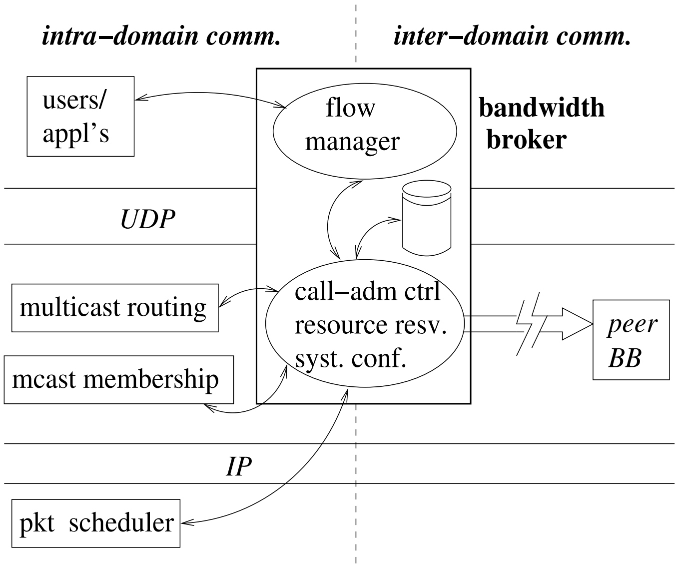


Figure 1: Framework architecture

and are assumed to implement the QoS services. A *domain* is a set of contiguous networks which operate with a common set of service differentiation models, provisioning policies and service definition. We denote as *boundary routers* those routers that connect contiguous domains.

In [17] an architecture is proposed to provide QoS according with the diff-serv model, that exploits so called *bandwidth broker* (BB) agents to perform the functionalities of call admission control, resource reservation and system configuration. A BB exists for each domain. The BBs span the end-to-end and network layers in the OSI model. In this work, we propose an admission control mechanism that can be adopted to realize a distributed BB entity. In Figure 1, we show the functional architecture that we consider in this work, with the BB internal structure in accordance with the proposal presented in [23]. The BB provides the applications with the interface to access the QoS services. It may exploit the services of both a membership protocol, and the multicast routing protocol to acquire knowledge about the network topology. When the QoS aggregate spans multiple domains, an inter-domain protocol is executed amongst peer BBs, to guarantee the proper configuration of the transit and destination domains.

We assume that a session announcement protocol (e.g., `sdr` [11]) is used to announce the needed session information, such as the transmission start time, the generated traffic profile and the multicast address of the destination group. We consider sources generating CBR traffic. All the recipients receive the same set of microflows; they are *homogeneous*, that is, they have the same QoS requirements.

2.1 Related works

So far, very few proposals have been presented about the BB implementation. The QBone testbed [23], under development by the Internet2 QoS Working Group, involves a BB that is initially "implemented" by a human operator who manually configures the system. Scenarios have been figured out to design an inter-BB communication protocol (SIBBS) [23]. No standard technique still exists to automatically perform intra-domain configuration.

In this paper, we discuss a mechanism to implement the resource reservation, call-admission control and system configuration functionalities both at intra-domain and inter-domain level, which adopt a distributed approach. We propose a measurement-based technique that tests for resource availability without requiring state information maintenance in the routers. Several solutions have been recently proposed to perform admission control, based on either *passive* or *active* measurements [19]. In the former case [12, 7], routers monitor the passing traffic. When they receive a service set-up request, they decide about the admission of that service basing on the gathered estimates about the current resource usage. This technique offers lower precision than the active measurement approach in the estimation of the available resources; yet, it is less intrusive. With the active measurement approach, *probing* traffic is generated, having the same profile as the data traffic. Admission is decided by evaluating the quality with which the probing traffic is received at the destination. Solutions that adopt the active measurement technique are for instance those presented in [14, 2, 8]. In [14] that technique is used to estimate the capacity of the bottleneck link along a path. The PCP protocol [2] performs admission control and bandwidth reservation for point-to-point voice sessions. In [8], the authors evaluate the suitability of that approach to implement admission control for the *Controlled-Load* service [25] in IS systems. All those solutions do not involve mechanisms to deal with multicast communications, differently from the protocol we propose in section 3.

3 Distributed Bandwidth Broker

In this section we describe the end-to-end *Call Admission Multicast Protocol* (CAMP) [18]. CAMP can be used to ensure bandwidth guarantees to multicast sessions in IP networks, thus providing them with the *Premium Service* [17]. CAMP supports dynamic changes in the multicast group membership. It performs the functionalities of a distributed BB and operates on-demand.

In section 3.1, we outline how CAMP works in the base case. In section 3.2, we focus on the issues concerning the management of the concurrency among dynamically joining recipients.

3.1 The CAMP algorithm

In Figure 2, we show the system architecture in which CAMP works. CAMP operates within the RTP/RTCP [21] protocol suite and performs the set-up of a RTP session. It receives from the application, via RTP, the profile of the data traffic that will be generated (specified for instance as a `TSpec` [22]). CAMP uses RTCP to monitor the QoS supplied to the recipients. CAMP performs the admission control using an *active measurement* approach [19, 5]. It generates *probing* traffic with the same profile as the data traffic generated by the application. Both the data and the `probe` packets are multicast using the UDP and IP protocols. To this aim, we assume that both a membership protocol (e.g., IGMP [6]) and a multicast routing protocol (e.g., PIM [9], MOSPF [15]) are available. The latter maintains a tree-

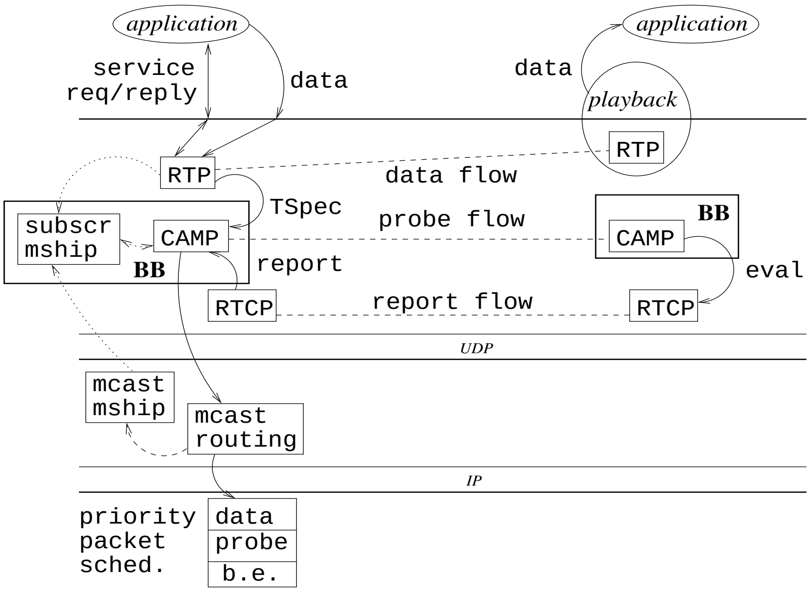


Figure 2: Architecture of CAMP-based end stations

based routing infrastructure connecting the multicast recipients. CAMP is independent of both those protocols. The *probing* phase has the aim of evaluating whether the available bandwidth is sufficient to support the new traffic. With respect to the classification given in [5], we adopt *out-of-band* probing with *dropping* of the `probe` packets as the congestion signal. This means that all the routers use a *priority* packet scheduling discipline: the `probe` packets are marked with a higher priority than the best effort packets, but a lower priority than the QoS packets. This priority assignement ensures that the probing traffic does not affect the established QoS sessions. On the other hand, `probe` packets can drain the available bandwidth for the new QoS session at the expenses of the best effort traffic.

To support multicast, two issues must be considered: (*i*) the receiver group membership can dynamically change; and (*ii*) different destinations can experiment different QoS in receiving the same traffic. We cope with problem (*ii*) above as follows: the source CAMP diffuses the generated traffic profile via its own RTCP reports. Each receiver r matches that profile against the received probing traffic profile and decides whether to accept the *whole* session or not. In both cases, r informs the source about its decision with its own RTCP reports. If r refuses the service, it unsubscribes from the group and prunes from the tree *before* sending its refusing report. When all the reports have been received by the source, if the service is accepted by at least one recipient, the source switches from the transmission of `probe` packets to the transmission of the data packets generated by the application, *without discontinuity*. The data packets are forwarded along the pruned tree.

For what concerns the problem (*i*) above: the effectiveness of the admission control requires that the data traffic is forwarded along a subtree only after that all the recipients grafted to that subtree have completed the probing phase by either accepting the service or pruning from the tree. Let us consider the example shown in Figure 3, in which a receiver grafts the tree at the in-tree node g, when the data transmission for the session 1 is already ongoing. The data packet cannot be immediately forwarded along the new branch. Indeed,

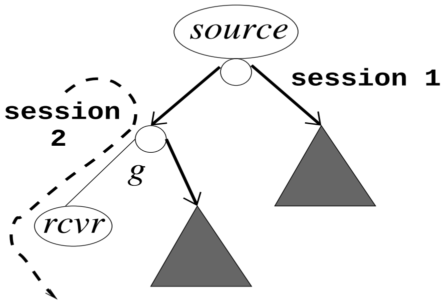


Figure 3: Example of concurrent sessions

in the event there is not enough available bandwidth, other established sessions traversing the same links (as session 2 in the example, whose packets are aggregated to the packets of session 1) would be negatively affected as their packets could be dropped. On the other hand, the admission control must be performed only on the new branch, to avoid the duplication of the resource reservation for the session 1 already installed on the path from the source to g.

We deal with those problems using a *proxy* mechanism. The source announces the multicast session via `sdr` and starts transmitting `probe` packets at the scheduled time, if at least one receiver is listening. A Camp proxy is instantiated in a router, when the router is an in-tree node and either:

1. it receives a `probe` packet for the group (source initialization phase); or
2. it notices the appearance of a new downstream interface for the group and it receives a data packet for that group (dynamic membership change).

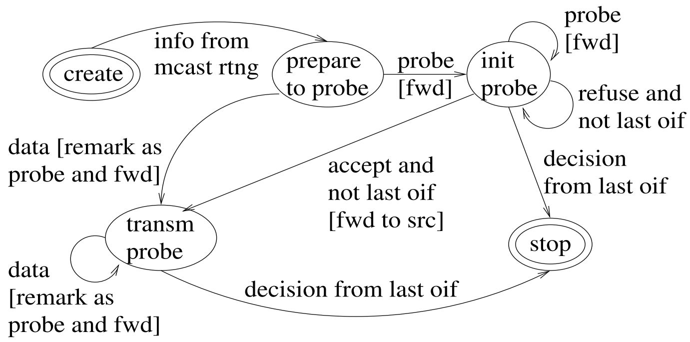
The proxy lifetime lasts until the set-up phase terminates for all the downstream destinations. The proxy uses the information in the local routing table to maintain a data structure (figure 4(a)) containing the list of downstream multicast interfaces (*oif*s)[1], the list of destinations for each *oif* and the *oif* state. Initially, all the *oif*s in the table are in the *probing* state. During its activity, the proxy remarks as `probe` packets all the incoming packets for the session, that must be forwarded to the probing *oif*s. The proxy labels the forwarded packets with its own address. The receivers send their RTCP reports to their proxies.

When a proxy receives an accepting report, and it is receiving `probe` packets from its parent, it knows that the source is in the initialization phase and it forwards that report to the source. When a report has been received from each destination downstream a *probing oif*, and all the reports are refusals, the *oif* is removed from the table. If at least an accepting report exists, the proxy changes the state of the corresponding *oif* to *data* and stops the packet remarking for that *oif*. If new downstream interfaces appear while the proxy is active, they are added to the proxy table. When no *oif* in the table is anymore marked as *probing*, the proxy turns off. The source Camp entity switches to the transmission of the data as soon as it receives an accepting report. The proxies upstream the destinations that have not yet sent a decision report operate as described before, remarking the data packets

[1] All the downstream interfaces in case (1); only the newly created downstream interface in case (2).

probing interfaces	*state*	*destinations*
oif 1	`data`	{rcvs}_1
...	...	...
...	...	...
...	...	...
oif n	`probing`	{rcvs}_n

(a)



(b)

Figure 4: (a) Data structure maintained by a proxy. (b) State transition diagram for the CAMP proxy

as "`probes`". They use the received reports to update their tables, but drop the reports; they turn off when there are not anymore *probing oif*s. Figure 4(b) summarizes the proxy operations.

The proxy mechanism solves the problems due to a dynamically changing tree topology and achieves the following goals:

1. the "`probe`" packets allow to evaluate the resource availability only along the new branches;

2. the new destinations immediately start receiving the real data, although possibly with a lower quality than required;

3. the data sent to the new destinations do not affect previously established sessions traversing the new branch, as the data packets are marked and treated as `probe` packets;

4. the membership is hidden to the source. Anyway, it must be known by the proxy.

It is worth to notice that, to guarantee the termination of the probing phase, the report loss must be prevented. This can be achieved by adopting TCP as the transport protocol used by RTCP.

3.2 Cascading proxies

The concurrency amongst receivers in joining the group may generate *cascading* proxies. In the example depicted in Figure 5, at time t_0 the data transmission is ongoing when the node C joins the group (Figure 5(a)) and it grafts the tree via the branch $A - B - C$. The node B becomes an in-tree node, and a proxy p_A is instantiated in the node A. Let us suppose that the node D joins the group at the time t_1 (Figure 5(b)), while the set-up phase for C is ongoing, and that D grafts the tree via the branch $B - D$. Since B is now an in-tree node, it instantiates a proxy, p_B. p_B receives data packets remarked as `probes` from p_A, and forwards them to both the downstream interfaces towards C and D. Let us suppose that at

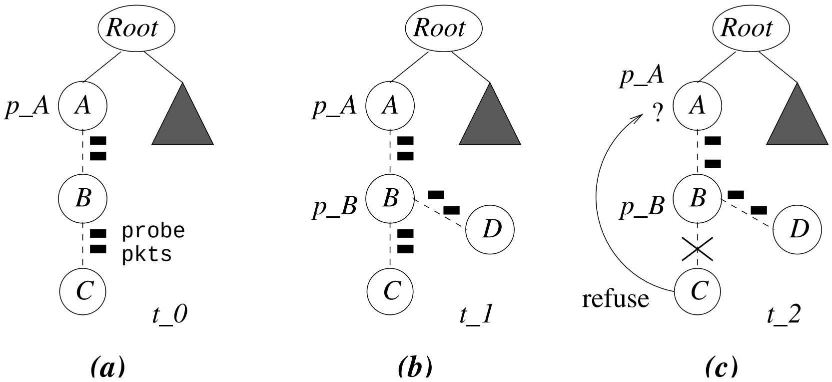


Figure 5: Example of cascading proxies

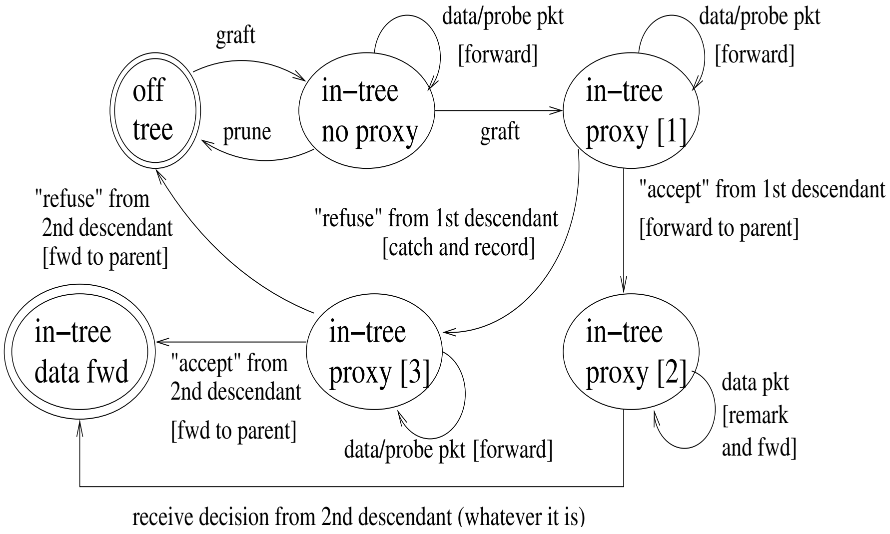


Figure 6: State transition diagram for the child proxy

time t_2, C terminates its set-up phase (before D). If C refuses the service, it prunes from the tree (Figure 5(c)). Yet, p_A should not turn off after removing the unique downstream interface from its table (Figure 4); rather, it must continue to forward the packets along the tree because D could still accept the service. To this purpose, B catches the C refusing report and waits for the D decision. On the other hand, if C accepts the service and B does not forward the C report to A, B does harm to C, which would receive the data with the QoS achievable by the probing traffic until D decides. Hence, B must forward the C accepting report to A, and remark the data as `probe`s before forwarding them to D. The state transition diagram shown in Figure 6 summarizes the B behaviour. Let us notice that, according to the conditions listed in section 3.1, B has as *probing* interface only the *oif* to D but not the *oif* to C. To support the cascading proxy mechanism, the receivers piggyback their own proxy address in their RTCP reports (as an application dependent field [21]), and the reports are forwarded upstream the tree, so that the intermediate proxy can snoop them.

The depicted interaction scheme between parent and child proxies is replicated for multiple tree levels if there are more than two cascading proxies.

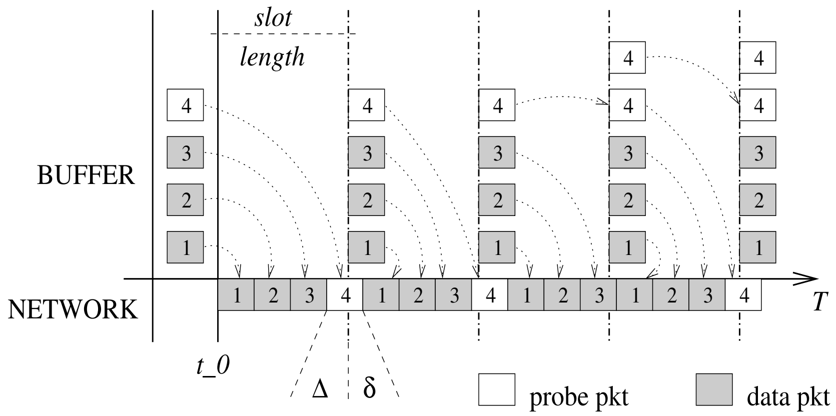


Figure 7: Example of bandwidth sharing among homogeneous sessions

4 Discussion

A key aspect of the active measurement-based approach concerns the length of the probing phase. As noticed in [5], a long probing increases the probability of *starvation* and thus of low network utilization. This is due to the greater probability of overlapping amongst several probing traffics along the same path: all the receivers of the different sessions may measure an unacceptably low rate, and refuse the service, while the path could accommodate at least one of the sessions. Long probing phases negatively affect the best effort traffic as well. On the other hand, the probing phase cannot be too short, otherwise the bandwidth availability cannot be estimated with sufficient accuracy, thus possibly leading to misacceptances. Let us consider the example shown in Figure 7: at time t_0 three sessions have already been accepted, while a fourth session starts its set-up phase. All sessions have the same rate; the *slot length* equals the inter-packet delay for one session. Let us suppose that the receivers consider only two `probe` packets to estimate the available bandwidth. As shown in the figure, the first two `probe`s are transmitted; if their inter-packet delay is such that the estimated rate is acceptable (with a given tolerance), the fourth session switches to the data transmission phase although there is not enough bandwidth for it. That example leads to conclude that a *lower bound* exists on the probing phase length[2].

A trivial lower bound is 2: this is the minimum because a recipient can estimate the incoming bandwidth. We have characterized the lower bound in a base case. Let us assume that:

- the sources generate traffic which is *perfectly* CBR;
- all the sessions have the same rate r;
- all the sessions use the same frame size f and the same packet size p (which is the frame size plus possible headers);
- the links are uniform, i.e., they all have the same capacity $\mathcal{C}$.[3]

[2]These arguments apply to the unicast case as well.
[3]This assumption can be easily relaxed by taking as $\mathcal{C}$ the capacity of the bottleneck link.

The inter-frame delay i is given by: $i = f/r$; this is the period with which a source supplies new frames to be transmitted[4]. The available bandwidth in i is $i \cdot \mathcal{C}$; throughout i we can transmit $i \cdot \mathcal{C}/p$ packets. The maximum number of acceptable sessions is: $\lfloor i \cdot \mathcal{C}/p \rfloor$. The residual bandwidth is:

$$\Delta = (i \cdot \mathcal{C}) - \lfloor \frac{i \cdot \mathcal{C}}{p} \rfloor \cdot p < p$$

If one more session is transmitted, as session 4 in the example, the used bandwidth exceeds the bandwidth available in i of:

$$\delta = (\lfloor \frac{i \cdot \mathcal{C}}{p} \rfloor + 1) \cdot p - (i \cdot \mathcal{C}) = \lfloor \frac{i \cdot \mathcal{C}}{p} \rfloor \cdot p + p - (i \cdot \mathcal{C}) = p - \Delta > 0$$

After $\lceil p/\delta \rceil$ slots, one `probe` packet cannot be transmitted because the channel is preempted by data packets (as it happens in the third slot in the example of Figure 7). Hence, the probing phase must last at least $\lceil p/\delta \rceil$ slots, and a recipients must wait to receive at least $\lceil p/\delta \rceil$ `probe` packets before deciding. After that interval, the session can be accepted only if the recipient tolerates a packet loss probability greater or equal to $(\lceil p/\delta \rceil)^{-1}$. It is worth to notice that the above lower bound can be computed using information (source traffic profile, bottleneck capacity) that the recipients can know. The evaluation does not take into account the possible interactions of a probing phase with the best effort traffic or other concurrent probing traffic. With those traffics the dropping of the `probes` would simply occur erlier, thus making congestion more evident. Anyway, as they could be not present in the network, they cannot be considered in the lower bound estimate.

Let us consider the case of VoIP traffic: according to G.711 [24], CBR traffic is generated with rate r of 64 Kbps, constituted by frames of $f = 8$ bits at intervals of $i = 125$ μsec. If $\mathcal{C} = 10$ Mbps, then the bandwidth available in a slot is $i \cdot \mathcal{C} = 1250$ bits. Assuming that each frame is sent in an IP packet, 40 bytes are needed for the TCP and IP headers; hence, $p = 328$ bits. The maximum number of acceptable sessions is 3, $\Delta = 266$ bits and $\delta = 62$ bits. Hence, the probing phase must last for at least 6 slots, that is, 750 μsec.

In the case of heterogeneous sessions, the probing phase must last long enough to guarantee that a session with high rate overlaps with the sessions with the lowest rate. Consider the example shown in Figure 8: over a channel with capacity 2 Mbps, 5 CBR sessions have been established having rate 128 Kbps and packet size 1 Kb each. Hence, the slot length is 7.8 msec., the available bandwidth in the slot is 15.6 Kb and the bandwidth occupancy in the slot is 5 Kb. At t_0, the probing is started for a session $\mathcal{S}$ with rate 1.5 Mbps and packet size 500 bits. Let us suppose that the bandwidth availability is estimated over 10 samples. The $\mathcal{S}$ source generates around 23 packets in a slot, and around 21 of them can be sent exploiting the bandwidth available in a slot. The first 10 packets of $\mathcal{S}$ are sent at a rate higher than the source rate, as shown in the figure. The recipients measure that high rate, because they start the evaluation upon the reception of the first `probe`, and cannot be aware of the existing backlog. As the consequence, the recipients can accept the service, although not enough available bandwidth exists for $\mathcal{S}$.

Similar problems must be dealt with to support VBR sources. The easiest, although inefficient, way to support VBR traffic is to consider it equal to a CBR traffic profile with

[4]In the example, $f = p$ and i is the slot length.

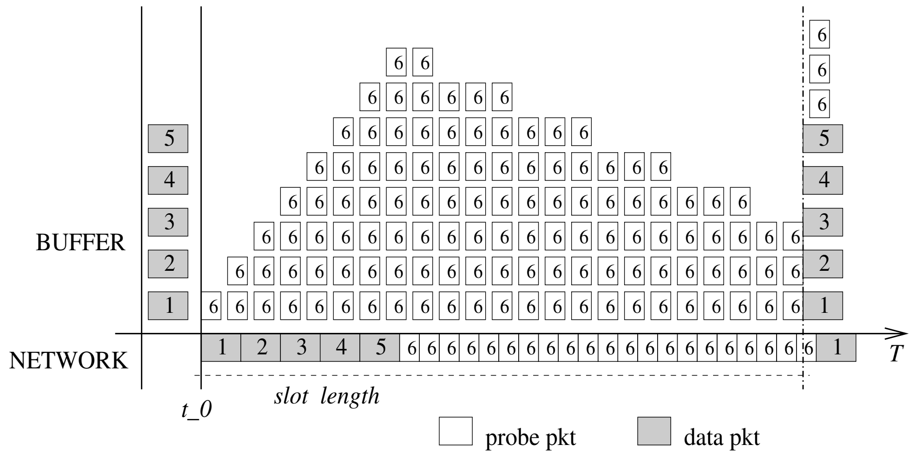


Figure 8: Example of bandwidth sharing among heterogeneous sessions

rate set to the VBR peak rate. The probing traffic is thus CBR, while during the transmission of the data the Camp entity should generate *dummy* packets in the periods of low source rate, to avoid that the acquired bandwidth is stolen by other traffic. This approach is for instance adopted in [2, 1].

5 Performance evaluation

We have implemented the architecture shown in Figure 2, in the frame of the ns-2 simulation package [10], to evaluate how Camp behaves when several QoS sessions compete with the best effort traffic for the available bandwidth. The simulations have been performed with a meshed network of 64 nodes, connected by optical links of 2 Mbps bandwidth and variable length in the range 50 to 100 Km. Background, best effort traffic is uniformly distributed all over the network; it is generated by 32 unicast connections among randomly chosen pairs of nodes. Best effort sources generate CBR traffic with a 0.66 Mbps rate, so that about 33% of the network resources is used. The size of best effort, `probe` and data packets is 512 bytes. We embedded a real RTP implementation [20] into the RTP template of ns-2. The recipients dynamically join the group; we performed experiments with different join rates. The multicast tree is incrementally built as join events occur. We considered the PIM multicast routing protocol; the source is located in the tree root. The source does not know the group of recipients; it generates CBR traffic whose rate assumes different values in the range 0.4 Mbps to 1.9 Mbps. During the probing phase, the recipients compare the received rate with the source rate: if the difference is below a tolerated threshold, a recipient sends a positive report. We performed measures for different thresholds [18]. The recipient decision is sent within the first RTCP report a destination generates after the reception of a number of `probe` packets, i.e. of samples, sufficient to ensure an accurate measure of the available bandwidth by covering the rate of the slowest traffic source. We performed simulations with different sample sizes. By simulating four QoS sources with rate 0.6 Mbps, we achieved a correct behaviour of the admission control procedure, i.e., the session refusal by all the recipients of the last started session, when considering only 10 `probe` packets. In this case

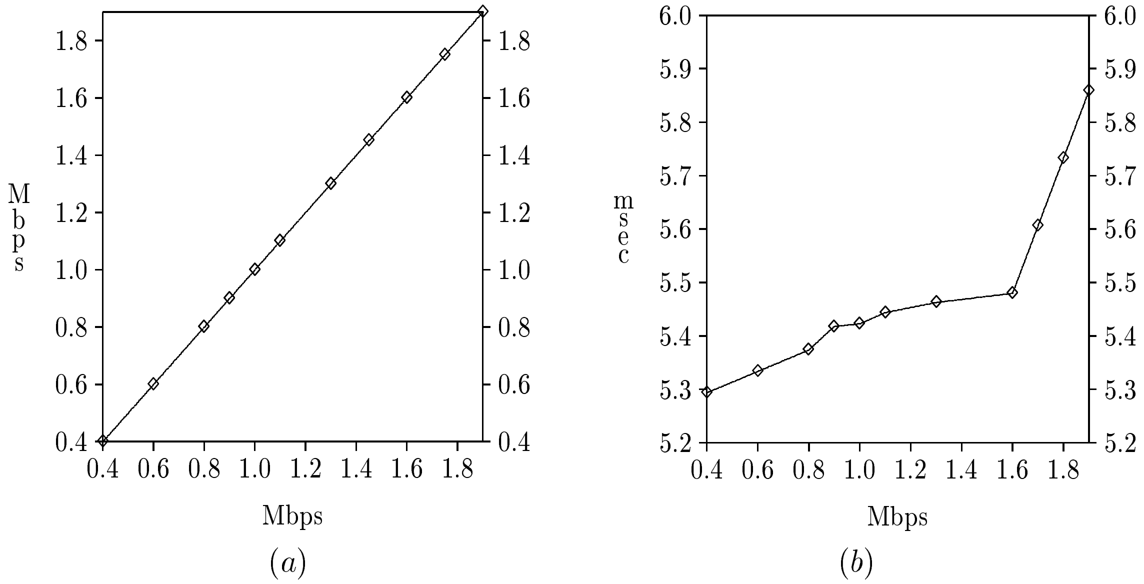


Figure 9: (*a*) Throughput vs. offered load for $|G| = 10$. (*b*) Average end-to-end delay vs. offered load for $|G| = 10$

the lower bound would be 2, assuming a frame size equal to the packet size. By contrast, we performed an experiment with five sources with rate 128 Kbps and one source with rate 1.5 Mbps, thus reproducing the setting shown in Figure 8. With the recipients considering as low as 10 `probe`s, all the sessions have been accepted although the offered load is 2.14 Mbps.

By performing simulations with different group cardinalities, we observed that the performance is almost independent of this parameter. We performed simulations with recipients that join the group with different rates: we considered cases in which a new join request is generated every 0.5 sec., every 1 sec. or every 2 sec. In all cases, a receiver decided about the service acceptance after having received a number of `probe` packets (precisely, 500) such that the set-up phases for different receivers partially overlapped. The proxy mechanism has proved to be effective in performing the admission control, and the measured performance indexes are independent of the frequency with which recipients join the group. The results shown in this section have been obtained for a group of 10 recipients, acceptance threshold set to 5% of the source rate and frequency of the join requests arrival 1 sec. The measures have been taken after 20 sec. from the end of the set-up phase of the last grafted recipient.

Simulations indicate that CAMP effectively performs the call admission control. In fact, the `probe` packets during the set-up phases and the data packets then, are able to drain the available bandwidth at the expenses of the best effort traffic. The recipients accept the transmissions and receive at the correct data rate. In Figure 9, we report the throughput (*a*) and the end-to-end delay (*b*) averaged over all the recipients; no receiver has refused the service. The end-to-end delay is almost constant before saturating the links. The delay increases when the offered load approximates the link capacity, while it is independent of the interference of the best effort traffic, with which the contention for the network resources starts at 1.3 Mbps of offered load. This indicates that the sessions characteristics are preserved from source to destination, independently of the other network load.

The jitter is the variance on the packet inter-arrival time and has been computed accord-

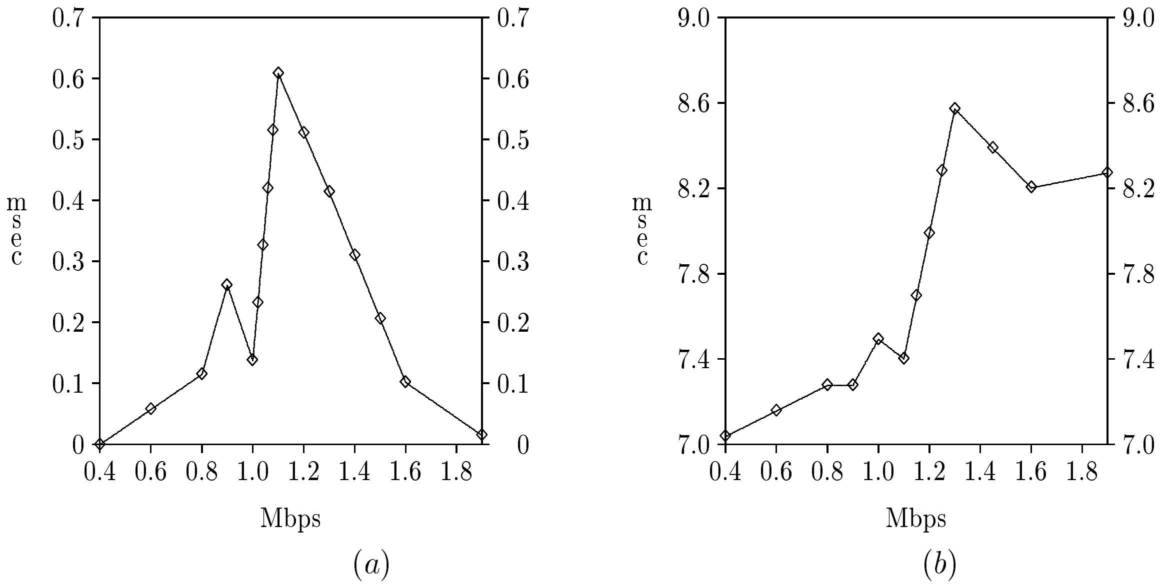


Figure 10: (a) Jitter vs. offered load, for $|G| = 10$. (b) Fair delay vs. offered load, for $|G| = 10$

ing to the algorithm given in the RTP specification [21]; it is reported in Figure 10(a). The jitter behaviour is uncontrolled, thus indicating that at the receiver side a delivery agent must be used to perform the playback of the source transmission. The jitter shows a peak in correspondence with the maximum contention between the QoS and the best effort traffic. After that value of offered load, the QoS traffic pushes the best effort traffic away from the tree branches, and best effort packets start to be dropped from the queues. As the consequence, the jitter decreases together with the decreasing of the amount of best effort packets competing to use the link resources. Jitter approaches 0 when the QoS traffic saturates the links.

The *fair delay* is the maximum difference between the end-to-end delays perceived by two different destinations. Its behaviour (Figure 10(b)) indicates that the destinations at a greater distance from the source greatly suffer the network congestion. In the worst case, this could result in a lower probability of successful service set-up for the farest destinations. However, we never observed service refusal, except for simulations in which we overloaded the network to verify the effectiveness of the admission control mechanism [18]. In those cases, CAMP correctly identified the receivers able to receive at the proper data rate and removed the others from the group.

The measures of the jitter and the fair delay show the effects of the presence of best effort packets at the core routers. As expected, the priority mechanism alone is not sufficient to ensure jitter control at the destinations. Although the proposed admission control mechanism has proved to be effective to provide bandwidth guarantees, the system characteristics may affect other QoS parameters. To highlight the impact of the best effort traffic over the QoS, we performed simulations with different best effort packet sizes (the `probe` packets always have the same size as the QoS data packets). In Figure 11(a) we report the end-to-end delay observed by the QoS packets that compete with best effort traffic generated as before. The delay observed by the data packets depends on the best effort packet size. As the links

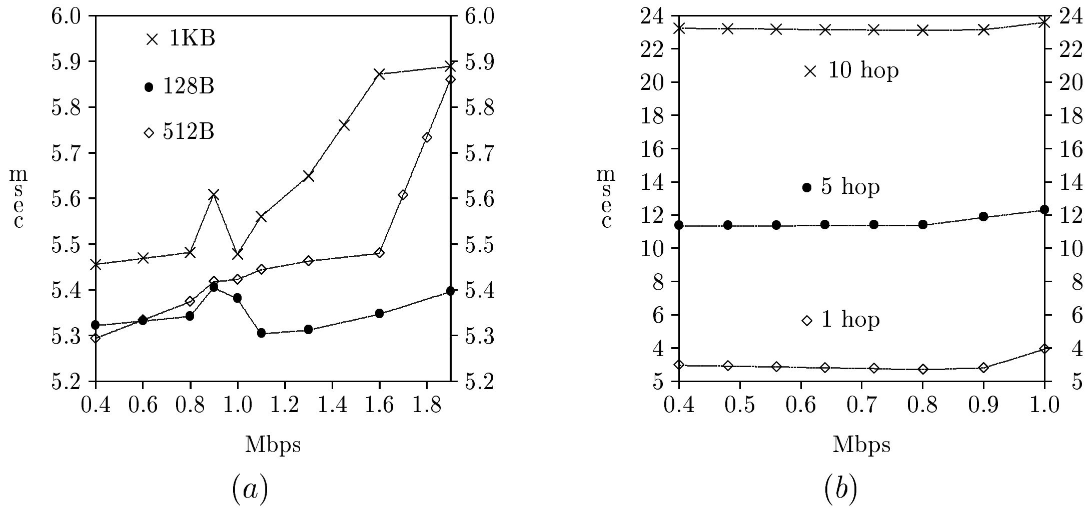


Figure 11: (a) Average end-to-end delay vs. offered load as a function of the best effort packet size, for $|G| = 10$. (b) Average end-to-end delay vs. offered load as a function of the receivers distance from the source.

cannot be preempted once a packet transmission is ongoing, QoS packets arriving at a node could have to wait at most for a best effort packet transmission time before gaining the link, although they have the highest priority. The impact of the measured delays over the received rate is however always negligible.

The QoS measured at the recipients may be affected as well by the distance of a recipient from the source. We considered a network having a tree topology, such that the QoS sources are located in the root node, to which three recipients are grafted at distances of respectively 1, 5 and 10 hops. The tree links have 2 Mbps bandwidth each. Only the QoS traffic is generated. Two sources exist: the former one has a 1 Mbps rate in all simulations; we varied the rate of the latter source in the range 0.4 to 1 Mbps. In figure 11(b), we show the average delay measured with respect to the load generated by the second source. The contention among the different sessions affects the queueing delays, thus altering the regular traffic profile. The impact on the received throughput is however negligible, as shown in Figure 12.

6 Concluding remarks

In this paper we propose the CAMP protocol to provide bandwidth guarantees to multicast applications with dynamic changes of the destination group membership. The mechanism performs admission control by adopting an active-measurement approach. Dynamic membership changes are supported by creating CAMP proxies in the routers that must carry out the admission control procedure for the new destinations. The proxies are destroyed when the set-up phase terminates. We implemented the CAMP protocol in the framework of the NS-2 simulation package. The achieved results show that the devised mechanism effectively performs admission control. The accepted services have the requested QoS guaranteed. Yet, further investigation has to be carried out concerning the interactions amongst several concurrent transmissions and their impact on the probability of successful service

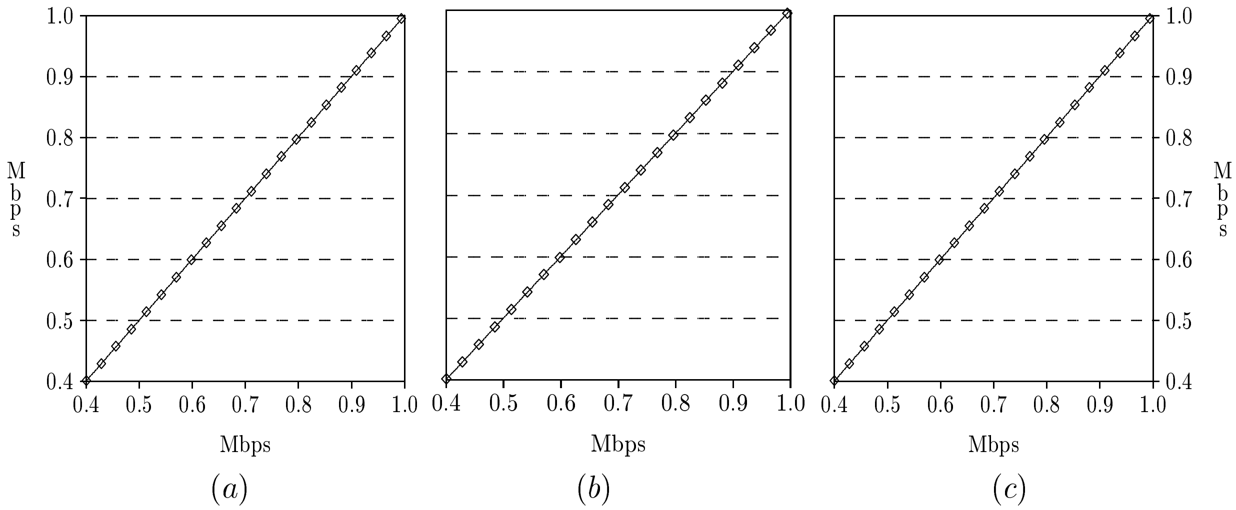


Figure 12: Throughput vs. offered load with distance among receivers and source of (a) 1 hop; (b) 5 hops; (c) 10 hops

establishment. The characterization of a lower bound on the probing phase length must be extended to consider the case of heterogeneous sessions and heterogeneous packet sizes.

According to the diff-serv model, core routers should be QoS-unaware. As the consequence, so far the proxy mechanism can be implemented only in the boundary routers, thus supporting changes in the multicast tree topology at the inter-domain level only. The implementation of the proxies requires changes with respect to the current router architecture. In line with the diff-serv model, it is probable that the boundary routers will be dedicated machines with peculiar capabilities to perform traffic classification, shaping and remarking; hence, proxy functionalities could be as well provided. We are investigating the possibility of implementing the proxy mechanism in the core routers by using mobile agents and exploiting the IP router alert option [13]; this would allow to support intra-domain membership changes.

So far, proxies can support dynamic multicast membership under the assumption that all the recipients grafted to the tree receive the same transmissions. We are extending their capabilities so that heterogeneous destinations, willing to accept different sets of data flows, can be supported. Similarly, the mixing and translation functionalities of RTP could be embedded into the proxies.

References

[1] Bianchi G., Blefari-Melazzi N. "A Migration Path for the Internet: From Best-Effort to a QoS Capable Infrastructure by Means of Localized Admission Control". *Proc. International Workshop on QoS in Multiservice IP Networks, Lecture Notes in Computer Science 1989*, page 422, Jan. 2001. Springer, Berlin.

[2] Bianchi G., Borgonovo F., Capone A., Fratta L., Petrioli C. "PCP: an End-to-End Measurement-Based Call Admission Control for Real-Time Services over IP Networks". *Proc. International Workshop on QoS in Multiservice IP Networks, Lecture Notes in Computer Science 1989*, page 391, Jan. 2001. Springer, Berlin.

[3] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W. "An Architecture for Differentiated Services". *RFC 2475*, Dec. 1998. Work in progress.

[4] Braden R., Clark D., Shenker S. "Integrated Services in the Internet Architecture: an Overview". *RFC 1633*, Jun. 1994. Work in progress.

[5] Breslau L., Knightly E., Shenker S., Stoica I., Zhang H. "Endpoint Admission Control: Architectural Issues and Performance". *Proc. SIGCOMM'00*, pages 57–69, 2000.

[6] Cain B., Deering S., Thyagarajan A. "Internet Group Management Protocol, Version 3". *Internet Draft, draft-ietf-idmr-igmp-v3-01.txt*, Feb. 1999. Work in progress.

[7] Cetinkaya C., Knightly E. "Egress Admission Control". *Proc. INFOCOM'00*, 2000.

[8] Elek V., Karlsson G., Rönngren R. "Admission Control Based on End-to-End Measurements". *Proc. INFOCOM'00*, 2000.

[9] Estrin D. et al. "Protocol Independent Multicast - Sparse Mode (PIM-SM)". *RFC 2362*, Jun. 1998. Work in progress.

[10] Fall K., Varadhan K. "ns Notes and Documentation". The VINT Project, Jul. 1999. `http://www-mash.CS.Berkeley.EDU/ns/`.

[11] Handley M. "The sdr Session Directory: An Mbone Conference Scheduling and Booking System". `http://mice.ed.ac.uk/mice/archive/sdr.html`, Apr. 1996.

[12] Jamin S., Shenker S., Danzig P. "Comparison of Measurement-based Admission Control Algorithms for Controlled-Load Service". *Proc. INFOCOM'97*, 1997.

[13] Katz D. "IP Router Alert Option". *RFC 2113*, Feb. 1997. Work in progress.

[14] Lai K., Baker M. "Measuring Link Bandwidths Using a Deterministic Model of Packet Delay". *Proc. SIGCOMM'00*, pages 283–294, 2000.

[15] Moy J. "Multicast Extensions to OSPF". *RFC 1584*, Mar. 1994. Work in progress.

[16] Nichols K., Blake S., Baker F., Black D. "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers". *Internet Draft, draft-ietf-diffserv-header-04.txt*, Oct. 1998. Work in progress.

[17] Nichols K., Jacobson V., Zhang L. "A Two-bit Differentiated Services Architecture for the Internet". *Internet Draft, draft-nichols-diff-svc-arch-00.txt*, Nov. 1997. Work in progress.

[18] Pagani E., Rossi G.P., Maggiorini D. "A Multicast Transport Service with Bandwidth Guarantees for Diff-Serv Networks". *Proc. International Workshop on QoS in Multiservice IP Networks, Lecture Notes in Computer Science 1989*, page 129, Jan. 2001. Springer, Berlin.

[19] Räisänen V. "Measurement-Based IP Transport Resource Manager Demonstrator". *Proc. International Conference on Networking, Lecture Notes in Computer Science 2094*, pages 127–136, Jul. 2001. Springer, Berlin.

[20] Schulzrinne H. "RTP: About RTP and the Audio-Video Transport Working Group". `http://www.cs.columbia.edu/~hgs/rtp/`.

[21] Schulzrinne H., Casner S., Frederick R., Jacobson V. "RTP: A Transport Protocol for Real-Time Applications". *RFC 1889*, Jan. 1996. Work in progress.

[22] Shenker S., Wroclawski J. "General Characterization Parameters for Integrated Service Network Element". *RFC 2215*, Sep. 1997. Work in progress.

[23] Teitelbaum B., Chimento P. "QBone Bandwidth Broker Architecture". *Internet 2 QoS Working Group Draft*, Jun. 2000. Work in progress. `http://qbone.internet2.edu/bb/`.

[24] Tobagi F. et al. "Service Differentiation in the Internet to Support Multimedia Traffic". *Proc. IWDC 2001, Lecture Notes in Computer Science 2170*, pages 381–400, Sep. 2001. Springer, Berlin.

[25] Wroclawski J. "Specification of the Controlled-Load Network Element Service". *RFC 2211*, Sep. 1997. Work in progress.