A Multicast Transport Service with Bandwidth Guarantees for Diff-Serv Networks*

Elena Pagani, Gian Paolo Rossi, and Dario Maggiorini

Dip. di Scienze dell'Informazione, Università degli Studi di Milano
{pagani, rossi, dario}@dsi.unimi.it

Abstract The Int-Serv and Diff-Serv differentiation approaches, which have been proposed by the IETF, are in practice unable to provide QoS guarantees to the emerging multicast-enabled applications by means of a scalable mechanism able to support QoS services on a per-call basis and dynamic group memberships.
This paper describes a new approach that is capable of ensuring bandwidth guarantees to multicast sessions on IP-based networks and satisfies the above requirements. It includes a scalable, end-to-end *Call Admission Multicast Protocol* (CAMP) that operates as a sort of distributed bandwidth broker and allows to combine the benefits of both the IS and DS approaches in a single approach which is simple, scalable, operates on a per-call basis and supports the group membership dynamics.

Keywords: Quality-of-Service, multicast, call admission control, diff-serv domains, bandwidth broker, performance evaluation.

1 Introduction

A great deal of efforts has been recently devoted to provide through the Internet a transport platform capable of satisfying the Quality-of-Service (*QoS*) requirements of the emerging multimedia and real-time applications. This goal cannot be achieved without the introduction of some traffic differentiation policy, that changes the best-effort nature of the Internet protocols into a new generation of protocols providing the QoS guarantees (e.g., in terms of delay, bandwidth, jitter [7,2]), and addresses the following general application requirements:

multicast: most of the video, voice and multimedia applications require the support of one-to-many or many-to-many interaction schema [6];
scalability: the QoS-capable transport mechanism should scale to the entire Internet and to large multicast groups;
dynamic sessions: the QoS-capable transport mechanism should accept the dynamic opening/closing of QoS sessions on an on-demand basis;

* This work was supported by the MURST under Contract no.9809321920 "Transport of multicast packets with QoS guarantees".

group membership: the membership of the multicast group may be *static*, i.e. the result of an announcement and subscription process through, for instance, `sdr` [11], or *dynamic*. In the former case, the source can be aware of the group membership, while, in the latter, the source is generally unaware of both the membership and the cardinality of the group over time.

The Int-Serv (IS for short [5]) and Diff-Serv (DS for short [3]) differentiation approaches, which have been proposed by the IETF, do not completely satisfy the above requirements and, although they consider the multicast communications, in practice, they do not offer a satisfactory solution for managing group membership and session dynamics. IS can dynamically provide QoS guarantees on a per-flow basis through RSVP [18]. To do that, it requires substantial changes in the router architecture; each router maintains per-flow status information by running a hop-by-hop information exchange protocol. This makes RSVP hard to scale and unusable, in practice, when large multicast groups are considered. DS, on the contrary, considers flow aggregates and all packets belonging to the same aggregate receive the same network service on the base of the TOS byte in their IP header (DS byte in IPv6) [13], that identifies, and differentiates, the packet's *Per-Hop-Behaviour* (PHB). The DS model is simple and more scalable because it confines the QoS complexity at the *border routers* (i.e., the routers of the access network or the routers on a domain border) and leaves the core routers nearly unchanged. This approach requires that an end-to-end admission mechanism (the *bandwidth broker* [14]) dynamically controls and limits the amount of incoming flows to be assigned to a given aggregate. So far, such an end-to-end admission mechanism has not been designed for multicast-enabled sessions. As the consequence, the current QoS multicast experiences (e.g., QBone [10]) still consider a static environment where the amount of multicast QoS sessions and the group membership are both statically defined in a sort of permanent multi-point virtual circuit.

This paper describes a new approach that is capable of ensuring bandwidth guarantees to multicast sessions on IP-based networks and satisfies the above requirements. It includes a scalable, end-to-end *Call Admission Multicast Protocol* (CAMP) that allows to combine the benefits of both the IS and DS approaches in a single approach which is simple, scalable, operates on a per-call basis and supports the group membership dynamics. CAMP performs the functionalities of a distributed bandwidth broker and has been designed to operate within the RTP/RTCP protocol suite [16] and on top of a Diff-Serv network. When the membership dynamically changes, CAMP utilizes a CAMP *proxy* which is instantiated within the core, intree router involved in the topology change. The CAMP proxy however, autonomously and locally finds the needed information and stays active for the call set-up time only. Then, it turns off and the core routers have not to maintain status information.

As far as we know, this is the first attempt to address all the above requirements for multicast traffic in DS networks. A similar approach has been recently followed in [4] and applied to point-to-point video/voice sessions. The simulation results are encouraging and indicate that the approach is efficient for CBR traffic. Although the transition to support VBR traffic seems affordable, this will be part of the forthcoming research.

The paper is organized as follows: in Section 2, we describe the basic version of the algorithm, and we introduce the system architecture. In Section 3, we discuss the modifications needed to support dynamic changes in the multicast group membership. In Section 4, we present our simulation results.

2 Call Admission Multicast Protocol

2.1 The System Architecture

In this work we consider a network $\mathcal{N}$ of interconnected LANs. The network constitutes a *domain* [3], that is, a contiguous set of routers which operate with a common set of service models, service definitions and provisioning policies. Throughout this work we consider a Diff-Serv domain, in which the QoS is provided on a per-packet basis.

We assume that the routers in $\mathcal{N}$ are *multicast-enabled*, QoS-unaware, and that they are configured to use the *priority* packet scheduling policy. A membership protocol is used (e.g., IGMP [8]) to discover the group membership of the attached LANs. A multicast routing protocol (e.g., DVMRP [15]) maintains a tree-based multicast infrastructure among the routers, but CAMP is independent of the underlying multicast protocol.

CAMP is implemented as a module of the RTP library, which is linked to the application (figure 1). It performs the call set-up of an RTP session and is activated once the application calls RTP passing the traffic profile, through, for instance, a `TSpec` [17]. CAMP exploits the services of RTP and RTCP [16]. RTCP is used to monitor the QoS provided at the recipients. RTP can be used at the receiver side to appropriately perform the playback of the source transmission. Upon the termination of the call set-up phase, the RTP session data transfer is initiated or terminated accordingly.

2.2 CAMP with Static Membership

This section describes the CAMP protocol that applies to static group membership. In this case, a session announcement protocol (e.g., `SDAP` [11]) can be used to announce the needed session information, such as the start time and the multicast address. Receivers perform an explicit join procedure and the group membership is defined before the transmission starts. The transmission source is aware of the membership of the group of recipients. Under these conditions, a multicast session is divided into two phases: the *session set-up* phase, performed by CAMP, and the *data transfer* phase, performed through RTP. In the set-up phase, CAMP uses the information supplied by the application to test if the network can guarantee the requested bandwidth. The test consists of sending `probe` packets to the destinations, so that the generated probing traffic has the same profile of the specified data traffic. The `probes` are forwarded along the multicast

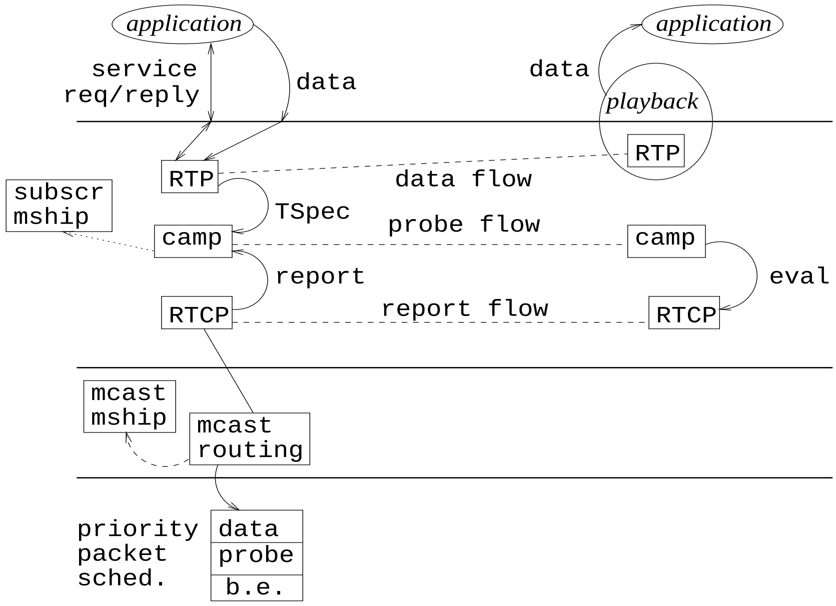

Figure 1. Architecture of a CAMP-based end station

routing tree. The basic idea consists of differentiating `probe`, QoS data and best effort packets by means of three priority levels. The `probe` packets are marked with a higher priority than the best effort packets and a lower priority than the QoS data packets. This priority assignment ensures that the probing traffic does not affect the existing QoS flows. On the other hand, `probe` packets can drain the available bandwidth for new QoS flows at the expenses of the best effort traffic.

At the receiver side, the CAMP entity evaluates the quality of the received probing traffic. The source diffuses its own traffic profile by using the RTCP reports. Each receiver r matches that profile against the received probing traffic profile and decides whether to accept the whole transmission or not. r informs the source about its decision with its own RTCP reports. A receiver that refuses the service unsubscribes from the group. As the consequence, the corresponding router may be pruned from the multicast routing tree.

If the service is accepted by at least one recipient, then the source switches from the transmission of `probe` packets to the transmission of the data packets generated by the source application, without discontinuity. Data packets are forwarded along the pruned tree. The priority policy ensures a sort of resource reservation: it guarantees that the bandwidth that has been 'assigned' to a given flow during its probing phase cannot be later assigned to other `probe` or best effort packets. Moreover, priority naturally provides flow aggregation.

The knowledge of the group membership is needed because the source *must* wait for the reports from *all* the receivers, before switching to the data transmission. The example of figure 2(a) explaines this point. The source generates a QoS flow $flow_1$. If it receives positive reports from a subset of receivers, let us say those in the subtree A, it knows that some destination is available for the data flow. If the source switches to the data transmission before receiving the reports from the receivers in the subtree B, other established QoS flows could be negatively affected. In fact, let us consider the flow $flow_2$ that traverses the B's branches, and does not leave enough bandwidth for $flow_1$. As the consequence, the receivers in B would eventually send negative reports and prune to refuse $flow_1$. If the switch is performed before receiving all the reports (i.e., before the needed prune operations), the data packets of the aggregate flowing on the subtree B will be dropped, independently of the flow they belong to.

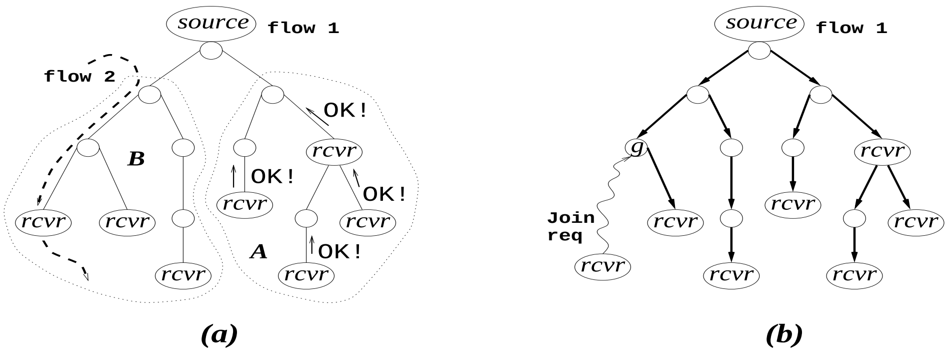

Figure 2. (a) Example of concurrent flows. (b) Example of dynamic group membership

The need of obtaining all the decisions requires that report loss is prevented, for instance by having RTCP that uses the TCP services. The source can notice when all the reports have been received, because, by hypothesis, it knows the (static) destination group membership beforehand. The case of dynamic group membership is discussed in the next section.

3 Supporting Dynamic Group Membership

In several applications, such as news services and video broadcasting, the source does not need, or desire, to be aware of the receivers group membership, while receivers are free to join and leave the group dynamically. In this new model, the source announces the multicast session via SDAP and starts transmitting at the scheduled time if at least one receiver is listening. In all these cases, the Camp model described above does not apply, because the tree infrastructure that supports the session flow is unable to maintain the bandwidth guarantees in case of topology changes. To understand where the problem arises, let us consider the example shown in figure 2(b), where a receiver wants to join the group when the transmission of a flow $flow_1$ is established. According to the multicast routing protocol, the joining node would graft to an intree router, say g. Anyway, the data packets cannot be forwarded along the new branch, without testing if enough resources are available to accommodate them, for the same arguments discussed before (see figure $2(a)$). The probing phase cannot be started at the root, to avoid the duplication of the resource reservation already installed for $flow_1$ on the path from the root to g. On the other hand, the probing can neither be started by g, which is QoS-unaware. In fact, $flow_1$ shall be transported through the new branch as part of the proper PHB to which it belongs and, in line with the DS model, the involved intree router should be unable to identify the subset of packets belonging to $flow_1$.

3.1 The CAMP Proxy

To overcome the above limitations we decided to install a CAMP *proxy* in the in-tree routers involved in the membership change. The proxies are in charge of managing the joining nodes.

When a new node wants to join a QoS multicast session, it explicitly joins the proper group of receivers (through, for instance, IGMP report exchange). Let us indicate with f the transmission flow and with G_f the group of the receivers of f. As soon as an intree router g, belonging to the tree T_f for G_f, creates a new downstream interface for T_f, it instantiates the CAMP proxy, say p_f^g, whose lifetime lasts until the set-up phase for the new host terminates. Once activated, the proxy receives the identifier of the new interface and generates, by gathering the information from the local routing table, the data structure of figure 3, where it temporarily maintains the status of the active probing phases. If a new join request is received while the proxy is active, a new entry of the data structure is created. The proxy turns off when all the entries in the data structure have been deleted (service refused) or switched to data transfer.

probing interfaces	*state*
output interface 1	*data*
...	...
...	...
...	...
output interface n	*probing*

Figure 3. Data structure maintained by a proxy

During its activity the proxy re-marks as `probe` packets all the incoming QoS data packets that should be routed to the probing interfaces. Moreover, the proxy snoops all the packets received from the downstream interfaces. When a RTCP report is received, p_f^g temporarily records it and drops it to prevent its forwarding. Should the report be positive, the proxy changes the corresponding entry from the state *probing* to *data*. Otherwise, it removes the entry. This simple mechanism solves the problem 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 branch;
2. the new destination immediately starts receiving the real data flow, although possibly with a lower quality than required;
3. the data sent to the new destination do not affect previously established flows traversing the new branch, as the data packets are marked and treated as `probe` packets;
4. the membership is hidden to the source.

To show that the proxy mechanism is in practice very general, let us consider the system condition when the multicast session is initiating. At that time, we assume that a certain amount of recipients joined the multicast tree (in the trivial case of no recipients the transmission would not be initiated to avoid the waste of bandwidth) and that the local CAMP proxies are active as described before waiting for some packet to forward downstream. At the scheduled session time, the CAMP protocol at the source host will start the probing phase. CAMP proxies act as described: remark (although uselessly) the incoming `probe` packets into outgoing `probe` packets and wait for snooping RTCP reports. In addition, they understand they are running a probing phase because of the type of the incoming packets. As long as they receive `probe` packets, they snoop RTCP reports to update the local data structure, but forward them upstream. When the source CAMP entity receives at least one positive report from the group, it switches to the transmission of the data and both the end systems and the proxies operate as described.

4 Performance Evaluation

We have implemented the architecture shown in figure 1, in the frame of the NS-2 simulation package [9], to evaluate how CAMP behaves when several QoS flows compete with the best effort traffic for the available bandwidth. We considered the DVMRP multicast routing protocol and we embedded a real RTP implementation [1] into the RTP template of NS-2. We simulated a meshed network of 64 nodes, connected by optical links of 2 Mbps bandwidth and variable length in the range 50 to 100 Km. The size of `probe`, best effort and data packets is 512 bytes. The background best effort traffic is provided by 32 unicast connections between randomly chosen source-destination pairs. Each best effort source generates traffic with a rate of 0.66 Mbps.

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. The 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. In all our simulations, the recipients decide after the reception of 500 `probe` packets. This way, the length of the set-up phase depends on the source rate.

A first set of experiments has been performed with two CBR sources located on the same node, and the acceptance threshold at the receivers set to the 5% of the source rate. The first source generates traffic at the rate of 1 Mbps; the rate of the second source assumes different values in the range 0.4 Mbps to 1.9 Mbps. The probing phase of the second source starts 1.5 sec. after the start of the first source, so that the two set-up phases partially overlap. By performing experiments with different group cardinalities for the two groups we observed that the performance are almost independent of the group cardinality. In the following, we only report the results we obtained when both groups have

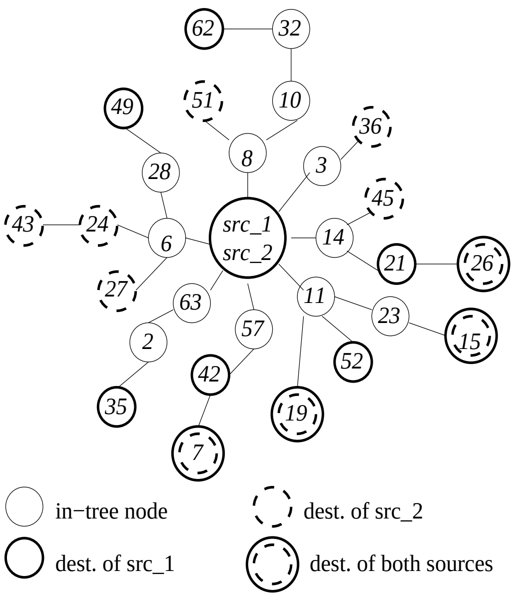

Figure 4. Overlapped multicast trees for the QoS sources

10 destinations. Groups partially overlap; the simulated topology restricted to the multicast routing trees for the two groups is shown in figure 4. The group membership is static throughout the simulation.

Simulations indicate that CAMP effectively performs the call admission control. In fact, until the sum of the source rates is lower than the link capacity, the recipients in both groups accept the transmissions and receive at the correct data rate. When the rate of the second source is higher than 1 Mbps, the source transmission can be accepted only by receivers linked through separate branches to the multicast trees. When the multicast trees of the two groups share the same link, only the destinations in one group shall be able to accept the transmission. CAMP correctly identifies the receivers able to receive at the proper data rate, removes the others from the group, and performs this task with the required fairness. In fact, it ensures that the transmission that started the probing phase earlier would not be negatively affected by the second group, and, when in competition, would receive the bandwidth guarantees as shown in figure 5. In figure 5(a) we report the number of recipients of both groups that accepted the transmission, with respect to the traffic generated by the second source. In figure 5(b) we show with a solid line the rate received at the destinations that accept the service. The dashed line represents the average rate obtained during the set-up phase by considering all the receivers, and also represents the data rate that would be observed at the destinations without the call admission control. As shown in figure 5(a), by setting the threshold to 2%, 5%, 10% of the source

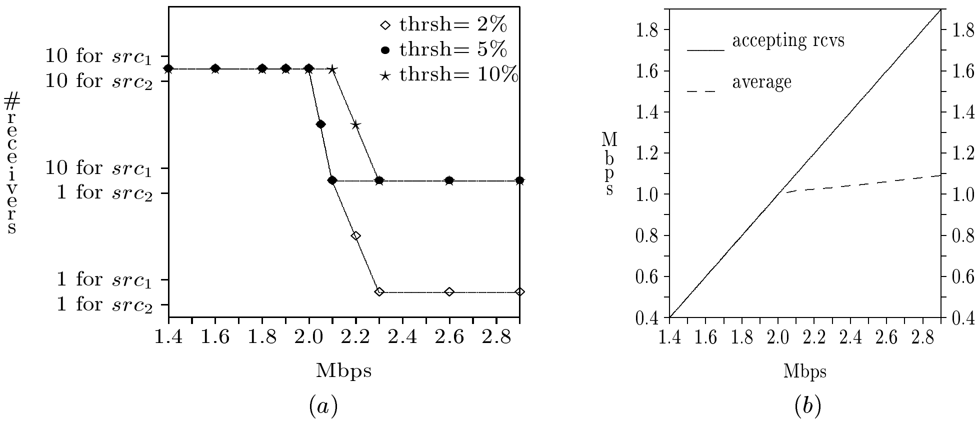

Figure 5. (a) Overall number of accepting recipients for both groups, for different threshold values and $|G| = 10$. (b) Average received rate for the second group of recipients vs. offered load, for $|G| = 10$ and threshold value 5%

rate, CAMP is able to properly provide the bandwidth guarantees to all the receivers in both groups until the data rate does not saturate the link capacity. At a higher request of bandwidth, and threshold at 5% and 10%, all the receivers in the first group accept the transmission while, in the second group, only the destinations on separate branches accept the transmission (in the simulation, only 1 receiver). As confirmed by the results of figure 5(b), the receivers able to accept the transmission receive the adequate bandwidth guarantees to receive at the correct data rate. When the threshold is set too low, 2%, the number of receivers able to accept the transmission becomes very low due to the observed delay and jitter, as discussed below. The acceptance threshold should be chosen according to the semantics of the source application, that is, it should be no greater than the maximum packet loss probability that can be tolerated so that the information does not become useless.

In figure 6, we report the observed end-to-end delay (figure 6(a)) and jitter (figure 6(b)). The jitter is the variance on the packet inter-arrival time and has been computed according to the algorithm given in the RTP specification [16].

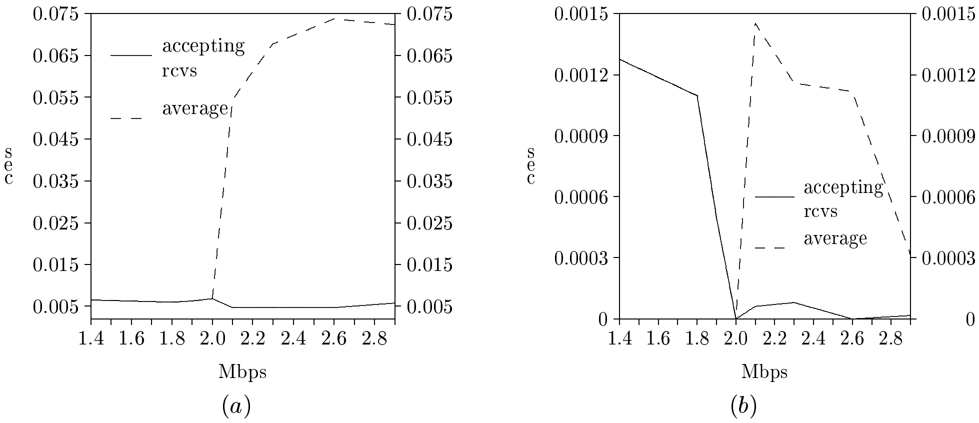

Figure 6. (a) Average end-to-end delay and (b) jitter for the second group of recipients vs. offered load, for $|G| = 10$ and threshold value 5%

The end-to-end delay is almost constant before saturating the links, and it remains unchanged for the accepting destinations with congested links. This indicates that the flows characteristics are preserved from source to destination, independently of the network load and the group cardinality. The measure of the jitter shows the effects of the presence of best effort packets at the core routers. As expected, the priority mechanism alone is not sufficient to ensure delay and jitter control at the destinations. 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 both the QoS sources generate at the rate of 1 Mbps (in this case, there is no space left for best effort traffic). At higher QoS rate some destination has to refuse the service (see figure 5(a)), thus allowing that a few best effort packets still succeed in accessing the network. As shown by figure 6(b), their effect is marginal over the links where the accepted traffic is forwarded.

By exploiting the RTP service is possible to annihilate the jitter, by delivering the received packets at a constant rate.

5 Concluding Remarks

In this paper we propose the CAMP library to perform call admission control in diff-serv domains for multicast applications. CAMP adopts an innovative approach, intermediate between the IS and the DS model, to support dynamic changes in the group of recipients. CAMP proxies are dynamically created in the routers that must take care of those changes. They perform the updates in the system configuration required by the membership changes, and then they are destroyed. We implemented the described mechanism in the frame of the NS-2 simulation package. The achieved results are promising: the devised approach 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 establishment.

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. In the near future, we will be able to perform experiments with the described architecture in the frame of a testbed that we are currently deploying over the departmental network [12].

References

1. `http://www.cs.columbia.edu/~hgs/rtp/`
2. Bagnall, P., Briscoe, R., Poppitt, A.: Taxonomy of Communication Requirements for Large-scale Multicast Applications. Internet Draft, draft-ietf-lsma-requirements-02.txt. Work in progress (1998)
3. Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., Weiss, W.: An Architecture for Differentiated Services. Internet Draft, draft-ietf-diffserv-arch-02.txt. Work in progress (1998)
4. Borgonovo, F., Capone, A., Fratta, L., Marchese, M., Petrioli, C.: PCP: A Bandwidth Guaranteed Transport Service for IP networks. Proc. IEEE ICC'99 (1999)
5. Braden, R., Clark, D., Shenker, S.: Integrated Services in the Internet Architecture: an Overview. RFC 1633. Work in progress (1994)
6. Bradner, S.: Internet Protocol Multicast Problem Statement. Internet Draft, draft-bradner-multicast-problem-00.txt. Work in progress (1997)
7. Bradner, S.: Internet Protocol Quality of Service Problem Statement. Internet Draft, draft-bradner-qos-problem-00.txt. Work in progress (1997)
8. Cain, B., Deering, S., Thyagarajan, A.: Internet Group Management Protocol, Version 3. Internet Draft, draft-ietf-idmr-igmp-v3-01.txt. Work in progress (1999)
9. Fall, K., Varadhan, K.: ns Notes and Documentation. The VINT Project: http://www-mash.CS.Berkeley.EDU/ns/ (1999)
10. Internet2 QoS Working Group Draft: Draft QBone Architecture. Internet Draft, draft-i2-qbone-arch-03.txt. Work in progress (1999)
11. Handley, M.: The sdr Session Directory: An Mbone Conference Scheduling and Booking System. http://mice.ed.ac.uk/mice/archive/sdr.html (1996)
12. Maggiorini, D., Pagani, E., Rossi, G.P.: A Testbed Environment for the Performance Evaluation of Modular Network Architectures. Proc. 1st IEEE European Conference on Universal Multiservice Networks (ECUMN 2000) (2000) 283–292

13. 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. Work in progress (1998)
14. Nichols, K., Jacobson, V., Zhang, L.: A Two-bit Differentiated Services Architecture for the Internet. Internet Draft, draft-nichols-diff-svc-arch-00.txt. Work in progress (1997)
15. Pusateri, T.: Distance Vector Multicast Routing Protocol. Internet Draft, draft-ietf-idmr-dvmrp-v3-09.txt. Work in progress (1999)
16. Schulzrinne, H., Casner, S., Frederick, R., Jacobson, V.: RTP: A Transport Protocol for Real-Time Applications. RFC 1889. Work in progress (1996)
17. Shenker, S., Wroclawski, J.: General Characterization Parameters for Integrated Service Network Element. RFC 2215. Work in progress (1997)
18. Zhang, L., Braden, R., Berson, S., Herzog, S., Jamin, S.: Resource reSerVation Protocol (RSVP) - Version 1 Functional Specification. RFC 2205. Work in progress (1997)