# Scalability Issues for Reliable Multicast Protocols

José F. de Rezende and Serge Fdida
LIP6 - CNRS
Université Pierre et Marie Curie
4, Place Jussieu - 75252 Paris Cedex 05
France
email: {Jose.Rezende, Serge.Fdida}@lip6.fr

**Abstract**

Many emerging applications are based on multimedia group communication. Multicast communication is a transmission mode that is now supported by a variety of local and wide area networks. Many group applications require a full reliable data transfer to all receivers. A problem in this context is how to provide a full reliable data transmission to a large number of receivers while still exploiting the advantages of multicast communication, such as bandwidth saving, and maintaining a suitable performance in a heterogeneous environment. This issue is generally referred as the problem of reliability versus scalability. In this paper we present different classes of multicast reliable protocols according to an existing taxonomy and discuss how well each of them handle large number of receivers.

## 1 Introduction

Distributed computing, distributed interactive simulation and cooperative applications require to support group communication over wide area networks. Multicast communication is a transmission mode that is now supported by a variety of local and wide area networks. However, multicast raises various new issues and problems related to data transmission and reliability. Different levels of reliability are required for different applications. There exist applications requiring a reliable transmission service but that allow for a number of unrecoverable errors. A shared whiteboard is an example for such an application. Others, like continuous media applications because of their real-time constraints can usually not afford retransmission of lost datas. On the other hand, full reliability is for instance needed for the transfer of a file to a group of users. In this case all receivers have to receive a correct and complete copy of the data. An efficient method to provide this level of reliability while still fully exploiting the advantages of multicast communication (e.g. bandwidth saving) is the key issue in the design of reliable multicast protocols.

A major problem in this context is how to provide a reliable service to a potentially large number of receivers, i.e. the problem of reliability versus scalability. Data transfer is reliable if all receivers will get an error free copy of the transmitted data. Error control algorithms are necessary to ensure reliable data transmission. Three techniques are used for error control : 1) automatic repeat request (ARQ) which uses acknowledgments, time-outs, and retransmissions, 2) forward error correction (FEC), and 3) error concealment (EC) at the receiver [1]. For continuous media that presents real-time constraints error recovery through retransmission is usually not appropriate. Forward error correction schemes and special encoding techniques are proposed to enhance reliability for audio and video [2, 3, 4, 5]. For discrete media, lost or corrupted data can be retransmitted to provide reliability. However, with an increasing group size the probability that one of the recipients receives a corrupted message increases and hence the likelihood that messages have to be retransmitted is also increasing. Further, a high overhead is due to the management of the receiver group. In a sender-initiated protocol, the one that places the responsibility for providing reliable multicast on the sender, the so called ACK-implosion effect can be observed with a growing number of receivers. The sender is mainly busy processing positive acknowledgments, and therefore not able to transmit more data.

The discussion in this paper concentrates on the problems of reliability versus scalability, i.e. how well reliable multicast protocols will handle large number of receivers. This paper is organized in three sections. In section 2 we introduce the different aspects related to the guarantee of reliability in a group communication environment and discuss the scalability problem. Further, in this section we present a taxonomy, derived from [6], to the classification of multicast reliable protocols presented in the literature that proposes a solution to the problem of scalability. In section 3 we introduce some other important issues related to scalability, such as distribution group model, misbehaving receivers and packet loss distribution. Finally, in section 4 we summarize the paper and give our final remarks.

## 2 Reliability in multicast communication

Reliable multicast communication entails a range of new problems affecting end-to-end mechanisms. For instance loss and error recovery is different from traditional unicast. In the error control mechanisms based on retransmission, the receiver(s) sends information to the source about the status of individual packets. The source identifies lost packets based on this information and then initiates the error recovery procedure. In order to achieve full reliability the sender has to keep a copy of a transmitted packet until all receivers in the receiver group have acknowledged its reception. Two types of schemes may be used to communicate information from the receiver to the source - positive acknowledgment (ACK) and negative acknowledgment (NAK) based schemes.

When using an ACK based scheme, the receiver sends back to the sender information about which packets it has received. Different strategies can be used depending upon when the receiver group must send information to the sender. It can be sent periodically, on the receipt of a certain amount of data or when explicitly required by the source. In an ACK based protocol the mechanism used for pacing the transmissions and for the detection of transmission errors can be sender or receiver-based [6]. In the first case, the sender is in charge of the retransmission time-out, i.e. the sender polls the receivers for ACKs implicitly by retransmitting after a time-out or explicitly by sending a special control message. In the latter, each receiver sets its own time-out and polls the source with an ACK after they timed out. In cumulative ACK schemes the receiver informs the sender about the highest sequence number below which all packets have been successfully received. In selective ACK schemes, the receiver informs the source about the sequence number of the packet(s) that it has received since the last ACK packet was sent. In [7], the term block-based ACK scheme is used to describe those schemes in which a block of packets rather than an individual packet is acknowledged by the receiver.

In NAK based schemes, the receiver sends a message to the sender only if it realizes that packets have been lost, detected by either an error, a gap in the sequence numbers used, or a time-out. After processing the information sent by the receiver group, the sender can verify if packets have been lost and if retransmission is needed. In this case, it can adopt different retransmission strategies - the most common ones are go-back-n and selective retransmission. In the first strategy, the source retransmits all packets starting from the first packet lost. In a selective retransmission scheme, only the lost packets are retransmitted.

Most early multicast/broadcast protocols are based on the ACK based scheme, also called sender-initiated approach [8]. In this class of multicast reliable protocols, the reliability does not only depend on the kind of network reliability and on the retransmission functions used but also on the size of the receiver group. The drawback of this approach is that with an increasing number of receivers the processing overhead at the sender results in a fast decreasing performance. This is due to an increasing load in the network and the source implosion effect. Both are caused by the high number of control messages that have to be transmitted from all receivers and to be processed by the sender. Specifically, at the sender host, the acknowledgment arrival rate has the potential of flooding the sender's message queue, causing acknowledgments to be lost. On the network, the size of the receiver group is directly proportional to the amount of network traffic and the potential decrease in the network throughput. In a CSMA/CD network, other sources of delay include the retransmitting and discarding of packets due to packet collision. Thus, the maximum throughput of sender-initiated protocols is dependent on the number of receivers. Therefore, either it is not possible to provide reliability for such large groups while maintaining a minimal performance or another way to ensure reliability has to be found.

In the remainder of this paper a number of solutions to the problem of scalability of reliable multicast protocols are examined. To ease the presentation of the different solutions to the scalability problem, we use the same taxonomy introduced by Levine *et al.* [6]. This new taxonomy of reliable multicast protocols organizes known approaches into four protocol classes. In addition to the most popular approaches to reliable multicasting, i.e. sender-initiated and receiver-initiated protocols, the paper introduces two other classes: tree-based and ring-based protocols. The first organizes the receivers in a tree and send ACKs along the tree; the latter send ACKs to the sender along a ring of receivers. Further, the authors discuss how many key papers in the literature fit within this taxonomy. In the present paper we introduce two new protocol classes, multiple multicast groups based and FEC-based. The former involves the use of multiple multicast groups for reducing receiver processing costs; the latter entails the use of forward error correction (FEC) coding associated to ARQ mechanisms to increase the scalability of reliable multicast protocols. In the next sub-sections, we first present the sender-initiated protocols and their optimizations. Following, the other five approaches are examined: receiver-initiated, hierarchical-based (or tree-based), ring-based, multiple multicast groups based and FEC-based protocols.

### 2.1 Sender-initiated protocols

Sender-initiated protocols offer reliable multicast by tracking the explicit membership of the receiver set and by providing retransmission for any lost or corrupted data. Therefore, the sender maintains the state of all the receivers to whom it has to send information and from whom it has to receive positive acknowledgments (ACKs). For each packet that each receiver obtains correctly, it sends a unicast ACK to the sender. Lost packets are managed through the use of timers. Specifically, the sender starts a timer at the time of a packet transmission and, if it expires prior to the sender having received ACKs for this packet from every single receiver, the sender retransmits the packet and restarts the timer. A sender-initiated protocol requires the source to receive ACKs from all the receivers, before it is allowed to release memory for the data associated with the ACKs. The source is in charge of deallocating memory after receiving all the ACKs for a given packet or set of packets [6].

The sender-initiated protocols can be optimized in numerous ways [9, 10, 11, 12, 13, 14]. A number of schemes have been proposed by [9] in the context of satellite broadcast channels. One is to retransmit the same packet (up to a limit) until positive acknowledgments are received from all receivers. Sending multiple copies of a packet all the time will severely reduce the source throughput, though, while sending them consecutively will not help if packet loss is due to congestion, and will in fact aggravate existing congestions. A modification was suggested where the transmitter looks for positive ACKs from receivers that did not previously send positive ACKs. This requires that the transmitter keep a list of all receivers at the transport layer. Reference [12] proposes to split the group into sub-groups. The size of the sub-group has to be adjusted to the maximum number of control messages that the sender can handle. The sender asks for acknowledgments from sub-groups and decides about the retransmission of messages once all members of each sub-groups have acknowledged a message. In [10], ACK aggregation for different packets into a single control packet is used.

In [11], a mechanism of ACK-avoidance, called damping and slotting, is used to reduce the number of ACKs received by the sender. The first mechanism, called slotting, is used by the receivers to distribute the acknowledgments over time. This is done by using a random time-out before sending an acknowledgment. Acknowledgments are always triggered by a control messages from the sender and are multicast to all members of the group. If a receiver gets a request for retransmission from another receiver for the same data, it refrains from sending its own status information. This second mechanism is known as damping. This strategy reduces the amount of reverse traffic from the receiver even with increasing group size. It is ensured that at least one receiver will reply and have a correct copy of the data. However, slotting and damping do not perform very well especially in a WAN environment. Further, this technique requires a multicast tree for each receiver if the underlying multicast routing mechanism uses source-based trees. In applications, such as video-on-demand, where the receivers interact rarely with the transmitter this would place an extra burden on the network routers. Reference [1] provides both a qualitative and a quantitative analysis of the damping and slotting technique. Pejhan *et al.* show that this scheme can be counterproductive in some cases. They also conclude that the benefits of this technique are more significant for schemes using positive acknowledgments. The quantitative analysis of this case shows that this scheme network can be detrimental in terms of cost to the network, when receivers are far apart and scattered around the network.

In [15] a new paradigm of communication (multicast-to-some) is defined to address scalability issues. It allows for a multicast message to be delivered to a random subset of the multicast group. This mechanism can be used by the multicast source to get replies from only a subset of destinations in the group. An acceptance probability $p$ is defined for each message sent by the source. It shows that with an appropriate setting of parameters the cost of multicast can be made to depend on number of desired responses rather than the size of the multicast group.

In the FRM algorithm [16] reliability is achieved by the use of a list-based group management scheme which is close to the one proposed by XTP 4.0 [17]. In this scheme a receiver group list is created from the positive replies of the receivers to a connection establishment request. In regular intervals the sender gathers control information during the communication from the receiver group to update this list. These intervals are among others determined by the sender buffer size and the maximum round-trip time. The current state of the multicast connection is determined by the sender according to the replies of the receivers. The sender keeps a copy of a transmitted packet until all receivers in the list have positively acknowledged its reception. If retransmission is requested, the packet will be multicast to the whole group. The maximum sustainable number of receivers is mainly restricted by two factors, namely limited buffer in the sender and the source implosion effect. Even if optimally implemented the size of the group is limited. Simulations results show that due to performance problems groups with more than 100 receivers can not be handled.

### 2.2 Receiver-initiated protocols

The receiver-initiated protocols place the responsibility for ensuring reliable packet delivery on the receivers. The sender continues to transmit new data packets until it receives a negative acknowledgment (NAK) from a receiver. When this occurs, the sender then retransmits the packet required by that receiver. The role of the receiver is to check for lost packets. Moreover, the receiver uses timers in a similar way that in the sender-initiated protocols. Because the source receives feedback from the receivers only when packets are lost and not when they are delivered, the source is unable to guaranty when it can safely release data from memory. There is no explicit mechanism in a receiver-initiated protocol for the source to release data from memory, even though its pacing and retransmission mechanisms are scalable and efficient. Levine *et al.* [6] demonstrate that the receiver-initiated protocols proposed to date require an infinite memory to work correctly. Specifically, this implies that the sender has to keep in memory every packet sent during all the lifetime of a multicast session. Hence, they cannot prevent deadlocks when they operate with finite memory. Another inherent drawback of the NAK based scheme is that it performs well only if the network does not reorder packets.

In the receiver-initiated protocols the sender is subject to the NAK-implosion. This can occurs when many receivers detect a packet loss, e.g. a packet lost in a common link of a multicast tree. The receiver-initiated protocols with NAK-avoidance gives a possible solution to this problem with an improvement of performance in some cases. This scheme is similar to the damping and slotting used in XTP3.6. In this protocol, the sender multicasts all packets and state information, giving priority to retransmissions. Whenever a receiver detects a packet loss, it waits for a random time period and then multicasts a NAK to the sender and all other receivers, and starts a timer. When a receiver obtains a NAK for a packet that it has not received and for which it has started a timer to send a NAK, the receiver starts the timer and behaves as if it had sent a NAK. The expiration of a timer without the reception of the corresponding packet is the signal used to detect a lost packet. With this scheme, it is hoped that only one NAK is sent back to the source for a lost transmission for an entire receiver set [8, 18]. Nodes farther away from the source might not even get a chance to request a retransmission. The NAK-avoidance scheme was originally designed for a limited scope, such as a LAN, or a small number of Internet nodes [6]. The main reason is that it requires timers to be set based on updates multicast by every node. With an increasing number of nodes, each node must do an increasing amount of work.

An example of receiver-initiated protocol is the Reliable Adaptive Multicast Protocol (RAMP) [19]. RAMP guarantees reliable and orderly delivery to all multicast recipients using immediate NAK-based scheme and a unicast error control feedback. This protocol selectively unicasts or multicasts a retransmission, depending on the number of received NAKs. It maintains explicit group membership at the sender and therefore does not scale very well for large groups. RAMP strictly follows the single sender multiple receiver paradigm (one-to-many), i.e. it uses a simplex model for multicast communications.

Floyd *et al.* [13] describe a framework for scalable reliable multicast (SRM) designed for shared whiteboard application (wb). Scalability is achieved by a receiver-initiated with NAK-avoidance protocol in conjunction with Application Level Framing (ALF). Algorithms for loss recovery (request/repair algorithms) are introduced. All receivers are able to retransmit since all data packets (application data, retransmission requests and all other data) are always sent to the entire group. SRM requires the application to be aware of the environment it is embedded in and that it is able to rebuild the data for retransmission. Further, an IP multicast distribution model is assumed that makes its implementation in an ATM environment difficult and costly. Thus, SRM is strongly focused on the specific environment it was designed for and on the kind of application it aims to support.

Reference [20] describes the MTP protocol that has a feature where a list of hosts that require to acknowledge packets is sent in the multicast packet. In both [21] and [20], the authors propose a threshold-based scheme for retransmissions in order to achieve a trade-off between unicast (which is time consuming and reduces source throughput) and multicast (which unnecessarily increases load on those networks with no packet losses) The threshold is based on the number of hosts sending NAKs for a given packet (unicast if less, multicast if more), although no specific value for, or means of deriving, the threshold is given.

The Log-Based Receiver-Reliable Multicast (LBRM) [22], initially conceived to support DIS applications, uses a hierarchy of log servers that store information sent by the source and receivers recover by contacting a log-server. However, the use of a single server can lead to the degradation of performance due to the load at this single point, and multiple servers require extra and complex mechanisms to guarantee the consistence of information between them.

The reliability semantic provided by the SPRM (Scalable Probabilistic Reliable Multicast) algorithm [23] is different from the full reliability which is achieved by using a list-based approach. This algorithm was inspired by the bucket algorithm of XTP 3.6 [11]. However, it differs substantially in a number of ways. The most important difference is that it does not assume that receivers have knowledge of each other or that replies to a control message are transmitted to the whole group. However, full reliability is not achieved, i.e. there is a small probability that some errors can not be recovered. In contrast to a list-based reliable scheme the decision to (re-)transmit data packets is based on partial knowledge of the receivers' state rather than the full, global knowledge of their state. The philosophy behind this reliability semantic is that it is the receivers' responsibility to detect and recover from packet loss, i.e. a receiver that detects an error will send a negative acknowledgment (NAK) to the sender. In conjunction with this, a sender-based acknowledgment strategy is used, i.e. the sender asks for status information to out-date buffers containing transmitted packets.

The sender keeps a number of so called buckets represented by the bucket list. Periodically the sender sends a control message, which is associated with a bucket, asking for replies from the receivers. Data is only retransmitted when a bucket is out-dated, i.e. a negative acknowledgment is not served immediately. The retransmission is optimized because requests are served with just one retransmission when a bucket is recycled. To avoid ACK-implosion and improve scalability, a back-off algorithm is used to reduce the number of echo messages. According to this algorithm a receiver does not reply to each control packet. The full set of receivers will only reply to the first control packet. A receiver will reply to the second control packet with a probability of 0.5, to the third with a probability of 0.25, etc. To ensure that in the average case at least one receiver will reply to each control packet the probability with which a receiver replies should never be less than $\frac{1}{2^k}$ where $k = \ln_2 N$. The receivers are informed about the value of $N$ by the sender that estimates the current number of receivers in the multicast session. This estimate is based on information gained during connection establishment and from leaving and joining receivers. Whenever a receiver has to send a negative acknowledgment the probability with which it replies to the next control packet is reset to one. This mechanism also ensures that a receiver replies on average to one in every $2^k$ control packets and does not fall completely silent. There may be a situation where a negative acknowledgment arrives after the buffers of the respective packet have been released. In this case the sender signals to the respective receiver that this error can not be recovered. Because of these non-recoverable errors this reliability mechanism can not provide a full reliable service, it offers a probabilistic reliability. SPRM guarantees that at least one receiver has received a correct copy of the message and that the sender will at least get one reply from the group.

The first comparative analysis of sender-initiated and receiver-initiated reliable multicast protocols was presented by Pingali *et al.* [8, 18]. Three generic error control protocols were studied, one sender-initiated (ACK-based) and two receiver-initiated (NAK-based). The first receiver-initiated protocol (N1) exhibits the following behavior. The sender multicast all packets. Whenever a receiver detects a lost packet, it transmits a NAK to the sender over a point-to-point channel and starts a timer. The expiration of the timer without prior reception of the corresponding packet serves as the detection of a lost packet. The second receiver-initiated protocol (N2) uses a variation suggested by Ramakrishnan and Jain [24] for a LAN and Floyd [13] for a WAN to reduce the number of NAKs (NAK-avoidance). The behavior of this generic protocol is the same of that explained above for a receiver-initiated protocol with NAK-avoidance. The analysis of these protocols focuses on host processing as the constrained resource instead of network bandwidth. They consider that the network bandwidth that continues to increase is becoming of less importance. In their analysis they consider two different applications, one-to-many and many-to-many applications. In the former one sender transmits a continuous stream of packets to $R$ receivers. In the latter all participants send as well as receive data reliably. In both cases, they assume that all loss events at all receivers for all transmissions are mutually independent and that the probability of packet loss is independent of receiver. They also assume that ACKs and NAKs are never lost. Pingali *et al.* provide a quantitative demonstration of the superiority of receiver-initiated approaches over sender-initiated approaches. Further, they found that among the two receiver-initiated protocols N2 performs better in the one-to-many scenario while N1 was superior in the many-to-many case. Levine *et al.* [6] address the correctness of the various classes of reliable multicast protocols and extends the analysis by Pingali *et al.* [8, 18] by analyzing the maximum throughput of the different protocol classes.

### 2.3 Hierarchical or tree-based protocols

The hierarchical (or tree-based [6]) approach is characterized by dividing the receiver set into groups, and distributing retransmission responsibility over an acknowledgment tree (ACK tree) structure. A generic protocol described in [6] assumes that each subgroup consists of $B$ children and a group leader. Each child can be a group leader of a subgroup forming a multilevel hierarchy. Acknowledgments from children in a group are sent only to the leader. A child in a group sends its acknowledgment to its parent as soon as it receives a correct packet, not when all its own children (if any) have sent their acknowledgment. These acknowledgments differ from ACKs and NAKs used in the other approaches and they are referred as hierarchical acknowledgments (HACKs). The use of HACKs is important for throughput since the source do not have to wait for ACKs to be aggregated all the way from the leaf nodes.

Hierarchical acknowledgments are sent in a tree composed of up to three types of nodes : a source node, leaf nodes and hop nodes. The source node, or sender, is the originator of a new packet, which it multicasts to all the receivers. As any group leader, this node has at most $B$ children from which to process HACKs and to send retransmissions. Leaf nodes are at the bottom of the tree and are not responsible for any children. They play the same role as receivers in the sender-initiated protocol, except that they send their HACKs only to their group leaders instead of sending ACKs to the source. Finally, the hop nodes are group leaders in between the source and leaf nodes. They send HACKs to their own group leaders one step higher in the up tree, and they collect HACKs from the children in their group, retransmitting if necessary. Leaders of subtrees hold the decision of when to delete packets from memory depending upon the receipt of HACKs from the children in he group.

Different reliable multicast protocols were proposed following the hierarchical approach [10, 25, 26]. The most early work in this area was presented in [10] which compares three basic schemes for reliable point-to-multipoint multicasting using the hierarchical approach. In [25] a new concept, named Local Group Concept (LGC), is presented. Here, global multicast groups are divided into separate subgroups. The receivers which pertain to the same subgroup should be located in close physical vicinity. In each subgroup a Local Controller is responsible for collecting status information from its receiver group and informing the sender by a single composite message to prevent source implosion. The Local Controller is also responsible for retransmission within the local group. Data that has to be retransmitted is obtained either from the local receivers or the original sender.

RMTP (Reliable Multicast Transport Protocol) [26] achieves reliability by using a packet-based selective repeat retransmission scheme and a multilevel hierarchical approach. The basic ideas of RMTP are derived from the Designated Status Protocol (DSP) described in [10]. Here, receivers are grouped into a hierarchy of local regions, with a Designated Receiver (DR) in each local region. DRs cache received data and respond to retransmission requests of the receivers in their local regions (figure 1). To conserve bandwidth RMTP receivers send HACKs periodically to their corresponding DR instead of sending acknowledgments for each individual packet. This periodic sending was adopted from [27, 10]. DRs send HACKs to the higher level DRs, until the DRs in the highest level send HACKs to the sender. Each HACK carries a sequence number $L$ and a bitmap $V$. Sequence number $L$ indicates that a receiver has correctly received all the packets with sequence number less than $L$. A 0 in the bitmap corresponds to a missed (or incorrectly received) packet while a 1 corresponds to a correctly received packet.

Hierarchical protocols free the source from having to know the entire receiver set, work with finite memory, provide better end-to-end delays, and operate solely on message exchanges within local groups [6]. Levine *et al.* show that the amount of work required at each node for hierarchical protocols does not increase with the number of receivers, i.e. the throughput of such protocols is not dependent on the number of group members. However, these protocols present some drawbacks. Both [26]and [25] assume that it is possible to retrieve data required for retransmission from the receivers as well as from the sender. Further, at least some dedicated receivers are aware of other receivers and the structure of the multicast group. It assumes that some nodes will agree to serve as Designated Receivers and provide such a service to the end users. To exploit their full potential it is also necessary to have knowledge about the underlying network. Further, it requires an

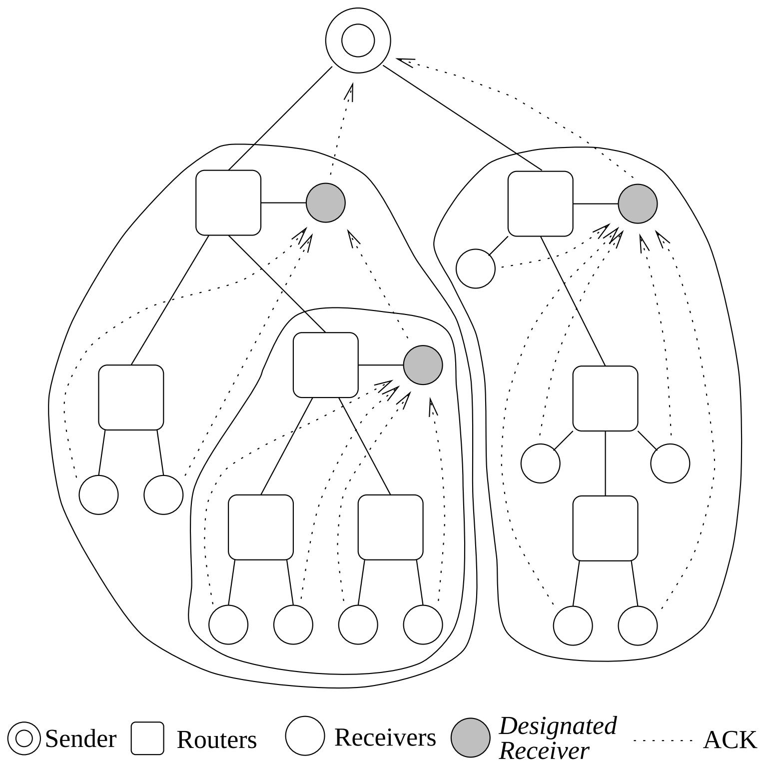


Figure 1: RMTP multilevel hierarchy.

important signaling and supposes a close cooperation between the participants. Also, it assumes a multicast distribution model many-to-many as provided by IP multicast.

## 2.4 Ring-based protocols

Another class of multicast reliable protocols introduced by [6] is ring-based protocols that were originally developed to guarantee atomicity and total ordering of transmissions at all receivers. Basically, two proposals exist in this class, the Token Ring Protocol (TRP) [28] for LAN, and the Reliable Multicast Protocol (RMP) [29] for WAN. The basic ideas of these protocols are 1) to have a single token site responsible for positively acknowledging packets and 2) receivers sending NAKs to the token site for selective repeat of lost packets. Thus, ring-based protocols combine the throughput advantages of NAKs with the reliability of ACKs. ACKs are multicast from the token site and serve also to timestamp packets to guarantee global ordering on the delivery of packets to the application. In both, TRP and RMP, retransmissions are unicast from the token site and therefore limit their throughput. Passing the token to the next member of the ring of receivers is only possible when the next site has correctly received all packets that the token site has received.

## 2.5 Multiple multicast groups based protocols

This approach presented in a recent work by Kasera *et al.* [30], involves the use of multiple multicast groups for reducing processing costs in a multicast session. Here, a single multicast group is used for the original transmission of packets and retransmissions are done to separate multicast groups, which receivers dynamically join or leave. The basic idea in using multiple multicast groups is to avoid unnecessary receiver processing overhead imposed by the reception of retransmissions of packets already correctly received. The behavior of a generic protocol based in multiple multicast groups is as follows [30]. Considering a system with a sender and $R$ receivers such that data is transmitted reliably from the sender to the receivers using $G+1$ channels. One of the channels is used for the original transmission of packets from the sender. The remaining $G$ channels are mapped to ranges of sequence numbers such that a lost packet is retransmitted on the channel corresponding to the range to which it belongs. A receiver, after detecting a packet loss, joins the appropriate channel for that packet's retransmission. Once the lost packet is received, the receiver leaves the appropriate channel.

The analysis presented in [30] first shows that protocols using an infinite number of multicast groups incur much less processing overhead at the receivers compared to protocols that use only a single multicast group. In a second time, they derive the number of unwanted packet at a receiver due to using only a finite number of multicast groups for a receiver-initiated with NAK-avoidance protocol using multiple multicast groups. Then, they explore the minimum number of multicast groups required to keep the cost of processing unwanted packets to a sufficiently low value. The results show that only a small number of multicast groups are required for a one-to-many applications and a moderate number for a many-to-many applications.

The major drawback of this approach is the processing of joins and leaves required to add and delete receivers from multicast groups. In the IP multicast routing, each join and leave message results in processing overhead at all intermediate routers. This operation can shift the burden from the receivers to the underlying network negating all the benefits gained with this approach. Kasera *et al.* propose the use of a receiver filtering mechanism that aims to avoid the processing inside the network, bringing the overhead to the network interface hardware of the receiver. This solution is specific to the IP multicast protocol and its overhead was not assessed by the authors.

An earlier example of multiple multicast groups based protocol is the work presented by Ammar *et al.* in [31]. In this work, they propose the idea of destination set splitting for improving the throughput of some specific positive acknowledgment based point-to-multipoint protocols. This work differs from Kasera's work since they suggest that receivers could be divided into groups based on their capabilities and the sender carries out as many simultaneous independent conversations as the number of groups.

## 2.6 FEC-based protocols

Forward Error Correction (FEC) coding is a commonly cited technique for lowering the impact of packet losses incurred in real-time transmission. Several research efforts [32, 33] advocate the use FEC coding for Internet audio. With FEC, the sender prevents losses by transmitting some amount of redundant data, called parity data, along with the original data that allows for the reconstruction of missing data at the receivers. Depending on the amount of redundancy, the residual packet loss rate can be made arbitrarily small, to the point that reliable transfers can be achieved without the need for a feedback channel. Alternatively, one can just be interested in a reduction of the residual loss rate, so that performance is generally improved but feedback from the receiver is still needed. Specifically, if the amount of original data lost is not more than the amount of parity data sent, the parity data can be used to reconstruct the lost original data. Besides reducing the time needed to recover the missing packets, this technique is very attractive for multicast applications since different loss patterns can be recovered using the same set of transmitted data. Multiple receivers can experience losses on different packets, and insuring reliability via individual repairs might become extremely expensive. A second advantage derives from the reduced need of handling a feedback channel from the receivers when full reliability is not needed, i.e. a small residual packet loss rate is acceptable. This is an extremely useful feature since it allows protocols to scale well to large number of receivers. In [5] Nonnenmacher *et al.*, which investigates how FEC can be combined with ARQ (Automatic Repeat Request) to guarantee full reliability, give two scenarios for the design of FEC-based protocols. In the first, FEC is introduced as a transparent layer underneath a reliable multicast layer that uses ARQ; in the second, FEC and ARQ are both integrated into a single layer that uses the retransmission of parity data to recover from the loss of original packets. The former scenario is called *layered FEC*; the latter is referred as *integrated FEC*.

FEC is generally based on the use of error control codes (ECC). The main requirement for such codes is the ability to recover from errors, i.e. wrong bit patterns in unknown positions. In the context of computer communications, error detection is generally provided by the lower layer protocols, and corrupted packets are discarded. The upper layer protocols thus only have to deal with *erasures*, i.e. missing data in a stream. Erasures are easier to deal with than errors, since the exact position of missing data is known. Reference [34] provides a simplified description of the basic operation of erasure codes.

A number of theoretical studies on the effectiveness of FEC in communication protocols has appeared in recent years [35, 36, 37, 5] dealing with general applications or specific protocols (e.g. IP over ATM), and both unicast and multicast applications. There is general consensus that FEC can be profitably used, although the issue of doing encoding/decoding in software, and the associated overhead, has not been so far much studied. In a recent work presented by Rizzo [34] a C implementation of an erasure code that is able to run at very high speed (up to 12 MBytes/s in a Pentium 133 MHz) is introduced. This public implementation shows the feasibility of FEC software even on desktop PCs. Such performance and the fact that the bandwidth requirements are typically less than $100KBytes$ suggests that coding will not affect packet sending rate [5]. Reference [37] gives a list of multicast applications that will benefit from the use of a redundant encoding.

An early paper by Metzner [35] studies a hybrid ARQ for multicast reliable protocols that uses Reed Solomon codes for parity computation. The analysis aims to quantify the benefits of the scheme in terms of the mean number of parity packets transmitted in the first retransmission round for a small number (up to 50) of receivers. Reference [36] looks at the performance of FEC when used in a layered FEC scenario for the case of independent and homogeneous loss. This work shows the benefits of layered FEC in terms of reducing network resource usage.

The extensive analysis by Nonnenmacher *et al.* [5] considers spatially or temporally correlated loss, homogeneous or heterogeneous loss, and loss rates experienced when facing up to $10^6$ receivers. Their results indicate that integrated FEC shifts resource usage from the network to the endsystems, i.e. the number of transmissions is reduced and therefore the network bandwidth used as well - at the cost of coding at the endsystems. Further, the traffic at the feedback channel is reduced as expected. Integrated FEC is better than layered FEC for all parameters. The bottleneck is identified to be the sender for integrated FEC protocols. Pre-encoding or a powerful sender machine will lead to an endsystem throughput up to 3 times higher than the one achived with non-FEC protocols, even when receivers are able to decode online. However, in the case of burst loss, layered FEC protocols are worse than non-FEC. The calculated processing rates at the sender shows that the scalability is achieved for large number of receivers up to 1 million. With respect to the loss distribution, high loss receivers determine the performance, even if they are a small subset of all receivers. With shared loss the whole repair efficiency of FEC is not used in the same extent than in the case of independent loss. This shows that the effectiveness of FEC-based protocols is sensible to loss distribution in the network. The mismatch between the sender and receiver processing rates for integrated FEC protocols raises some issues as the use of pre-encoding or dedicated hardware at the sender.

# 3 Some considerations

In this section we present some other important issues related to the scalability of reliable multicast protocols. With respect to the underlying network, the multicast distribution model provided and the packet loss distribution observed are of great importance to the performance achieved by the multicast reliable protocols. Also, the presence of *misbehaving receivers* within the receiver set has a great effect on the maximum throughput achieved by some of the protocol classes described in the previous section.

## 3.1 Multicast distribution model

In a heterogeneous communication environment a number of different networks are connected ranging from LANs to WANs and high-speed networks such as ATM. Many LANs and MANs (e.g. Token Ring, Ethernet, FDDI, DQDB) support multicast based on their inherent broadcast characteristics. In contrast to the distribution group model supplied by IP multicast (multicast group abstraction), ATM [38] provides unidirectional point-to-multipoint connections where a multicast tree is created with a single sender as the root node and receivers as leaf nodes (1-N multicast tree). In this model the receivers do not have any knowledge of each other and there is no communication among them. This communication model can be regarded as a subset of the more general model proposed by IP multicast where everybody can send to the group address and every member is receiving messages addressed to the group. The distribution model of IP multicast can actually be achieved at the ATM layer by using a multicast server [39]. However, this solution adds complexity to the ATM model. It may also prevent the from utilizing the QoS capabilities provided by ATM networks.

The design of a multicast reliable protocol has to take into account both the multicast distribution model supplied by the underlying network and the type of application supported, i.e. one-to-many or many-to-many application. In fact, to provide data transfer reliability in an heterogeneous environment, a multicast reliable protocol is necessary that can easily be supported over ATM style networks as well as over meshed networks like the Internet. One example of this problem is where one-to-many applications are supported over a network that provides a one-to-many distribution model such as ATM. Here, the need of a feedback channel from the receivers which span all members of the receiver group can be very costly in terms of signaling and management overhead. One-to many applications can be exemplified by a telelecture or a teleconference where one participant acts as a sender all of the time.

It appears that providing Quality of service guarantees and multicasting at the same time is a challenge whatever the technology being considered.

## 3.2 Packet loss distribution

A critical issue for reliable multicast protocols built upon wide area multicast communication services is the manner in which packet losses occur within the multicast network. The underlying packet loss process is of tremendous importance to error control protocols. An important work in this area was presented in [40] that analyze packet loss data collected on multicast-capable hosts geographically distributed over the MBone. In this work, Yajnik *et al.* aim to examine the spatial and temporal correlation in packet loss among participants in a multicast session. By spatial correlated loss, they mean the lack of reception of the same packet at many sites, and temporally correlated loss corresponds to the loss of consecutive packets at a given receiver. Their results show that the loss on the backbone links of the multicast transmission tree was small relative to the total loss seen by the receivers. The spatially correlated loss was small, on the average, except for the loss due to the link next to the source. With respect to temporally correlated losses, they found that a majority of the loss bursts were solitary losses. A few extremely long loss bursts greater than 8 sec. (or 100 packets) were also observed. A similar work by Bolot *et al.* [41] presents the packet loss measurements from a 10,000 packet trace between MBone sites in France and England. With respect to temporally correlated loss, they find also that losses appear to be isolated.

Bhagwat *et al.* [7] describes a recursive analytic method for computing the probability that a packet is not received at one or more receivers given a specific multicast tree and known, independent loss probabilities on each link. They use a block based acknowledgment and selective retransmission protocol to evaluate the impact of the loss rate and the multicast tree topology on the achievable throughput. Their results shows that even when the buffer overflow probability at switches and receivers is low, the cumulative loss probability seen by a source can be quite high.

A more extensive study to determine the loss patterns observed in networks such as the MBone would be of great value to assess the performance of reliable multicast protocols. This would show which aspects of the loss affect the most their performance. To give an of how deeply the loss distribution can influence the performance of a reliable multicast protocol, we consider a network with very long bursts. This loss pattern could promote the use of a go-back-n retransmission rather than a selective retransmission scheme due to the processing overhead at the receiver imposed by the latter. Therefore, one need to be cautius when considering the behavior of the underlying network, whose characteristics might also change overtime.

## 3.3 Misbehaving receivers

Another issue frequently addressed in this context is the problem of misbehaving receivers that jeopardize the success of the data transmission because of either their inability to process data at a given rate (slow receivers) or their presence in congested links. Such receivers can slow down the communication considerably by sending delayed acknowledgments and requesting frequent retransmissions. For example, they may constantly bother the rest of the multicast group by multicasting NAKs. This problem is often referred as *crying baby problem.* Even worse, if a receiver falls behind the others in the sequence space, the sender has to keep up with the pace of this slow receiver to guarantee a full reliability. This causes a decrease of the end-to-end throughput noticed by all receivers. The scenario of a slow receiver within a group is the one that is most likely to occur due to the different receiver capabilities and resources. A flexible method to deal with this problem is required.

A new concept in multicast communication, called Active Group Integrity (AGI), is to define a set of mandatory receivers or a minimum number of receivers that have to be present in order to consider the communication successful [42]. This concept is closely related to the application semantic. It should be possible to specify a (sub-)set of receivers that are compulsory for the success of the communication and hence determine its outcome. Others, less important participants can also have less strict reliability requirements. None of the introduced algorithms so far, apart from [16], addresses the problem of misbehaving receivers and their consequences on the performance of multicast reliable protocols.

## 4 Conclusions

Group communication and multicast entails a range of new problems affecting all components of the architecture. A problem in this context is how to provide fully reliable data transmission to a large number of receivers while still fully exploiting the advantages of multicast communication and maintaining a suitable performance in a heterogeneous environment. Most conventional multicast reliable protocols require the sender to keep tracking of the receiver set over time and make retransmission of missing and corrupted data. With an increasing number of receivers the processing overhead at the sender increases as well. An efficient, scalable way to provide full reliability for group applications in different environments is needed.

In this paper we present different classes of multicast reliable protocols according to a taxonomy and discuss how well each of them handle large number of receivers. Six classes of reliable multicast protocols are examined: sender-initiated, receiver-initiated, tree-based, ring-based, FEC-based and multiple multicast groups based protocols. Levine *et al.* [6] show that, although the maximum throughput of receiver-initiated, tree-based and ring-based protocols are all independent of the number of receivers as the probability of errors goes to zero, only tree-based protocols can be completely scalable under any condition, i.e., when the probability of error is non-negligible. FEC-based protocols are in a general consensus an attractive method for scaling purposes on multicast reliable protocols, though the feasibility of FEC software has so far impacted the widespread use of FEC techniques in the context of computer communications. However, recent works on this area put FEC software in a right perspective by showing that it is not exceedingly expensive [34]. Multiple multicast groups based protocols give a new outlook to the problem of scalability, although some issues, such as filtering and signaling overhead, have to be further explored.

The performance achieved by a reliable multicast protocol does not only depend on the mechanisms used but also on the loss patterns observed and on the multicast distribution model supported by the underlying network. Considering all these factors, a good perspective to correctly assess the performance of reliable multicast protocols, and also for comparison purposes, is the use of traces during multicast sessions directly in a simulation of a multicast network [40]. The misbehaving receivers have as well a bad impact on the performance of these protocols. New mechanisms have to be conceived to cope with this problem and the AGI concepts looks as a promising way to explore. Finally, solutions for multicast flow and congestion control issues are also needed.

## References


[1] S. Pejhan, M. Schwartz, and D. Anastassiou, "Error Control Using Retransmission Schemes in Multicast Transport Protocols for Real-Time Media," *IEEE/ACM Transactions on Networking*, vol. 4, pp. 413–427, June 1996.

[2] E. W. Biersack, "Performance Evaluation of Forward Error Correction in an ATM Environment," *IEEE Journal on Selected Areas in Communications*, vol. 11, no. 4, pp. 631–640, 1993.

[3] B. Lamparter, O. Boehrer, W. Effelsberg, and V. Turau, "Adaptable Forward Error Correction for Multimedia Data Streams," in *Reihe Informatik 9/93*, (Universiteat Mannheim, Mannheim), 1993.

[4] V. Hardman, M. Sasse, M. Handley, and A. Watson, "Reliable Audio for Use over the Internet," in *INET'95*, 1995.

[5] J. Nonnenmacher, E. Biersack, and D. Towsley, "Parity-Based Loss Recovery for Reliable Multicast Transmission," in *Proceedings of ACM SIGCOMM'97*, (Cannes, France), Sept. 1997.

[6] B. N. Levine and J. J. Garcia-Luna-Aceves, "A Comparison of Known Classes of Reliable Multicast Protocols," in *Proceedings of International Conference on Network Protocols*, (Columbus), Oct. 1996.

[7] P. Bhagwat, P. P. Mishra, and S. K. Tripathi, "Effect of Topology on Performance of Reliable Multicast Communication," in *Proceedings of IEEE INFOCOM'94*, (Toronto, Canada), pp. 602–609, June 1994.

[8] S. Pingali, D. Towsley, and J. Kurose, "A Comparison of Sender-Initiated and Receiver-Initiated Reliable Multicast Protocols," in *ACM SIGMETRICS'94*, (Santa Clara, CA), pp. 221–230, 1994.

[9] K. Sabnani, "Multidestination Protocols for Satellite Broadcast Channels," Tech. Rep. Ph.D. thesis, Columbia University, 1982.

[10] S. Paul, K. Sabnani, and D. Kristol, "Multicast Transport Protocols for High-Speed Networks," in *Proceedings of IEEE Int. Conference on Network Protocols*, (Boston, USA), Aug. 1994.

[11] W. T. Strayer, B. J. Dempsey, and A. Weaver, *XTP: The Xpress Transfer Protocol*. Addison and Wesley, 1992.

[12] L. Hughes and M. Thomson, "Implosion-Avoidance Protocols for Reliable Group Communications," in *Proc. of 19th Conference on Local Computer Networks*, (Minneapolis, USA), Oct. 1994.

[13] S. Floyd, V. Jacobson, S. McCanne, L. Zhang, and C.-G. Liu, "A Reliable Multicast Framework for Light-weight Sessions and Application Level Framing," in *Proceedings of ACM SIGCOMM'95*, (Cambridge, USA), Aug. 1995.

[14] D. Towsley, "An Analysis of a Point-to-Multipoint Channel Using a Go-Back-N Error Control Protocol," *IEEE Transactions on Communications*, vol. 33, pp. 282–285, Mar. 1985.

[15] M. H. Ammar, "Probabilistic Multicast: Generalizing the Multicast Paradigm to Improve Scalability," in *Proceedings of IEEE INFOCOM'94*, (Toronto, Canada), pp. 848–855, June 1994.

[16] J. F. de Rezende, A. Mauthe, and S. Fdida, "La Communication Multimédia de Groupe de Bout-en-Bout," in *Proc. of Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'96)*, (Rabat, Maroc), Oct. 1996.

[17] XTP Revision 4.0, "Xpress Transport Protocol Specification," in *XTP Forum*, (Santa Barbara, CA), Mar. 1995.

[18] S. Pingali, D. Towsley, and J. Kurose, "A Comparison of Sender-Initiated and Receiver Initiated Reliable Multicast Protocols," *IEEE Journal on Selected Areas in Communications*, vol. 15, Apr. 1997.

[19] A. Koifman and S. Zabele, "RAMP: A Reliable Adaptive Multicast Protocol," in *Proceedings of IEEE INFOCOM'96*, (San Francisco, CA), Mar. 1996.

[20] J. Crowcroft and K. Paliwoda, "A Multicast Transport Protocol," *ACM Computer Communication Review*, vol. 18, pp. 247–256, Aug. 1988.

[21] R. Braudes and S. Zabele, "Requirements for Multicast Protocols," RFC 1458, May 1993.

[22] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton, "Log-Based Receiver-Reliable Multicast for Distributed Interactive Simulation," in *Proceedings of ACM SIGCOMM'95*, (Cambridge, USA), Aug. 1995.

[23] J. F. de Rezende, A. Mauthe, S. Fdida, and D. Hutchison, "Fully Reliable Multicast in Heterogeneous Environments," in *Proc. of Protocols for High-Speed Networks (PfHSN'96)*, (Sophia Antipolis, France), Oct. 1996.

[24] S. Ramakrishnan and B. N. Jain, "A Negative Acknowledgement with Periodic Polling Protocol for Multicast over LAN," in *Proc. IEEE INFOCOM '87*, pp. 502–511, Mar. 1987.

[25] M. Hofmann, "A Generic Concept for Large-Scale Multicast," in *Proceedings of International Zurich Seminar on Digital Communication (IZS'96)*, (Zurich, Switzerland), Feb. 1996.

[26] J. C. Lin and S. Paul, "RMTP: A Reliable Multicast Transport Protocol," in *Proceedings of IEEE INFOCOM'96*, (San Francisco, CA), pp. 1414–1424, Mar. 1996.

[27] A. N. Netravali, W. D. Roome, and K. Sabnani, "Design and Implementation of a High-Speed Transport Protocol," *IEEE Transactions on Communications*, vol. 38, Nov. 1990.

[28] J.-M. Chang and N. F. Makemchuk, "Reliable Broadcast Protocols," *ACM Transactions on Computer Systems*, vol. 2, pp. 251–273, Aug. 1984.

[29] B. Whetten, S. Kaplan, and T. Montogomery, "A High Performance Totally Ordered Multicast Protocol," available from ftp:
research.ivv.nasa.gov/pub/doc/RMP/RMP_dagstuhl.ps, Aug. 1994.

[30] S. Kasera, J. Kurose, and D. Towsley, "Scalable Reliable Multicast using Multiple Multicast Groups," in *Proceedings of 1997 ACM Sigmetrics Conference*, 1997.

[31] M. H. Ammar and L. Wu, "Improving the Throughput of Point-to-Multipoint ARQ Protocols through Destination Set Splitting," in *Proceedings of IEEE INFOCOM'92*, (Florence, Italy), pp. 262–269, June 1992.

[32] J.-C. Bolot and A. V. Garcia, "The Case for FEC-based Error Control for Packet Audio in the Internet," *ACM Multimedia Systems*, 1997.

[33] M. Podolski, "A Study of Speech/Audio Coding on Packet Switched Networks," Tech. Rep. M96/6, UC-Berkeley/ERL, Dec. 1996.

[34] L. Rizzo, "On the Feasibility of Software FEC," Tech. Rep., Universitá di Pisa, Jan. 1997.

[35] J. Metzner, "An Improved Broadcast Retransmission Protocol," *IEEE Transactions on Communications*, vol. 32, pp. 679–683, June 1984.

[36] C. Huitema, "The Case for Packet Level FEC," in *Proc. of Protocols for High-Speed Networks (PfHSN'96)*, (Sophia Antipolis, France), Oct. 1996.

[37] L. Rizzo, "Effective Erasure Codes for Reliable Computer Communication Protocols," Tech. Rep., Universitá di Pisa, Jan. 1997.

[38] *UNI 3.1 - User Network Interface.* Prentice Hall, 1994.

[39] G. Armitage, "Support for Multicast over UNI 3.0/3.1 based ATM Networks," RFC 2022, Nov. 1996.

[40] M. Yajnik, J. Kurose, and D. Towsley, "Packet Loss Correlation in the MBone Multicast Network," in *IEEE Global Internet Conference*, (London, UK), Nov. 1996.

[41] J.-C. Bolot, H. Crepin, and A. Garcia, "Analysis of Audio Packet Loss in the Internet," in *Proc. 1995 Workshop on Network and Operating System Support for Digital Audio and Video*, (Durham, New Hampshire), Apr. 1995.

[42] L. Mathy, G. Leduc, O. Bonaventure, and A. Danthine, *A Framework for Group Communications.* O. Spaniol, W. Bauerfeld and F. Williams (eds.), Elsevier Science Publishers (North-Holland), Hamburg, June 1994.