A Reliable Multicast Using the Xpress Transfer Protocol

Bert J. Dempsey
Alfred C. Weaver

Computer Science Report No. TR-91-02
January 1991

A Reliable Multicast Using the Xpress Transfer Protocol

Bert J. Dempsey and Alfred C. Weaver

Department of Computer Science
Thornton Hall
University of Virginia
Charlottesville, Virginia 22903
(804) 924-7605
bjd7p@virginia.edu, weaver@virginia.edu

1. Types of Multicast Service

Multicast services needed for current distributed applications on LANs fall generally into one of three categories: *datagram*, *semi-reliable*, and *reliable* (Figure 1). Transport layer multicast datagrams represent unreliable service in which the transmitting context 'fires and forgets'. XTP executes these semantics when the MULTI and NOERR mode bits are both set. Distributing sensor data and other applications in which application-level error recovery strategies are appropriate benefit from the efficiency in multidestination delivery offered by *datagram* service. *Semi-reliable* service refers to multicasting in which the control algorithms of the transport layer — error, flow, and rate control — are used in transferring the multicast distribution to the set of receiving contexts, the *multicast group*. The multicast defined in XTP provides *semi-reliable* service. Since, under a *semi-reliable* service, joining a multicast group means listening on the group address and entails no coordination with other members, a *semi-reliable* facility can be used for communication between a client and a server group as well as true peer-to-peer group communication. Resource location in a LAN is an important application domain. The term 'semi-reliable' refers to the fact that group membership changes go undetected. No attempt is made to assess the current membership of the group at any time — before, during, or after — the data transfer.

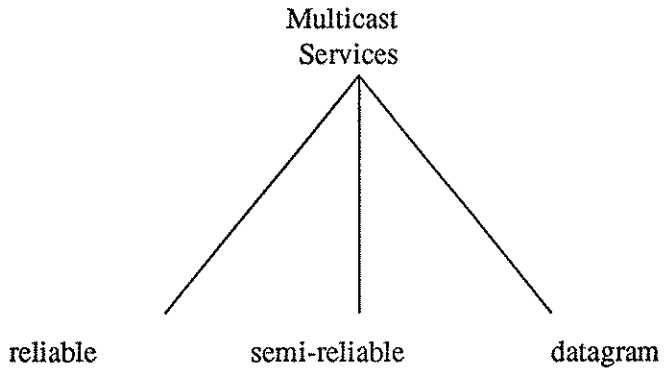

Figure 1 — Multicast Services

Applications that require a strong degree of data and behavior consistency between the process group members such as distributed databases, replicated program execution modules, and automated control programs need *reliable* multicast service. These applications use peer-to-peer communication to update global group state or data copies. The desired multicast semantics include (1) delivery of each message to either all or none of the operational processes, (2) receipt of messages in the same order by all group members, and (3) detection of group membership changes and reporting of these changes as they occurred relative to the message delivery stream.

Various research efforts address the need for *reliable* multicast. Message ordering and reliability can be achieved by having a single group member ([1]) or special system node ([2]) that assigns a sequence number to all messages to the group. The sequencer takes on the responsibility of holding past messages in a *history buffer* and retransmitting to group members that detect lost messages through discontinuities in the sequence of message numbers. Tseung ([3]) uses a systems hardware solution by adding special network nodes that perform the sequencing and logging tasks, thus addressing connectivity errors as well. In [4] a token circulates between group members. The token-based scheme permits all of the requirements of

reliable multicast service to be met as well as preserving atomic delivery semantics during network partitions and consistent message sequencing when process groups overlap. Finally, Birman and Joseph [5] [6] explore a range of ordering properties for distributed applications and propose a two-phase commit protocol for the FIFO ordering needed by *reliable* multicast.

2. Name Space Management

All group communication schemes need a Directory Services entity to manage the name space for multicast groups. Providing this support in a distributed fashion can be accomplished in the following fashion. The local name manager entity receives a group creation call from the user and hashes into the set of available addresses for an address to associate with the new group. The manager then broadcasts a request to the well-known address of all name managers and waits for any reply indicating a collision. This broadcast is performed some number of times for reliability before the local manager decides that the address can be safely assigned to the new group. The group creator becomes the first member of the new group, and the new group address is broadcast to the set of local managers. For real-time systems where the long latency of group creation may be unacceptable, a (human) user-managed table of 'permanent groups' should be available to shortcircuit run-time verification of address uniqueness.

3. A Reliable Multicast Service Layered onto XTP

XTP can provide datagram and semi-reliable multicast services. The full functionality for a reliable multicast service, however, clearly lies outside the realm of a transport layer protocol. Below is an outline of how a reliable multicast service can be layered onto XTP. Using the extra channel available through tagged data (i.e. data carried in BTAGs and ETAGs) a higher layer *reliable multicast protocol* (RMP) can be constructed. The hook into XTP that would enable this protocol is a control bit associated with tagged data fields indicating that these fields are carrying RMP information.

Figure 2 shows one of the four members of a multicast group transmitting to the group. The RMP entity at the transmitter manages reliability for the one-to-many (forward direction) data flow via control information solicited from and sent by remote RMP entities. If the number of remote listeners is known at the transmitter (which RMP guarantees), then the local instance of RMP sets up and manages that number of unicast connections for reverse direction data flow. Each listener establishes a reverse direction connection, which is an ordinary XTP unicast connection. In both the forward and reverse directions, RMP entities multiplex user data and RMP control information (in the tagged data channel) over a single connection.

Message serialization under RMP is trivial since only one member can be transmitting messages at any time. Since each transmission requires the participation of all group members, the n-way connection set-up described here serves as an exclusive lock on the group. The state of message delivery to the group is available in the progress of the current transmitter's one-to-many connection. This eliminates the need for an explicit global ordering of messages to the group and simplifies the task of identifying messages during failure recovery.

RMP can achieve atomic delivery of messages by performing a two-phase commit. The RMP transmitter notifies RMP group members to withhold a message from the user until all users send confirmation, through tagged data in the reverse connections, that they have the message. Then the transmitter issues a 'commit' directive. Failure recovery techniques must ensure that if any group member receives the 'commit' and delivers the message that all operational members will deliver the message as well.

Once a group is created, a new member joins (or leaves) the group by contacting the current transmitter in the group. The transmitter uses a two-phase commit mechanism to atomically change the ordered list of group members kept by each member. Notification of the

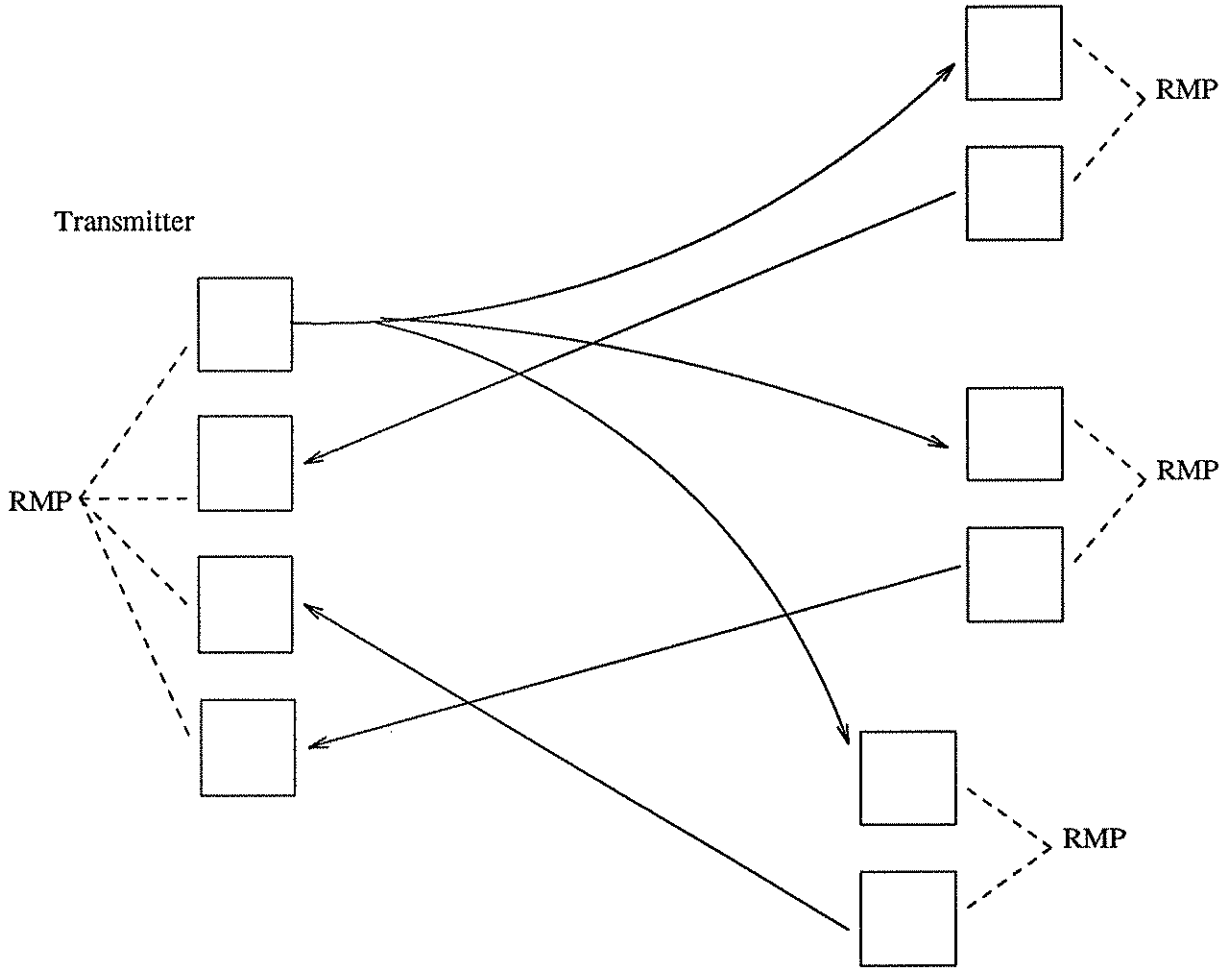

Figure 2 — Reliable Multicast with Reverse Connections

change will be passed up to the application, which will act on the information in an application-dependent fashion. By coordinating membership changes on a message boundary, the changes can be sequenced in the message stream being delivered to each local user (Figure 3). The transmitter brings the joining member into (or drops the leaving member out of) the conversation at the correct message boundary. Like user-generated messages, membership change notifications will be delivered to either none or all operational members in the same

order. If no member of the group is currently transmitting, the new (or leaving) member can itself contact the group and coordinate its joining (or leaving).

Members leaving the group due to process or host failures will be detected by the current (or next) transmitter to the group when the transfer fails due to the reception of less than the proper number of control messages. This action triggers failure recovery mechanisms to rebuild the ordered list of members, clean up any state inconsistencies (e.g. messages delivered to only a subset of the operational process group members), and recalculate delivery parameters affected by the group membership change (e.g., rate control and timer defaults). These recovery actions can be handled by a token-passing mechanism, similar to that in [4], using reliable unicasts. If no secondary failures occur, the token will need to circulate twice around the virtual

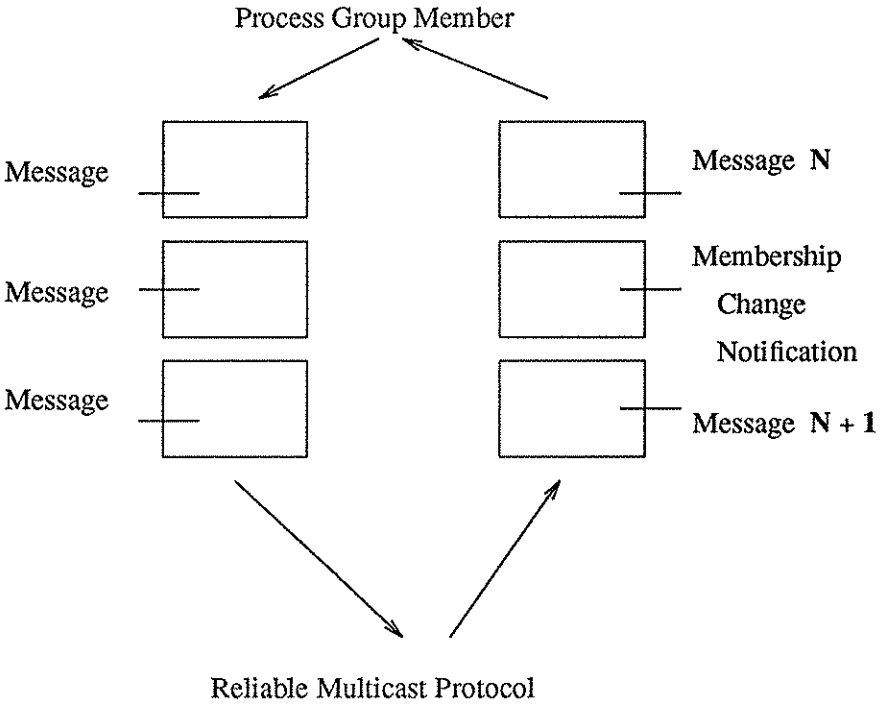

Figure 3 — User Notification on Group Membership Change

ring of group members in order to create and commit the new list at each member site.

References

[1] Chang, J. and Maxemchuk, N. F., "Reliable Broadcast Protocols", *ACM Transactions on Computer Science*, Vol. 2, No. 3, pp. 251-273, (August 1984).

[2] Kaashoek, M. F., Tanenbaum, A. S., Hummel, S. F. and Bal, H. E., "An Efficient Reliable Broadcast Protocol ", *Operating Systems Review*, Vol. 23, No. 4, (October 1989).

[3] Tseung, L. C. N., "Guaranteed, Reliable, Secure Broadcast Networks", *IEEE Network*, pp. 33-37, (November 1987).

[4] Rajagopalan, B. and McKinley, P., "A Token-Based Protocol for Reliable, Ordered Multicast Communication", *Proceedings of Eighth Symposium on Reliable Distributed Systems* , Seattle, Washington , (October 1989).

[5] Birman, K. and Joseph, T., "Reliable Communication in the Presence of Failures", *ACM Transactions on Computer Systems*, Vol. 5, No. 1, pp. 47-76, (February 1987).

[6] Joseph, T. and Birman, K., "Reliable Broadcast Protocols", in **Distributed Systems**, Mullender, S., (ed.), ACM Press, 1989.