THE INTERGROUP PROTOCOLS: SCALABLE GROUP COMMUNICATION FOR THE INTERNET

K. Berket, L. E. Moser and P. M. Melliar-Smith *
Department of Electrical and Computer Engineering
University of California, Santa Barbara, CA 93106
Email: karlo@alpha.ece.ucsb.edu, {moser@ece.ucsb.edu, pmms@ece.ucsb.edu}

Abstract

Existing reliable totally ordered group communication protocols have been developed for local-area networks and do not, in general, scale well to large numbers of nodes and wide-area networks. The InterGroup suite of protocols is a scalable group communication system that supports a receiver-oriented selection of service. The protocols are intended for a wide-area network, with a large number of nodes, that has highly variable delays and a high message loss rate, such as the Internet. The levels of message delivery service range from unreliable unordered to group ordered with certified guarantees of message delivery.

1 Introduction

Reliable totally ordered delivery of multicast messages is a useful service that simplifies the programming of distributed applications. With such a service, an application can be guaranteed that all processes in the groups that comprise the application receive the same messages in the same order. This helps to maintain the consistency of replicated information and to coordinate the activities of the various processes.

Several years of work on multicast group communication protocols have produced a number of systems that provide these services, including Isis [7], Horus [18], Reliable Multicast Protocol (RMP) [19], Trans/Total [12], Transis [2] and Totem [1, 14]. All of these systems were designed for local-area networks (LANs) and do not, in general, scale well to large numbers of nodes and wide-area networks, such as the Internet.

With the increasing popularity of the Internet, there is an increasing interest in protocols that are scalable. The MBone [8] provides a scalable best-effort multicast service for the Internet, and is a valuable infrastructure for group communication systems. It might appear that the same group communication protocols that work well over LANs could be run over the MBone. However, the reliable multicast, ordering and membership algorithms of existing LAN-based protocols are not scalable, and the MBone and IP Multicast do not provide basic group communication services. IP Multicast is a UDP-based best-effort service which means that, when a message is sent, it has some probability of being delivered to all of the group members. There is no service provided that determines who received the message or even who the group members are.

Reliable multicast protocols can be classified [11, 16] as sender-initiated, receiver-initiated with NACK-avoidance and tree-based protocols. Sender-initiated protocols place the responsibility for retransmission on the sender alone. For multicasting, sender-initiated protocols suffer from the ACK implosion problem; thus, focus has shifted to the other classes of protocols.

Receiver-initiated protocols with NACK-avoidance (RINA) have been shown [16] to improve performance over the basic receiver-initiated protocols, which suffer from the NACK implosion problem. SRM [9] and LBRM [10] are two examples of RINA protocols. Because they do not use ACKs, these protocols still suffer from the need for the application (SRM) or a hierarchy of log servers (LBRM) to keep the entire global state.

Tree-based protocols, such as AT&T's RMTP [15], are characterized by dividing the set of receivers into groups and distributing retransmission responsibility over an acknowledgment tree (ACK tree). It has been shown [11] that tree-based protocols do not suffer from the problems encountered in the other classes of protocols, but currently they require manual setup of the retransmission tree.

The goals of the InterGroup protocols described in this paper are twofold: (1) to extend the flexibility of the receiver-driven approach to the logical conclusion where each receiver picks its own level of service, and (2) to support a scalable form of group ordering and virtual synchrony [7, 13]. The overall goal is a protocol suite that is appropriate for the environment of the Internet (long latencies, high message loss rates, relatively frequent network partitioning) with scalability to both large numbers of nodes and large distances.

*This work was supported by the Director, Office of Energy Research, Office of Computation and Technology Research, Mathematical, Information, and Computational Sciences (MICS) Division, of the U.S. Department of Energy under Contract No. DE-AC03-76SF00098.

The InterGroup protocols employ a combination of sender-initiated and receiver-initiated reliable multicasts. To limit the scalability problems imposed by sender-reliable multicasts, InterGroup uses a self-organizing hierarchy of nodes for ACKing. Receiver-driven reliable multicasts are implemented on top of this tree structure.

2 Group Membership

The main hindrances to scalability of existing group communication systems are the message delivery guarantees provided to the application and their effect on the maintenance of the membership. Existing group communication systems require a strict view of the membership, in which every process in the group must be accounted for explicitly. This results in membership repair algorithms that are expensive in terms of time and messages required. There appears to be no way around those membership repair algorithms, whose message cost is $O(n^2)$, where n is the number of processes in the group. Moreover, the duration of the interval between membership changes is inversely proportional to n; thus, if the value of n is large, too much time may be spent in the membership algorithm itself. To ensure virtual synchrony, some group communication systems stop delivering messages while membership changes are taking place and, thus, for large values of n, they may deliver no messages at all. However, to obtain a consistent view of the membership and to ensure the message delivery guarantees, those expensive membership repair algorithms must be run.

The question thus remains: ''How can we make a group communication protocol more scalable?'' The InterGroup protocols approach this problem from various directions, all with the goal of reducing the effects that a membership repair algorithm has on the delivery of messages. The solution includes changing the delivery guarantees in a manner that reduces the strictness of group membership, allowing voluntary group joins and leaves that avoid the expensive membership repair algorithm, and adding a receiver-oriented selection of service that permits heterogeneity of the receiver set. These novel mechanisms and their effects on the group communication service are discussed below.

2.1 Basic Terminology

Before explaining the InterGroup approach to group communication, we define the following terms:

- A *node* is a single processor or a set of processors represented by a unique identifier within the context of an InterGroup session and the underlying network.
- A *process* is a logical entity represented by the identifier of the node to which it belongs and a unique identifier for that process on the node.
- A *multicast group* is a set of nodes that can communicate over the network and that are executing the InterGroup protocols.
- A *process group* is a set of processes that belong to nodes that are members of the multicast group.
- A *sender group* is a subset of a process group containing all of the processes in the group that have been multicasting messages at a rate r, that is larger than a given threshold *thresh*, during the immediately previous time interval of length t.
- A *receiver group* is a subset of a process group containing all of the processes that do not belong to the sender group of that process group.
- A *strict membership view of a group G by a process P* requires that P has explicit knowledge of the membership of G.
- A *virtual synchrony membership view of a group G by a process P* requires that P has a strict membership view of G, and that the view is updated according to the rules of a virtual synchrony service [7, 13].
- A *weak membership view of a group G by a process P* provides implicit knowledge of the membership of G, and is similar to the view offered to P by IP Multicast group membership. It does not guarantee knowledge of the exact membership of G by P but, if the information held by all of the reachable/non-failed processes in G were pooled together and reported to P, the explicit membership of G would be known by P. That is, P is capable of extrapolating the strict membership view of G through communication with some subset of the members of G.

Every process P in a process group G has a virtual synchrony view of the sender group of G, a weak membership view of the receiver group of G and, thus, a weak membership view of G as a whole.

2.2 Voluntary Joins and Leaves

Voluntary join and leave mechanisms are provided in InterGroup to facilitate changes to the group membership and to increase the scalability of the protocol. Most group communication protocols handle voluntary joins and leaves in the same way as merges and failures, respectively. The algorithms required for merging and for fault detection and repair increase in cost, as the size of the group grows. Because the algorithms for voluntary joins and leaves are simpler and faster on average, the InterGroup protocols make a distinction between such changes and those caused by outside forces (faults).

3 Message Delivery Guarantees

Message delivery guarantees allow easier development of a distributed application because they define the behavior that the application can expect of the underlying protocol. They must be stated in a clear and precise manner, and must be useful to the application programmer; otherwise, the application programmer will have to spend development time worrying about the underlying network, reliability, etc.

As an example of message delivery guarantees in existing group communication systems, we now consider those defined by the Totem system [1, 14]. Totem provides two message delivery guarantees: *agreed delivery* and *safe delivery*. Agreed delivery guarantees that the message is in a total order with respect to messages from the current processes that comprise the application. Safe delivery guarantees that the message has been received by all of the relevant current members. Safe delivery provides more knowledge to the application processes than agreed delivery, and safe messages have a longer latency to delivery than agreed messages.

In Totem, the sender of a message chooses, on a per message basis, whether the message is to be delivered agreed or safe. The receivers are not involved in that selection, and a safe message delays the delivery of agreed messages that follow it in the total order. Because of these two effects, the delay occurs at all of the current processes that comprise the application. Moreover, the application programmer may be mislead to think that safe delivery includes the knowledge that the message will be processed by all of the current members of the group. Such a guarantee would require common knowledge, which is impossible to achieve in an asynchronous distributed system. All that safe delivery guarantees is that each of the current members will process the message, if it does not fail before doing so.

The above understanding has led to a definition of message delivery guarantees for the InterGroup suite of protocols that are different from those of other group communication systems. Although the delivery guarantees are different, they are hopefully useful and scalable and, thus, appropriate for applications that run over the Internet.

3.1 Timestamped Delivery

In the InterGroup system, a message is said to be *timestamp delivered* when it is delivered in timestamp order with respect to messages from the current membership of the sender group. We refer to the membership of the sender group at the timestamp of the message being delivered as the *timestamped delivery membership*. This membership is used by the protocol to achieve reliable totally ordered message delivery and to ensure virtual synchrony at the protocol level; it is not delivered to the application. At the protocol level, both senders and receivers are aware

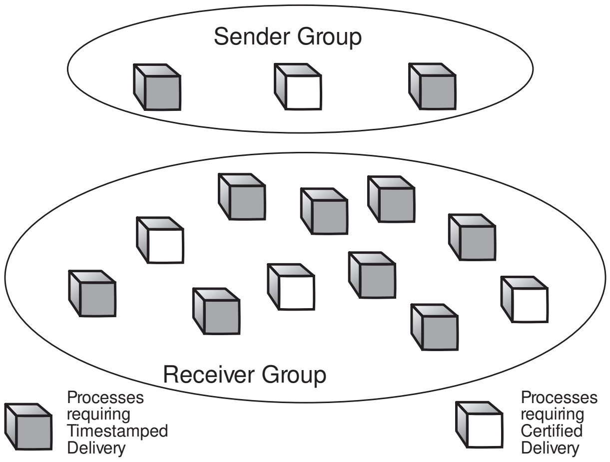

Figure 1: Within the self-organizing hierarchy of nodes, different types of message delivery and group membership services for the InterGroup system.

of the membership of the sender group, and have no exact knowledge of the current membership of the entire group.

Timestamped delivery of messages provides a total order on messages within a process group, which we call *group ordered delivery*. To satisfy the timestamped delivery guarantee, a process does not need to know the current membership of the receiver group (*i.e.*, it is a weak membership), because messages are born ordered (*i.e.*, their delivery order is determined when they are originated). Global synchronization is not required when the membership of the receiver group changes, because the receivers do not generate messages directly for the group and, thus, they do not contribute to the group order. Consequently, the guarantees of timestamped delivery in InterGroup are the same as those of agreed delivery in Totem (provided that process group and system are interchanged).

3.2 Certified Delivery

In InterGroup, a *certified membership group* is a subset of a process group containing all of the non-failed/non-terminated processes that joined the process group with a request for certified delivery. A strict membership model is imposed on this group. The certified membership group may include both senders and receivers; however, not all senders and receivers need to be members of this group. Figure 1 shows the relationship between the certified membership group and the other groups defined in InterGroup.

A certified message, when it is delivered, provides a guarantee that the message has been received at the protocol level by all processes that are members of the current certified membership group (*i.e.*, the certified membership group with the highest timestamp less than the timestamp of the message being delivered). Thus, certified delivery is similar to safe delivery in Totem with the following differences. First, certified delivery is not on a per message basis determined by the sender. The selection of this service is purely receiver-oriented and is valid for all messages delivered to the application by the process during its lifetime in the certified membership group. Second, certified delivery affects the message delivery latency only for members of the certified membership group. All other processes in the process group deliver messages according to their own chosen message delivery guarantees; they are not affected by the stronger guarantees required by the processes in the certified membership group. Certified delivery is, therefore, more powerful than safe delivery. The application programmer is given a choice of guarantees, and that choice does not affect any process with weaker delivery guarantees.

Timestamped delivery and certified delivery are the strongest message delivery guarantees provided by the InterGroup protocols. The weaker ones are not mentioned here because they do not fall into the traditional guarantees provided by group communication protocols. They are discussed in [5].

4 The Logical Hierarchies of InterGroup

The InterGroup system employs two orthogonal logical hierarchies to achieve scalability.

The first is a self-organizing two-level hierarchy of nodes, adapted from that of SRM [17], whose structure depends on the nodes in the physical network. In Section 4.1 we describe the logical structure of this hierarchy, and how it affects buffer management and reliable message delivery.

The second is a dynamic two-set hierarchy whose structure depends on the activity of the processes. In Section 4.2 we describe the logical structure of this hierarchy, and the fault-tolerance mechanisms associated with it.

To simplify the explanations, we assume that all of the processes request timestamped delivery on startup. A more general description involving the other cases and heterogeneity of the message delivery guarantees is presented in [5].

4.1 The Node Hierarchy

The logical structure of this hierarchy considers the location of, and the latencies between, nodes in order to reduce the global control traffic of the protocols and allow scalable buffer management and reliable message delivery. The nodes are organized in a multicast group as a tree, with the root nodes called the *coordinators*, and the leaf nodes called the *children*, as shown in Figure 2. Each child node is associated with a coordinator. The *local group* of a coordinator is composed of the child nodes associated with that coordinator (including the coordinator itself). The *coordinator group* of the hierarchy consists of all coordinators in the hierarchy. The coordinator group membership supports a form of virtual synchrony in order to perform buffer management correctly (for more details see [6]). Every node limits its communication of control information (distances, loss rates, etc) to its local group, except for the coordinators, which also communicate with the other members of the coordinator group.

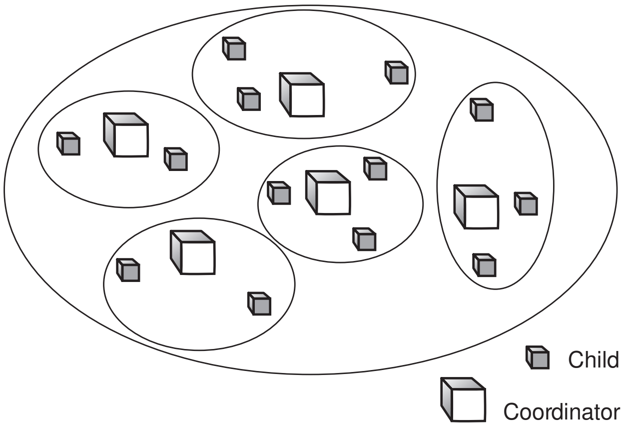

Figure 2: The node hierarchy.

4.1.1 Buffer Management

We now present the buffer management strategy for InterGroup, limited to a single process group for simplicity of presentation.

A child node in a local group forms a set containing the highest timestamps of messages delivered to each process associated with it. The node creates an ACK message containing the lowest timestamp in this set and multicasts that message to the members of its local group.

A coordinator receives ACKs from all nodes in its local group. It forms a set containing the highest timestamps of all messages that it has delivered to every process associated with it, and also the highest timestamps contained in ACK messages that it has received from all nodes in its local group. The coordinator creates an ACK message that contains the lowest timestamp in this set and multicasts that message to the coordinator group.

A coordinator also receives ACKs from all nodes in the coordinator group. It forms a set containing the highest timestamps contained in ACK messages that it has received from all nodes in the coordinator group (including itself). The lowest timestamp in this set represents a message that every process in the process group has received. It also signifies that every process in the process group has received all messages with lower timestamps. The coordinator then multicasts this timestamp to its local group, thus allowing every process in its local group to remove those messages from its message buffers.

Further details about the timestamp acknowledgment mechanisms can be found in [4].

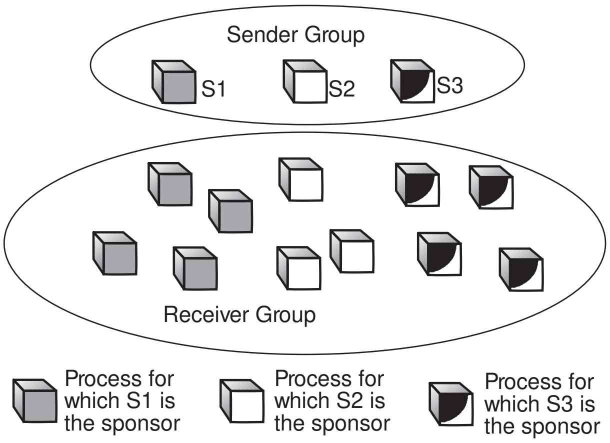

Figure 3: The process group hierarchy.

4.1.2 Reliable Message Delivery

The InterGroup protocols use an SRM-based approach to negative acknowledgments, adapting the algorithms in [9]. Due to space constraints, we present here only an outline of the reliable delivery protocols. More details can be found in [3].

Upon recognition of a missing message, a process R sets a randomized request timer based on its estimate of the one-way distance from the node associated with the sender process S of the missing message, and the node associated with R.* If R receives a retransmission request for the same message before the request timer expires, R performs a random exponential backoff, and resets the request timer to the new value. When the request timer expires, R multicasts a retransmission request to the process group.

When a process P receives a retransmission request from R for a message that P is capable of retransmitting, P sets a randomized repair timer based on an estimate of the one-way distance between the node associated with R and the node associated with P. If P receives a retransmission of the missing message before the repair timer expires, then P cancels the repair timer. When the repair timer expires, P multicasts the retransmission of the missing message.

Overall, the mechanisms used to achieve reliable message delivery using the self-organizing hierarchy of nodes eliminate ACK and NACK implosion (so long as the number of children of each node is limited), independent of the number of group members.

4.2 The Process Group Hierarchy

At any time each process can be a member of one or more process groups. In each process group, a process can be a member of either the sender group or the receiver group, as shown in Figure 3. The membership of the sender group is explicitly known to each process that is a member of the process group. The membership view of the receiver group is a weak membership view and, thus, is not explicitly known. Nevertheless, fault detection and membership changes in the process group as a whole must be handled correctly.

To become a member of a process group, a process subscribes to a class D Internet address reserved for the process group. It then proceeds to become a member of the receiver group by finding a member of the sender group, to act as its *sponsor*, as shown in Figure 3. If a process decides to join the sender group, it sends a Request message to its sponsor, and is admitted to the sender group at the timestamp of the Sponsor message multicast by its sponsor. When a member of the sender group decides to retreat to becoming a receiver, it sends a Leave message indicating that it is leaving the sender group, and is removed from the sender group at the timestamp of the Leave message. During its lifetime in the process group, a process may dynamically switch between the sender group and the receiver group many times.

The process group hierarchy is used to improve the scalability of the process group membership algorithm. To achieve virtual synchrony, every member of the process group (senders and receivers alike) must deliver a membership change at the same logical time with respect to the ordering of messages. In particular, two processes that proceed from one membership view to the same next membership view must deliver the same messages in the first membership view.

The process group hierarchy allows the sender group membership to be determined by the members of the sender group alone, while also determining the logical time at which this change occurs relative to the ordering of messages. It allows the membership algorithm to run without needing the details of the hierarchy of nodes and reliable message delivery, and without halting the ordering of messages during the membership change.

The fault detection mechanisms, outlined below, have been designed to support scalability and to make use of the membership and hierarchy mechanisms described previously.

The receiver group membership view is a weak membership view and, thus, faults of processes in the receiver group do not need to be reported to the process group because they do not affect the progress of that group. Thus, fault detection and removal of faulty members of the receiver group are unnecessary.

In contrast, the sender group membership view is a virtual synchrony membership view, and invocation of the membership repair algorithm cannot be avoided when faults of processes in the sender group occur. Each member of the sender group is responsible for detecting faults of all other members of the sender group. This

*The estimate of the one-way distance to a node is obtained from control information exchanged through the hierarchy of nodes.

is done using timers and Keep Alive messages that each sender transmits when it has no data messages to send.

The membership repair algorithm and sender fault detection are described in more detail in [6].

5 Conclusion

The motivation for the InterGroup protocols is the lack of a group communication system that is scalable and runs effectively over the Internet. In this paper we have given an overview of the InterGroup suite of group communication protocols. We have identified problems with existing group communication protocols in dealing with such an environment, and have described the InterGroup approach to solving those problems. We are currently implementing the InterGroup protocols in Java, and will begin real-world testing soon.

References

[1] D. A. Agarwal. *Totem: A Reliable Ordered Delivery Protocol for Interconnected Local-Area Networks*. PhD thesis, University of California, Santa Barbara, August 1994.

[2] Y. Amir, D. Dolev, S. Kramer, and D. Malki. Transis: A communication subsystem for high availability. In *Proceedings of the 22nd IEEE International Symposium on Fault-Tolerant Computing*, pages 76--84, New York, NY, July 1992.

[3] K. Berket, L. E. Moser, and P. M. Melliar-Smith. The InterGroup protocols: Scalable group communication for the Internet. Technical Report 97-08, University of California, Santa Barbara, December 1997.

[4] K. Berket, R. Koch, L. E. Moser, and P. M. Melliar-Smith. Timestamp acknowledgments for determining message stability. Technical Report 98-12, University of California, Santa Barbara, April 1998.

[5] K. Berket, L. E. Moser, and P. M. Melliar-Smith. Receiver-oriented selection of message delivery services. Technical Report 98-22, University of California, Santa Barbara, August 1998.

[6] K. Berket, L. E. Moser, and P. M. Melliar-Smith. The InterGroup membership protocols. Technical Report 98-23, University of California, Santa Barbara, August 1998.

[7] K. P. Birman and R. Van Renesse. *Reliable Distributed Computing with the Isis Toolkit*. IEEE Computer Society Press, 1994.

[8] H. Eriksson. MBone: The multicast backbone. *Communications of the ACM*, 37:54--60, August 1994.

[9] S. Floyd, V. Jacobson, C.-G. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *IEEE/ACM Transactions on Networking*, 5(6):784--803, December 1997.

[10] H. W. Holbrook, S. K. Singhal, and D. R. Cheriton. Log-based receiver-reliable multicast for distributed interactive simulation. *Computer Communication Review*, 21(4):328--341, October 1995.

[11] B. N. Levine and J. J. Garcia-Luna-Aceves. A comparison of known classes of reliable multicast protocols. In *Proceedings of the IEEE International Conference on Network Protocols*, pages 112--121, Columbus, OH, October 1996.

[12] P. M. Melliar-Smith, L. E. Moser, and V. Agrawala. Broadcast protocols for distributed systems. *IEEE Transactions on Parallel and Distributed Systems*, 1(1):17--25, January 1990.

[13] L. E. Moser, Y. Amir, P. M. Melliar-Smith, and D. A. Agarwal. Extended virtual synchrony. In *Proceedings of the 14th IEEE International Conference on Distributed Computing Systems*, pages 56--65, Poznan, Poland, June 1994.

[14] L. E. Moser, P. M. Melliar-Smith, R. K. Budhia D. A. Agarwal, and C. A. Lingley-Papadopoulos. Totem: A fault-tolerant multicast group communication system. *Communications of the ACM*, 39(4):54--63, April 1996.

[15] S. Paul, K. K. Sabnani, J. C. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (RMTP). *IEEE Journal on Selected Areas in Communications*, 15(3):407--421, April 1997.

[16] S. Pingali, D. Towsley, and J. F. Kurose. A comparison of sender-initiated and receiver-initiated reliable multicast protocols. *Performance Evaluation Review*, 22:221--230, May 1994.

[17] P. Sharma, D. Estrin, S. Floyd, and L. Zhang. Scalable session messages in SRM using self-configuration. Technical Report 98-670, USC, February 1998.

[18] R. van Renesse, K. P. Birman, and S. Maffeis. Horus: A flexible group communication system. *Communications of the ACM*, 39(4):76--83, April 1996.

[19] B. Whetten, T. Montgomery, and S. Kaplan. A high performance totally ordered protocol. In *Proceedings of the International Workshop on Theory and Practice in Distributed Systems*, pages 33--57, Dagstuhl Castle, Germany, September 1994. Springer-Verlag.