Semantically Reliable Multicast: Current status and future work

BRIEF ANNOUNCEMENT

José PEREIRA*
Universidade do Minho
jop@di.uminho.pt

Luís RODRIGUES
Universidade de Lisboa
ler@di.fc.ul.pt

Rui OLIVEIRA
Universidade do Minho
rco@di.uminho.pt

Abstract

In multicast communication systems, a single perturbed recipient can drastically affect the performance of a complete group of processes. One way to alleviate this problem is to weaken reliability requirements by allowing some messages to be omitted. We propose a multicast service that exploits semantic knowledge to select which messages can be omitted without compromising the application's correctness. This service is based on the concept of *message obsolescence*: A message becomes obsolete when its content is overwritten or implicitly conveyed by a subsequent message.

Besides summarizing initial research results [10] showing that message obsolescence can be expressed in a generic way and can be used to achieve a higher stable throughput, this text advances a definition of the service and outlines our current research directions.

**Departamento de Informática*
Universidade do Minho
Campus de Gualtar
4710-057 BRAGA PORTUGAL

Semantically Reliable Multicast: Current status and future work

José Pereira
Universidade do Minho
jop@di.uminho.pt

Luís Rodrigues
Universidade de Lisboa
ler@di.fc.ul.pt

Rui Oliveira
Universidade do Minho
rco@di.uminho.pt

Abstract

In multicast communication systems, a single perturbed recipient can drastically affect the performance of a complete group of processes. One way to alleviate this problem is to weaken reliability requirements by allowing some messages to be omitted. We propose a multicast service that exploits semantic knowledge to select which messages can be omitted without compromising the application's correctness. This service is based on the concept of *message obsolescence*: A message becomes obsolete when its content is overwritten or implicitly conveyed by a subsequent message.

Besides summarizing initial research results [10] showing that message obsolescence can be expressed in a generic way and can be used to achieve a higher stable throughput, this text advances a definition of the service and outlines our current research directions.

1 Motivation

Reliable multicast [7] ensures that the same messages are delivered to all processes in a group. When messages are produced by the source faster than the time it takes for some target to consume them, the surplus needs to be temporarily buffered. However, buffering is effective only for short bursts of traffic. If message multicast rate is consistently high, unbounded buffer space would be required and message delivery by the slowest process would increasingly lag behind. Eventually, the message source must be slowed down or, alternatively, the slow recipient needs to be excluded from the multicast group.

In contrast to what happens in point-to-point protocols, a multicast communication channel is shared among the receivers. As such, besides reducing the rate to the slow process as desired, flow control impacts all receivers. This is awkward since, even if no system component between a fast sender and a fast receiver is congested, the maximum throughput between nodes may not be achieved.

Such performance degradation resulting from a single slow process and flow control[1], often referred as the "crying baby syndrome", is a significant threat to the deployment of reliable multicast protocols for applications that require sustained high message transmission rates [2].

2 Related work

A path to address the problem is to weaken reliability requirements, so that slower receivers are not required to deliver all messages and thus

[1] Notice that this is a consequence of reliability itself and should not be confused with performance problems resulting from implementation mechanisms for ensuring reliability, such as *ack implosion*. For a detailed treatment of these see [6].

do not slow down the sender. For instance, Δ-causal [1] and deadline constrained causal [13] protocols may omit the delivery of late messages in order to unblock the delivery of subsequent messages.

It has also been proposed that no automatic retransmission of lost messages is done [4]. Instead, the receiver application should be notified and given the possibility to explicitly request retransmission of lost messages when considered relevant.

A different approach is Bimodal Multicast [3], which offers probabilistic reliability guarantees: The probability of a message being delivered to some but not all processes can be made as small as necessary by adjusting protocol parameters. In addition, if probabilistic guarantees are not considered sufficient, notification of message losses enables receivers to take any corrective action deemed necessary. Nonetheless, even if some mechanism is implemented to notify receivers when messages are dropped, the application might be unable to take any corrective action since it has no knowledge of that message's content, and thus, whether it is important or not.

Further research has proposed the parallel use of two multicast protocols: An unreliable protocol used for payload and a reliable protocol used to convey meta-data describing the content of data messages sent on the payload channel [12]. Using this information the receiver may evaluate the relevance of lost messages. Our approach is inspired on this principle, but exploits the semantic knowledge at the sender side instead.

3 Semantic reliability

The basic idea behind our approach is that in a distributed application some messages either overwrite or implicitly convey the content of other messages sent in the past, therefore making them irrelevant. If obsolete messages have not been yet delivered, they can be safely purged without compromising the application's correctness. If a slow receiver exists but enough messages can be purged, the protocol will not need to slow down the sender, thus ensuring that the performance of fast processes remains unaffected.

Our approach thus aims at *semantic reliability*, as all current information is delivered to all receivers, either implicitly or explicitly, without necessarily delivering all messages.

For instance, applications embodying operations with overwrite semantics, in particular, applications managing read-write items are the most obvious example of applications that exhibit message obsolescence. In these applications, any update of a given item is made obsolete by subsequent write operations.

Many distributed algorithms are structured in logical rounds. When the algorithm advances, messages from previous rounds become obsolete. Recognizing this property, it has been shown [8] how distributed consensus can be solved in asynchronous distributed systems augmented with failure detectors using unreliable channels and bounded message buffers.

Notice that, to be effective, the obsolescence property cannot be exploited solely at the application layer, since liveness or timing constraints force the application to immediately forward outgoing messages to the communication channel. Being so, messages become out of reach and cannot be discarded even if immediately made obsolete.

4 Service definition

We consider an asynchronous message passing system as defined in [7]. Briefly, the system is composed of a set P of n sequential processes. Processes can fail by crashing and communicate by message passing through a fully connected network. Processes that do not crash are correct.

Semantically Reliable Multicast is defined in terms of two primitives: SR-MULTICAST(m) and SR-DELIVER(m), where m is a message from a set M of all possible messages. When a process $p \in P$ executes SR-MULTICAST(m) we say it multicasts m and when it executes SR-DELIVER(m) we say it delivers m.

The required semantical information is formalized as a relation on messages. For each pair of related messages $m \sqsubseteq m'$, we say that m is *obsoleted* by m'. This relation is defined by each application and we assume that it is a partial order and is coherent with causal order of events.

The intuitive meaning of this relation is that if $m \sqsubseteq m'$ and if m' is delivered, the correctness of the application is not affected by omitting the delivery of m. Semantically Reliable Multicast is thus defined as satisfying the following properties:

Validity: If a correct process multicast a message m, all correct processes eventually deliver m' such that $m \sqsubseteq m'$.

Agreement: If a correct process delivers a message m, then all correct processes eventually deliver m', such that $m \sqsubseteq m'$.

Integrity: For any message m, every correct process delivers m at most once, and only if some process previously multicast m.

It can be trivially shown that if no $m, m' \in M$ exist such that $m \sqsubseteq m'$, Semantically Reliable Multicast is reduced to conventional Reliable Multicast [7]. If for all $m, m' \in M$ such that m is multicast by the same process as and before m' then $m \sqsubseteq m'$, it results in the extension to multicast of the 1-stubborn channel [5].

5 Initial results

Our initial research [10] has focused on showing that between extreme configurations, there are concrete applications exhibiting obsolescence patterns that result in meaningful purging rates and that semantic reliability produces a significant performance advantage. Specifically, the amount of purging observed determines how different receiving rates within the same multicast group can be accommodated.

As a case study we focused on the publishing system that is used to disseminate information about operations and quotes in an on-line stock trading system. In this context, both the timeliness and the reliability of the updates are extremely important, in addition to sustaining a high throughput to a large number of processes. Unfortunately, when one of the recipients is congested, flow control can degrade the performance of the complete system [11].

Traffic generated in such application has a distinct profile: A small percentage of stocks is accountable for a large share of operations [9] leading to a high probability of messages about the same share being issued close to each other. As a consequence, our protocol can be configured to tolerate receivers which are up to 40% slower than those required to process all messages in due time. These results were achieved using a analytical model and validated using simulation. This model enables reasoning about the efficiency of the protocol and the configuration of system parameters according to the obsolescence pattern of the target application, thus being a valuable tool for both protocol and application developers.

6 Conclusions and research directions

Our work has illustrated the advantages of using the notion of message obsolescence in the design of protocols for high throughput applications. The resulting protocol selectively purges messages that are consuming system resources without compromising the application's correctness, enabling processes with different receiving rates to coexist within the same multicast group. We draw the conclusion that semantic reliability is a viable approach to ensure a higher stable multicast throughput in the presence of perturbed group members.

In contrast to solutions that admit message loss and offload responsibility of corrective action to applications, our proposal also has the advantage of providing a self contained solution which can be researched and developed independently from applications. For the application developer, semantic reliability provides strong correctness guarantees trough a simple programming interface, as is expected in reliable process groups.

We are currently extending this work in several directions. First, we are studying how the notion of message obsolescence interacts with other aspects of reliable communication, such as ordering constraints and membership, in order to present an integrated group communication suite for high throughput applications.

In addition, we are researching how the obsolescence relation can be conveyed from an application to a protocol, such that their specification and implementation can be kept separate, the overhead minimized and that the interface is safe, i.e. no invalid obsolescence relations can be specified.

References

[1] R. Baldoni, R. Prakash, M. Raynal, and M. Singhal. Efficient Δ-causal broadcasting. *International Journal of Computer Systems Science and Engineering*, 13(5):263–269, September 1998.

[2] K. Birman. A review of experiences with reliable multicast. *Software Practice and Experience*, 29(9):741–774, July 1999.

[3] K. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems*, 17(2):41–88, 1999.

[4] D. Clark and D. Tennenhouse. Architectural considerations for a new generation of protocols. In *SIGCOMM Symposium on Communications Architectures and Protocols*, pages 200–208, Philadelphia, PA, September 1990. ACM.

[5] R. Guerraoui, R. Oliveira, and A. Schiper. Stubborn communication channels. Technical Report 98-278, Département d'Informatique, École Polytechnique Fédérale de Lausanne, 1998.

[6] K. Guo. *Scalable Message Stability Detection Protocols*. PhD thesis, Cornell University, Computer Science, May 1998.

[7] V. Hadzilacos and S. Toueg. Fault-tolerant broadcasts and related problems. In Sape Mullender, editor, *Distributed Systems*, chapter 5, pages 97–145. Addison Wesley, 1993.

[8] R. Oliveira. *Solving consensus: From fair-lossy channels to crash-recovery of processes*. PhD thesis, École Polytechnique Fédérale de Lausanne, February 2000.

[9] P. Peinl, A. Reuter, and H. Sammer. High contention in a stock trading database: A case study. *ACM SIGMOD Record*, 17(3):260–268, September 1988.

[10] J. Pereira, L. Rodrigues, and R. Oliveira. Semantically reliable multicast protocols. Submitted to the Nineteenth IEEE Symposium on Reliable Distributed Systems, October 2000.

[11] R. Piantoni and C. Stancescu. Implementing the Swiss Exchange Trading System. In *Proceedings of The Twenty-Seventh Annual International Symposium on Fault-Tolerant Computing (FTCS'97)*, pages 309–313. IEEE, June 1997.

[12] S. Raman and S. McCanne. Generalized data naming and scalable state announcements for reliable multicast. Technical Report CSD-97-951, University of California, Berkeley, June 1997.

[13] L. Rodrigues, R. Baldoni, E. Anceaume, and M. Raynal. Deadline-constrained causal order. In *The 3rd IEEE International Symposium on Object-oriented Real-time distributed Computing*, March 2000.