Channel Islands in a Reflective Ocean: Large-Scale Event Distribution in Heterogeneous Networks

Jon Crowcroft, Jean Bacon, Peter Pietzuch, George Coulouris, and Hani Naguib
University of Cambridge

ABSTRACT

We discuss the design of a multicast event distribution service intended to support extremely large-scale event distribution. To date, event notification services have been limited in their scope due to limitations of the infrastructure. At the same time, Internet network and transport layer multicast services have seen limited deployment due to lack of user demand (with the exception more recently of streaming services, e.g., on Sprint's U.S. core network and in the Internet II). Recent research in active networks and reflective middleware suggests a way to resolve these two problems at one go. The goal of this article is to describe a reflective middleware system that integrates the network, transport, and distributed middleware services into a seamless whole. The system integrates this "low-level" technology into an event middleware system, suitable for telemetry, novel mobile network services, and other as yet unforeseen applications.

INTRODUCTION

In this article we discuss requirements of and propose a design for a multicast service that can distribute event messages to subscribers throughout the Internet on a scale comparable to today's transport-level services. We can envisage a world in which pervasive computing devices generate 10,000,000,000 events per second. We can foresee a time when there are thousands of millions of event subscribers all over the planet, with publishers having popularities as low as no or only a single subscriber, or as high as the entire world.

Event-driven and messaging infrastructures are emerging as the most flexible and feasible solution for enabling rapid and dynamic integration of legacy and monolithic software applications into distributed systems. Event infrastructures also support deployment and evolution of traditionally difficult-to-build active systems such as large-scale collaborative environments and mobility-aware architectures [1].

Event notification is concerned with propagation of state changes in objects in the form of events. A crucial aspect of events is that they occur asynchronously. Event consumers have no control over when events are triggered. On the other hand, event suppliers do not generally know which entities might be interested in the events they provide. These two aspects clearly define event notification as a model of asynchronous and decoupled communication, where entities communicate in order to exchange information, but do not directly control each other.

The architecture of an event distribution overlay layer is illustrated in Fig. 1. In this we can see that a publisher creates a sequence of events that carry attributes with given values. A consumer subscribes to a publisher, and may express content-based filters to the publisher. In our system, these filter expressions can be distributed upstream from the consumer toward the publisher. As they pass through application-level event notification distributors, they can be evaluated and compared, and possibly combined with other subscription filters. Notifications of interest are passed upstream all the way to the publisher, or to the application-level event notification distributor nearest the publisher, which can then compute a set of fixed tags for data; it can also, by consulting with the IP and GRA routers, through the reflective multicast routing service, compute a set of IP multicast groups over which to distribute the data. This will create the most efficient trade-off between source and network load, and receiver load, as well as tag and filter evaluation, as the events are carried downstream from the publisher, over the IP multicast, GRA, and application-level event notification nodes.

Devising and evaluating the detailed performance of the algorithms to carry out these tasks form the core of the requirements for future work.

0163-6804/02/$17.00 © 2002 IEEE

BACKGROUND

The Internet Engineering Task Force (IETF) is just finishing specifying a family of reliable multicast transport protocols, for most of which there are pilot implementations. Key among these for the purposes of this research is the exposure to end systems of router filter functionality in a programmable way, known as *generic router assist* (GRA). This is an inherent part of the Pragmatic General Multicast service, implemented by Reuters, Tibco, and Cisco in their products, although it has not been widely known or used outside of the *TIBNET* products until very recently.

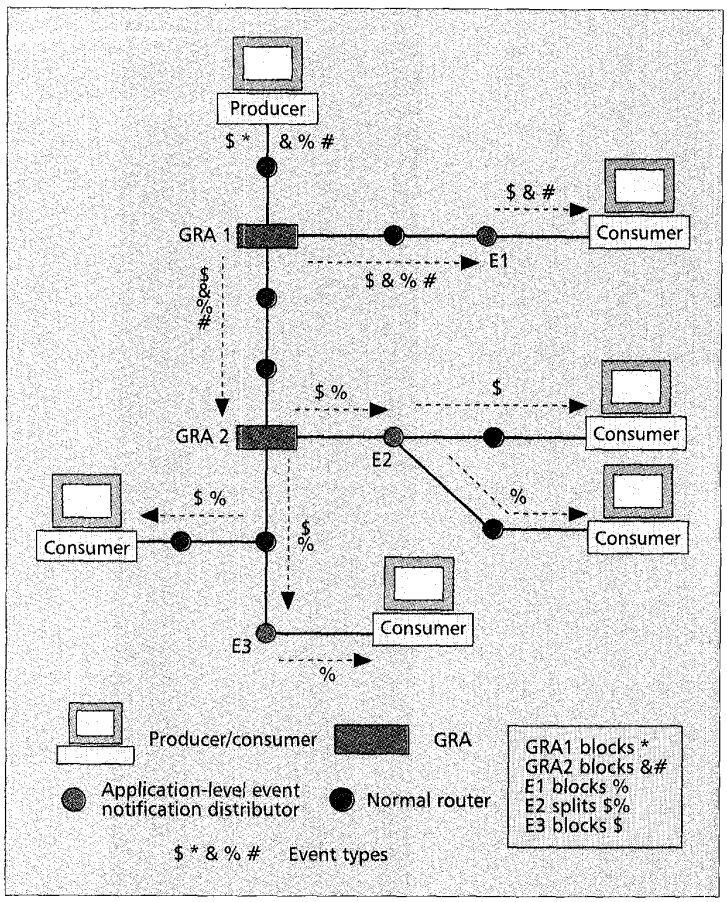
The last decade has seen great leaps in the maturity of distributed systems middleware, and in one particular area, event notification systems, in support of a wide variety of novel applications. Current work on event notification middleware [2, 3] has concentrated on providing the infrastructure necessary to enable content-based addressing of event notifications. These solutions promote a publish-subscribe-match model by which event sources publish the metadata of the events they generate, event consumers register for their events of interest passing event filter specifications, and the underlying event notification middleware undertakes the event filtering and routing process. Solutions differ usually on whether they undertake the filtering process at the source or at an intermediary mediator or channel. The trade-off lies in whether to increase the computational load of sources and decrease network bandwidth consumption, or minimize the extra computational load on the sources and outsource the event filtering and routing task to a mediator component (hopefully located close to the source). All of these solutions do not leverage the potential benefits that event multicasting to consumers, which requires the same type of events and applies very similar filters, could bring. They usually require an individual unicast communication per event transmitted.

At the same time, the underlying network has become very widespread. New services such as IP multicast are finally seeing widespread deployment, especially in core networks and intranets.

The combination of these two technologies, event services and multicast, originates historically with Tibco [4], a subsidiary of Reuters. However, their approach is somewhat limited since it takes a strictly layered approach.

At the highest level, there is a publish/subscribe system, which in *TIBNET* uses *subject-based addressing* and *content-based addressing*. Receivers subscribe to subjects. The subject is used to hash to a multicast group. Receivers subscribe to a subject but can express interest by declaring filters on content. The *TIBNET* system is then hybrid. In the wide area, IP multicast is used to distribute all content on a given subject topic to a set of site proxy servers. The site proxy servers then act on behalf of subscribers at a site, filter appropriate content out of each subject stream, and deliver the remains to each subscriber.

Between the notification and IP layers there is a transport layer protocol called Pragmatic General Multicast (PGM). To provide semi-reliable in-order delivery, the subject messages are mapped onto PGM [5] messages, which are then multicast in IP packets. PGM provides a novel retransmission facility that takes advantage of router level functionality for "nack aggregation," preventing message implosion toward the event source, and to provide filtering [6] of retransmissions so that only receivers missing a given message sequence number receive it. The PGM protocol is essentially a lightweight signaling protocol that allows receivers to install and remove filters on parts of the message stream. The mechanism is implemented in Cisco and other routers that run IP multicast. The end system part of the protocol is available in all common operating systems.



■ **Figure 1.** *Channel islands event distribution overlay architecture.*

Almost all other event notification systems have taken the view that IP multicast was rarely deployed,[1] and that the overheads in the group management protocols were too high for the rate of change of interest/subscription typical in many applications usage patterns [8].

Instead, they have typically taken an alternative approach of building a server level overlay for event message distribution. Recent years have seen many such overlay attempts [9]. These have met with varying degrees of success. One of the main problems of application layer service location and routing is that the placement of servers does not often match the true underlying topology of the physical network, and is therefore unable to gain accurate matching between a distribution tree and the actual link throughput or latencies. Nor is

[1] *Ironically, this view was fueled partly by a report by Sprint [7], when in fact the entire Sprint IP service supports multicast and they have at least 3500 commercial customers streaming content.*

There are two ideas we draw from in moving forward. First we exploit advances in the network support for multicast. Second, we distribute an open interface to the multicast tree computation that IP routers implement.

the system able to estimate accurately the actual available capacity or delay. Even massive-scale deployments such as Akamai do not do very well.

Second, the delays through application level systems are massively higher than those through routers and switches (which are after all designed for packet forwarding, rather than server or client computation or storage resource sharing). The message is that overlays and measurement are both hard to optimize, and inefficient. It is frequently the case that in the long term, business migrates into the infrastructure (c.f. voice, IP, etc.). We expect many overlay services to do this. We believe that this process will accelerate due to the use of state-of-the-art network middleware and software engineering approaches. However, this process will not stop: there is an endless stream of new services being introduced "at the top," and making their way down to the bottom, to emerge as part of the critical information infrastructure.

The Proposed Approach

We see a number of advantages in continuing forward from where Tibco left off in integrating efficient network delivery through multicast, with an event notification service including:

- **Scale**: We obviate the need to deploy special proxy servers to aid the distribution.
- **Throughput**: The system is therefore able to distribute many more events per second.
- **Latency**: Event distribution latency will approximate the packet level distribution delay, and avoid the problems of high latency and jitter incurred when forwarding through application level processes on intermediaries.

There are two ideas we draw from in moving forward. First we exploit advances in the network support for multicast, such as GRA service in the PGM router element in IP multicast. Second, we distribute an open interface to the multicast tree computation IP routers implement. The way we do this is through reflection.

Reflection is becoming commonplace in middleware [10], but has not been applied between application level systems and network level entities to our knowledge. The choice here is to offer a common application programming interface (API) to both the multicast and filtering services, so the event notification module implementer need not be aware which layer is implementing a function. We envisage an extremely simple API:

```
Create(Subject)
Subscribe/Join(Subject)
Publish/Send(Subject, Content)
Receive(Subject, Content Filter
Expression)
```

In most current event services, objects and filters are specified using a string hierarchy. An XML [11] or SOAP-based hierarchy would offer stronger typing. The router level creates both a real distribution tree for subjects and a subtree for each filter or merged filter set. This is done with regard to the location (and density) of receivers. We can use a multicast tunnel or multicast address translation service to provide further levels of aggregation within the network. This requires the routers to perform approximate tree matching algorithms.

We are building a piece of reflective middleware that will add a thin layer between an existing event notification service and the reflective routing and filter service. This involves extending the PGM *signaling* protocol that installs and activates the filters via IP router alerts. We are investigating efficient hashes for subject to group and content to sequence number mapping [12].

We intend to evaluate our approach by applying it to a large-scale event driven (sentient) application, such as novel context-aware applications for the emerging Universal Mobile Telecommunications System (UMTS) mobile telephony standard or large-scale location tracking applications. For example, there is the possibility of developing location tracking (people, vehicles, and baggage) for large new airport terminals. There has been a surge in interest in mobile event sinks, as we can see in recent published work [13–15].

Discussion

One of the goals of this work is to explore the way the multicast trees and filtering system evolve in large heterogeneous application environments. Another goal is to see how multicast routing can be "laid open" as a service to be used to build distribution trees for other layers. Finally, we believe that the three levels we have may not be enough, and that as the system grows larger still, other services may emerge.

What we have designed is effectively a two-tier system, building on previous work [12], which entails multicast trees and, within these, filters. To these, we have added a third overlay layer.

The purpose of the overlay is to accommodate a form of qualitative heterogeneity discussed below, whereas the lower two layers of multicast and filtering target the area of quantitative performance differences.

The overlay layer is required, first because current event distribution systems are built without any notion of a multicast filter-capable transport. Thus, we must have an overlay of event distribution servers. These can, where the lower services are available, be programmed to take advantage of them, *among themselves*, thus providing a seamless mechanism to deploy the new service transparently to publisher and subscriber systems.

Second, we believe there are inherent structural reasons why such an application layer overlay is needed. These include:

- **Policies**: Different regions of the network will have different policies about which events may be published and which not.
- **Security**: There may be firewall or other security mechanisms that impede the distribution via lower-level protocols.
- **Evolution**: We would like to accommodate local evolution of multicast routing mechanisms (in the same way interdomain routing protocols such as Border Gateway Protocol, BGP, allow intradomain routing to evolve).
- **Interworking**: We would like to support a variety of event distribution middleware systems. We have some initial results in this area [11, 16].
- **Others**: There are other such "impedance mismatches" we may encounter as the system scales up.

A novel aspect of our approach is that the overlay system does not itself construct a distri-

bution tree. Instead, a set of *virtual* members are added to the lower level distribution system, which then uses its normal multicast routing algorithms to construct a distribution tree among a set of event notification servers separated in islands of multicast-capable networks. These servers then use an open interface to query the routers as to the computed tree, and use this as their own distribution topology. In this way the overlay can take advantage of detailed metric information to which the router layer has access (e.g., delay, throughput, and current load on links) instead of measuring a poor shadow of that data which would lead to an inaccurate and out-of-date set of parameters with which to build the overlay. In some senses, what we are doing here is a form of multicast traffic engineering for the existing network.

We believe that our proposed system can provide a number of engineering and performance enhancements over previous event notification architectures. Future work will evaluate these, including:

- System performance — Improvement in scalability, including reduction in join/leave publish/subscribe latency, increase in event throughput, and so on.
- Network impact — Impact on router load of filter processing, group join, leave, and multicast packet forwarding.
- Expressiveness and seamlessness of API — We plan to try it with a variety of event notification systems and to export a portable implementation via public CVS to see what the open source community does with it.
- Mobility — The dynamic nature of the location of an event sink in a mobile system has attracted recent research [1]. Our system design has the inherent ability to incorporate dynamicity in the location of event receivers. We will evaluate this in the presence of real-world mobility statistics.

References

[1] J. Bates *et al.*, "Integrating Real-World and Computer-Supported Collaboration in the Presence of Mobility," IEEE WET-ICE (Workshop on Emerging Technologies), 1998.

[2] A. Rifkin and R. Khare, "A Survey of Event Systems," http://www.cs.caltech.edu/~adam/isen/event-systems.html

[3] A. Carzaniga, D. S. Rosenblum, and A. L. Wolf, "Design and Evaluation of a Wide-Area Event Notification Service," *ACM Trans. Comp. Sys.*, vol. 19, no. 3, 2001, pp. 332–83.

[4] TIBCO, http://www.tibco.com

[5] T. Speakman *et al.*, "Pragmatic Generalised Multicast," work in progress, http://search.ietf.org/internet-drafts/draft-speakman-pgm-spec-07.txt

[6] B. Cain and D. Towsley, GMTS "Generic Multicast Transport Services," *Proc. Net. 2000*, Paris, France, May 2000, http://www.east.isi.edu/RMRG/cain-towsley3/

[7] C. Diot *et al.*, "Deployment Issues for the IP Multicast Service and Architecture," *IEEE Net.*, Special Issue on Multicasting, Jan./Feb. 2000.

[8] L. Opyrchal *et al.*, "Exploiting IP Multicast in Content-Based Publish-Subscribe Systems," *Lect. Notes in Comp. Sci. 1795*, Heidelberg, Germany, 2000, pp. 185–207.

[9] Y. Chu, S. Rao, and H. Zhang, "A Case for End System Multicast," *Proc. ACM SIGMETRICS*, Santa Clara, CA, June 2000, pp 1–12.

[10] F. Costa and G. Blair "Integrating Meta-Information Management and Reflection in Middleware," *2nd Int'l. Symp. Distrib. Objects and App.*, Antwerp, Belgium, Sept. 21–23, 2000, internal rep. no. MPG-00-20, pp. 133–43.

[11] A Hombrecher *et al.*, "Reconciling Event Taxonomies across Administrative Domains," work in progress (submitted for publication).

[12] P. Pietzuch and J. Bacon, "Hermes: A Distributed Event-Based Middleware Architecture," *Proc. 1st Int'l. Wksp. Distrib. Event-Based Sys.*, Vienna, Austria, July 2002.

[13] L. F. Cabrera, M. B. Jones, and M. Theimer, "Herald: Achieving a Global Event Notification Service," *Proc. 8th Wksp. Hot Topics in OS*, 2001.

[14] G. Cugola, E. Di Nitto and G. P. Picco, "Content-Based Dispatching in a Mobile Environment," 2001.

[15] G. Cugola and E. Di Nitto, "Using a Publish/Subscribe Middleware to Support Mobile Computing," *Middleware for Mobile Comp. Wksp.*, 2001.

[16] A Hombrecher, J Bacon, and K Moody, "Federating Heterogeneous Event Systems," *10th Int'l. Conf. Coop. Info. Sys.*, Oct. 30–Nov. 1, Univ. of CA, Irvine.

Additional Reading

[1] A. Mankin *et al.*, "IETF Criteria for Evaluating Reliable Multicast Transport and Application Protocols" RFC 2357, June 1998.

Biographies

JON CROWCROFT [SM] (jac22@cam.ac.uk) is Marconi Professor of Networked Systems in the Computer Laboratory at the University of Cambridge. Prior to that he was at University College London (UCL) Computer Science, where he ran a number research projects in multimedia communications for 20 years. He graduated in physics from Trinity College, Cambridge University in 1979, and gained his M.Sc. and Ph.D. in computing from UCL. He is a member of the ACM, and a Fellow of the BCS, the IEE, and the Royal Academy of Engineering. He was a member of the IAB, general chair for ACM SIGCOMM '95–'99, and Program Committee co-chair for SIGCOMM 2003. With Mark Handley he co-authored *WWW: Beneath the Surf* (UCL Press) and *Internetworking Multimedia* (Morgan Kaufman); he also authored *Open Distributed Systems* (UCL Press/Artech House). He has just published with Iain Phillips from Loughborough a *Linux Kernel Networking Implementation* book.

JEAN BACON is a reader in distributed systems at the University of Cambridge Computer Laboratory. She is Editor in Chief of *Distributed Systems Online*, the IEEE Computer Society's first Web-based magazine (http://dsonline.computer.org/) and is an editorial board member of its sponsoring magazines, *Internet Computing* and *Pervasive Computing*. Her research interests are in the areas of event-driven middleware and open, distributed, role-based access control. She is a member of the IEEE Computer Society's Board of Governors.

PETER PIETZUCH is a second year Ph.D. student at the University of Cambridge and a graduate member of Queens' College. He is working in the Computer Laboratory as part of the Opera Research Group under the supervision of Dr. Jean Bacon. His work is funded by QinetiQ, Malvern. In 2000 he received a B.A. degree after finishing the Computer Science Tripos at Cambridge. At the time hewas a member of Girton College. His main research focus lies in the area of event-based (publish/subscribe) systems. He is investigating how event-based techniques can be used to build next-generation middleware architectures for large-scale Internet-wide distributed applications. He is currently working on Hermes, a scalable distributed event-based middleware architecture. He is also interested in peer-to-peer overlay networks and application-level routing algorithms. Peer-to-peer routing can be applied to networks of event brokers for disseminating events in a publish/subscribe system.

GEORGE COULOURIS leads the QOS-DREAM project on quality of service for dynamically reconfigurable multimedia systems. His other recent research has been in the fields of computer-supported cooperative work and its applications in medicine, middleware for distributed multimedia and security models for groupware. He is an Emeritus Professor of Computer Systems in the Department of Computer Science, Queen Mary and Westfield College, University of London. He is the author of the well-known text *Distributed Systems — Concepts and Design* with Jean Dollimore and Tim Kindberg (3rd edition, Addison-Wesley, 2001).

HANI NAGUIB is a research associate of the University of Cambridge Engineering Department, working on the QoS-Dream project with Professor Coulouris. He is in the L.C.E. Group in the Information Engineering Division.

We believe that our proposed system can provide a number of engineering and performance enhancements over previous event notification architectures. Future work will evaluate these.