ASIA: Application-Specific Integrated Aggregation for Publish/Subscribe Middleware[*]

Sebastian Frischbier
TU Darmstadt
frischbier@dvs.tu-darmstadt.de

Alessandro Margara
Vrije Universiteit Amsterdam
a.margara@vu.nl

Tobias Freudenreich
TU Darmstadt
freudenreich@dvs.tu-darmstadt.de

Patrick Eugster
Purdue University
p@cs.purdue.edu

David Eyers
University of Otago
dme@cs.otago.ac.nz

Peter Pietzuch
Imperial College London
prp@doc.ic.ac.uk

ABSTRACT

The publish/subscribe (pub/sub) communications paradigm is suitable for building large-scale, widely distributed applications. Distributed pub/sub middleware scales well because it decouples communicating clients. However, *complete* decoupling of clients make it more challenging to design distributed applications using pub/sub middleware: often clients want *some* information about each other. We thus augment the pub/sub communication model through addition of an integrated aggregation mechanism—ASIA—that facilitates bidirectional exchange of information without compromising scalability. Our prototype implementation demonstrates that ASIA can be integrated into a typical distributed pub/sub middleware with little effort, and that the aggregation capability adds little overhead in terms of message throughput and latency.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*Distributed applications*

General Terms

Design, Management, Measurement

1. INTRODUCTION

Most large-scale distributed applications, such as social networking sites, logistics management and server monitoring, require communication between software components that are deployed over a large number of sites. In many cases, these applications can derive benefit from *publish/subscribe* (pub/sub) communications middleware: messages can be disseminated efficiently from information producers (publishers) to consumers (subscribers) based on a fine-grained specification of subscribers' interests. *Distributed pub/sub middleware* involves the construction of an overlay network of message brokers that can build dissemination graphs linking publishers and subscribers. Its potential for scalability is based on the *decoupling* of publishers and subscribers: subscribers remain largely anonymous, with unidirectional flow of information from publishers to subscribers.

Despite its potential applicability, distributed pub/sub middleware has not been as widely adopted as would be expected [4]. An obstacle is the complete decoupling of publishers and subscribers, making it more challenging to support applications that require any additional information exchange between them without compromising scalability.

Consider a global logistics organisation that tracks the delivery status of its goods, relying on large-scale enterprise architectures in combination with decentralised cyber-physical systems (e.g. RFID [5] and wireless sensor networks) to manage freight across different means of transportation. Data about the delivery status and position of each good is desired along the supply-chain and provided in near real-time to the logistics provider, its sub-contractors and end-customers. However this may come at a significant cost in terms of energy (e.g. battery-powered sensors), or where manual checking of data is required. Thus, usage information about demand and supply of tracking data has to be obtained.

Current options to obtain such desired information include the use of less scalable, centralised pub/sub middleware systems, group communication systems that provide a form of membership view [1], or straightforward extraneous direct point-to-point communication between software components. Several dedicated distributed aggregation systems have also been proposed, e.g. SDIMS [7] and Adam2 [6]. However, deploying such a stand-alone system alongside a pub/sub middleware has multiple drawbacks, duplicating network connections because it cannot access the brokers' routing information and increased network traffic as aggregation results cannot be piggy-backed [2]. The alternative that we provide is an *Application-Specific Integrated Aggregation* mechanism—ASIA—for distributed pub/sub middleware. ASIA[1] allows the various components of the dis-

[*]Supported by German BMBF Software-Cluster EMERGENT (01IC10S01), LOEWE Dynamo PLV, Dutch national program COMMIT, NSF grant 0644013, Alexander von Humboldt Foundation, DARPA grant N11AP20014. The authors assume responsibility for the content.

[1]`www.dvs.tu-darmstadt.de/research/events/asia/`

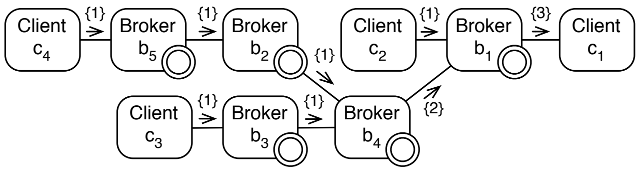

Figure 1: Aggregation messages regarding subscriber counts reaching client c_1

tributed system to collect (aggregated) information about each other, e.g. their interests, or their publication rates, thus enabling reactive behaviours.

2. ASIA MODEL

ASIA *integrates* a conventional mixed topic and content-based distributed pub/sub model with an *aggregation mechanism*, which collects and delivers information as required by applications. Fig. 1 shows an example subscriber count aggregation reaching client c_1. Our approach provides the following features:

Integration: ASIA does not rely on a centralised aggregation system and introduces little overhead. It uses the pub/sub system's routing trees for aggregating data. This provides two advantages: (*i*) it eliminates the costs for creating and maintaining a separate infrastructure; (*ii*) it enables piggy-backing of information to existing messages.

Precision-control: ASIA provides clients with a mechanism to specify the desired precision of their requested data. We can either lower the precision of distributed aggregation computation to reduce the number of aggregation-related messages sent, or have more precise data with a higher additional message load. In our logistics example, it may be costly in terms of battery power for containers to frequently report their GPS coordinates. By using ASIA, the containers can receive a "back channel" informing them of the number of subscribers to their messages. If a container's delivery priority is escalated, it can react by expending more energy to report its position more frequently.

Broker-state: ASIA has access to internal broker state and can provide clients with aggregated system metrics. For example, salient properties about the broker network itself can be reported to interested clients.

Bidirectional feedback: ASIA supports both downstream (publisher to subscriber) and upstream (subscriber to publisher) aggregations, by exploiting the overlay topology maintained by the pub/sub middleware for event propagation.

ASIA thus allows for decoupled, distributed communication between publishers and subscribers while still providing them with information about the system.

3. IMPLEMENTATION AND EVALUATION

We have implemented ASIA within the REDS open-source pub/sub middleware [3], which provides a framework of Java classes and defines the architecture of a generic broker using a set of components with well-defined interfaces. Our implementation of ASIA in REDS requires fewer than 9000 new lines of code, and no changes to existing code: we thus believe that other pub/sub middleware can be similarly extended without major refactoring efforts. To evaluate ASIA, we implemented several different aggregation functions, including subscriber and publisher counts, subscription and publication rates (over a time-based window), and counts of active publishers (in a time-based window). This provides a reusable set of basic operations (e.g. sum, moving average, etc.).

With our experimental evaluation we want to gauge the overhead introduced by ASIA when compared to a "bare" pub/sub system, and to understand the benefits of integrating aggregation with pub/sub. We consider three metrics: (*i*) overall network traffic generated, (*i*) maximum throughput, and (*iii*) delay for delivering messages to clients.

We use 32 Intel Core i7 nodes (8 x 3.4 GHz, 8 GiB RAM, Linux 3.0.3). We consider a network of 16 brokers, each connected to 3 other brokers and serving 100 clients. An additional 16 nodes host the clients.

Overhead. ASIA does not introduce a visible overhead with respect to a traditional pub/sub system, both in terms of throughput and in terms of delay for message delivery. Moreover, the low traffic overhead introduced by ASIA's aggregation mechanisms rapidly decreases when a higher imprecision is allowed. ASIA's traffic is close to that of the baseline pub/sub system at higher imprecision levels, despite providing aggregation information.

Benefits. Compared to a separate aggregation system, ASIA reduces overall network traffic as aggregation update messages are piggy-backed to the data packets exchanged between brokers, whenever possible. Moreover, the integration of the aggregation mechanisms within the pub/sub middleware makes it possible to re-use the same overlay, without incurring an additional cost for building and maintaining a separate infrastructure.

We conclude that ASIA is a practical extension of pub/sub middleware to provide clients with aggregation information that can be computed in a cheap, distributed manner.

4. REFERENCES

[1] G. Chockler, I. Keidar, and R. Vitenberg. Group communication specifications: a comprehensive study. *ACM Comput. Surv.*, 33(4):427–469, 2001.

[2] G. Cugola, M. Migliavacca, and A. Monguzzi. On adding replies to publish-subscribe. In *DEBS'07*, 2007.

[3] G. Cugola and G. P. Picco. REDS: a reconfigurable dispatching system. In *SEM'06*, 2006.

[4] D. Eyers, T. Freudenreich, A. Margara, S. Frischbier, P. Pietzuch, and P. Eugster. Living in the present: on-the-fly information processing in scalable web architectures. In *CloudCP*, 2012.

[5] S. Frischbier, K. Sachs, and A. Buchmann. Evaluating RFID Infrastructures. In *ITG-Fachbericht - 2. Workshop RFID*, 2006.

[6] J. Sacha, J. Napper, C. Stratan, and G. Pierre. Adam2: Reliable distribution estimation in decentralised environments. In *ICDCS'10*, 2010.

[7] P. Yalagandula and M. Dahlin. A scalable distributed information management system. In *SICOMM'04*, 2004.