1. The Evolution of Publish/Subscribe Communication Systems*

Roberto Baldoni, Mariangela Contenti, and Antonino Virgillito

Dipartimento di Informatica e Sistemistica
Università di Roma "La Sapienza"
Via Salaria 113, 00198 Roma, Italia.
{baldoni,contenti,virgi}@dis.uniroma1.it

1.1 Introduction

In the last years, a growing attention has been paid to the publish/subscribe (pub/sub) communication paradigm as a mean for disseminating information (also called events) through distributed systems on wide-area networks. Participants to the communication can act as *publishers*, that submit information to the system, and as *subscribers*, that express their interest in specific types of information. Main characteristics of such *many-to-many communication paradigm* are: the interacting parties do not need to know each other (anonymity), partners do not need to be up at the same time (decoupling in time), and the sending/receipt does not block participants (decoupling in flow). So, the publish/subscribe paradigm has been largely recognized as the most promising application-level communication paradigm for integration of information systems.

Pub/sub systems can be classified in two main classes: the *topic-based* class (e.g. TIB/Rendezvous [1.11]) and the *content-based* class (e.g. SIENA [1.9] and Gryphon [1.3]). In a topic-based system, processes exchange information through a set of predefined subjects (topics) which represent many-to-many distinct (and fixed) logical channels. Content-based systems are more flexible as subscriptions are related to specific information content and, therefore, each combination of information items can actually be seen as a single dynamic logical channel. This exponential enlargement of potential logical channels has changed the way to implement a pub/sub system. As an example, a content-based pub/sub system cannot indeed rely on (i) *centralized* architectures based on (ii) *network level* solutions (such as IP multicast) for subscriptions and information diffusion as a single server could not keep up with a high number of subscribers and the limited number of IP multicast addresses could not fit the potential large number of logical channels. Therefore, very recently it is becoming clear that the best way to implement such systems is at *application-level* through a set of *event brokers* which exchange

* This work is partially supported by a grant from MURST in the context of project "DAQUINCIS" and by grants of the EU in the context of the IST project "EU-Publi.com".

information on a point-to-point basis through a TCP/IP platform. The cooperation among the event brokers has to take into account the following basic problems: *subscription and information routing*. Subscription routing refers to the process of disseminating subscriptions among the brokers in order to create a mapping between subscriptions and subscribers. Information routing includes the *matching* and the *forwarding* operations. Matching refers to the operation of identifying the set of receivers (i.e., subscribers) for each information item arrived at a broker. This is done by using the mapping between subscriptions and subscribers. Forwarding refers to the operation of routing through a sequence of TCP connections to get the final destinations. The overall aim of the brokers cooperation is therefore (i) to optimize some application-related performance metrics such as the overall information throughput or the number of TCP hops per information diffusion, and (ii) to ensure the specification of the pub/sub service such as reliable and/or timely information diffusion etc.

1.2 State of the Art

First pub/sub prototypes have been developed and deployed over LANs e.g. TIB/Rendezvous [1.5] and Elvin [1.8]. To reduce the network traffic the native LAN broadcast property was exploited. Publishers multicast information within the broadcast domain of the LAN and then each subscriber locally filters out information that does not match its interest. The next step was to bring a pub/sub system on a WAN in order to enlarge the potentiality of these systems in terms, for example, of number of subscribers. An enhanced version of TIB/Rendezvous based on rendezvous router daemons [1.11], SIENA [1.9] and Gryphon [1.3] are examples of such wide-area pub/sub systems. To get scalability, these prototypes are based on an architecture of networks of event brokers that exchange messages through a TCP/IP platform which provides basic point-to-point connectivity.

They differ each other from the way they implement the basic pub/sub mechanisms (subscription and information routing). More specifically, in TIB/Rendezvous information routing is done without taking into account the content of the information. Each information sent by an event broker (called *router daemon* in TIB terminology) is diffused to the other router daemons through a multicast tree based on the topology of TCP connections between the router daemons. Each router daemon stores information on subscribers of its own LAN and subscription routing is then performed only across LAN boundaries. Gryphon employs an efficient information matching algorithm [1.1], that assumes static subscription tables in which each node maintains all the subscriptions in the system. Therefore subscription routing reduces actually to flood the network of event brokers with each subscription. Siena embeds subscription routing algorithms [1.2] that specify how the subscription table at each broker is maintained, preventing subscriptions flooding.

This is done by exploiting containment relationships among subscriptions. All previous systems share a common factor. The forwarding operation is done through a fixed topology based on TCP connections. However, to improve the overall application performance, a dynamic and self configuring underlying TCP topology would help to reduce for example the number of TCP hops to be traversed by an information to reach the set of intended subscribers.

Such dynamic TCP topologies are similar to *overlay network infrastructures* (e.g. Overcast [1.4], Tapestry [1.13], Pastry [1.6], i3 [1.10]) studied in the context of scalable information diffusion applications (e.g. multimedia streaming applications). Overlay networks main target is to set up on-the-fly an application-level distribution tree where the root routes content messages (e.g., a "live" video) to a dynamic group of clients. They aim at achieving high performance (as trees are built to optimize network service parameters), robustness (as trees can rearrange to route around failed nodes), and generality (because any host connected to the Internet can join the network). The notion of overlay network matches very well the structure of a topic-based pub/sub system when considering each topic is associated with a multicast tree (e.g., Bayeux [1.12] and Scribe [1.7]) and information routing is done exploiting a basic lookup service of the overlay network. This approach cannot be employed, as it is, by content-based systems since there can be a different multicast tree for each published information.

1.3 Research Directions

Pub/sub systems represents a nice meeting point of many research communities such as databases, software engineering and distributed systems. As an example, in the context of databases, the definition of "ad-hoc" languages for subscription declaration and publishing advertising is an open problem. Using SQL as such language would imply an a-priori knowledge of the schema of the database which, in a content-based pub/sub system, corresponds to the a-priori knowledge of the structure of the information space. This could be not available or it could change along the time. In the sequel we point out a few research directions related to distributed computing[1]. We consider at first the problem of defining smart *subscription routing* algorithms that, on one hand, try to limit the diffusion of all subscriptions to every broker maintaining, on the other, a certain degree of availability of the mapping between the subscription and the subscribers. For example, SIENA is not fault tolerant in this sense. If a broker fails, its subscriptions are lost. A solution to

[1] We do not consider important research topic such as security, impact of mobility and performances of a pub/sub system. These aspects are actually orthogonal to any pub/sub solution and they have to be taken into account by any solution in order to be a valid one.

this problem could be to consider the information space partitioned among overlapping subsets where each broker is responsible of the mapping between subscribers and subscriptions of a given set. In this way a subscription could be stored into more than one broker and, therefore, the failure of a broker does not lead to the loss of that subscription. Criteria and techniques for defining the sets (or other data structures), for assigning sets to brokers and to master the failover phase have to be studied.

Another issue is the designing and the implementation of *information routing based on dynamic TCP topologies* or, more specifically, how to effectively exploit overlay networks techniques in the information routing problem with particular emphasis on content-based pub/sub systems. TCP connections between brokers can be dynamically rearranged to adapt to the logical structure of subscriptions. Such a dynamic mechanism aims at reducing the number of TCP hops per information diffusion, avoiding that a broker acts as pure forwarder of information (i.e., that broker does not store any subscription for that information). At the same time, the delay per information diffusion can be reduced by associating a weight function with each potential TCP connection. This function gives a measure of the available bandwidth for that TCP link in order to decide which pair of brokers has to be connected (a transcontinental TCP link is expected to have much less bandwidth than a TCP link between two brokers in the same campus). Finally, a deep study is needed on the relationship between the subscription routing and information routing, that could for example consider the use of different dynamic underlying topologies optimized for each of the two functions.

Another future direction of research is on the definition of *formal specifications of the services*, in terms of liveness and safety, provided by a pub/sub systems (like the formal specifications of services provided by group toolkits). To our knowledge there is no agreement on such formal specification and this makes difficult, firstly, to give a formal proof of any algorithm implementing a given service, and, secondly, to compare a given algorithm with another one. This formal specification has to take into account pub/sub communication paradigm peculiarities. For example, keeping the anonymity of the end-users implies a pub/sub system cannot provide end-to-end service semantics between a publisher and a subscriber (like in a client/server interaction). It can rather provide a "one-side" service semantics: one for a subscriber and another for a publisher. In the latter case, the pub/sub system guarantees the requirements imposed by the publisher on that information (such as lifetime of the information, information priority etc.). Therefore in the next two paragraph we consider separately the notification semantics and the publishing semantics and some of their implications.

Notification semantics. The notification semantics is often left non-specified. While it is clear which will be the information delivered to subscribers by the pub/sub system, it is not clear the condition which defines *if*, *when* and *how many times* an information will be delivered at a subscriber. This is

amenable to the decoupling in time of a pub/sub system and to the intrinsic asynchrony and concurrency of publish and subscribe operations. The specification of clear notification semantics that defines during a period of subscription which information will be delivered (e.g. all the information seen by the event brokering system that matches the subscription, only the information that matches the subscription since the arrival of the subscription to the even broker etc.) and how many times (e.g at-most once, exactly once and at-least once) this information will be delivered within the subscription interval is mandatory if one wants to use pub/sub systems as communication paradigm for mission-critical or dependable applications.

Publishing semantics. Indirect communication using queuing of messages between a producer and a consumer leaves to the consumer the management of the removal of the messages from the queue (for example when the consumer fetches a message). In a pub/sub system a subscriber has no right to delete an information as the interaction follows a "pull" style. On the other hand some actor has to define a lifetime of an information, otherwise the size of information stored into a pub/sub system could rapidly overflow. Lifetime of an information is clearly publisher dependent. For example if the publisher issues information on a stock quote, the lifetime of this information could be either a fixed amount of time or till the arrival of the next quote. The pub/sub system is then responsible for guaranteeing such policy.

References

1.1 G. Banavar, T. Chandra, B. Mukherjee, J. Nagarajarao, R.E. Strom, and D.C. Sturman. An Efficient Multicast Protocol for Content-based Publish-Subscribe Systems. In *Proceedings of International Conference on Distributed Computing Systems*, 1999.

1.2 A. Carzaniga, D.S. Rosenblum, and A.L. Wolf. Design and Evaluation of a Wide-Area Notification Service. *ACM Transactions on Computer Systems*, 3(19):332–383, Aug 2001.

1.3 Gryphon Web Site. http://www.research.ibm.com/gryphon/.

1.4 J. Jannotti, D. K. Gifford, K. L. Johnson, M. F. Kaashoek, and Jr. J. W. O'Toole. Overcast: Reliable multicasting with an overlay network. In *Proc. of the 4th Symposium on Operating Systems Design and Implementation*, 2000.

1.5 B. Oki, M. Pfluegel, A. Siegel, and D. Skeen. The Information Bus - An Architecture for Extensive Distributed Systems. In *Proceedings of the 1993 ACM Symposium on Operating Systems Principles*, December 1993.

1.6 A. Rowston and P. Druschel. Pastry: Scalable, distributed object location and routing for large-scale peer-to-peer systems. In *IFIP/ACM Middleware 2001*, 2001.

1.7 A. Rowston, A. Kermarrec, M. Castro, and P. Druschel. Scribe: The design of a large-scale notification infrastructure. In *3rd International Workshop on Networked Group Communication (NGC2001)*, 2001.

1.8 B. Segall and D. Arnold. Elvin Has Left the Building: A Publish /Subscribe Notification Service with Quenching. In *Proc. of the 1997 Australian UNIX and Open Systems Users Group Conference*, 1997.

1.9 SIENA Web Site. http://www.cs.colorado.edu/users/carzanig/siena/.
1.10 I. Stoica, D. Adkins, S. Ratnasamy, S. Shenker, S. Surana, and S. Zhuang. Internet indirection infrastructure. In *First International Workshop on Peer-to-Peer Systems*, 2002.
1.11 TIB/Rendezvous Web Site. http://www.rv.tibco.com.
1.12 S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. Katz, and J. Kubiatowicz. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *11th Int. Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2001.
1.13 S. Q. Zhuang, B. Y. Zhao, A. D. Joseph, R. Katz, and J. Kubiatowicz. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, University of California at Berkeley, Computer Science Division, April 2001.