

Internet-scale Event Notification: Architecture Alternatives

Jie Ren

Information and Computer Science,
University of California, Irvine

Definition and Classification of Event

The basis for Internet-scale event notification is the definition and classification of event. The most natural way is adopting the object-orientation view. That is, each event has a type, and the types form a generalization-specialization hierarchy.

The big issue here is: where do these types get defined? Definitely a directory service is needed, and this service should be open and dynamic. But it's unusual to define a directory service for a protocol. The only common directory PROTOCOL is DNS, which provides a very simple mapping between hostname and IP address. In Web, the service is provided by business like Yahoo and Google. There is no service specification about the content of web pages in HTTP. There are other directory protocols, such as LDAP. And there are a lot of type definitions, such as MIME, but they are static. A more relevant example is newsgroup, which has a mechanism for creating a new group dynamically.

The definition of event should contain two parts: the event-type-independent part and event-type-dependent part. In the first part, the most important information would be type of event, id of producer, and id of event itself. The most natural way for the second one would be a set of value-pair definitions.

Protocol and Architecture

Goal

Our goal is to design an architecture and protocol that can deliver various events from publishers to subscribers in a scalable, fast and reliable way.

Existing work and tentative extension

First decision is about subject-based routing or content-based routing. Subject-based routing, as in NNTP, is more efficient. However, it is less powerful and flexible.

A related question is which is the core event service and which could be done as add-on. The number and boundary are still unclear.

Carzaniga's work can serve as a starting point. It is cited in several papers. It is criticized as non-scalable, since it requires that each router has full knowledge about subscription/publication and topology. Its algorithm may not produce stable routing path, and its architecture should be extended to support both intra-domain and inter-domain routing.

TCP itself is an end-to-end point-to-point protocol. It's not designed for group communication, although it can be used to emulate group communication. IP itself is a routed protocol, behind the illusion that the packet goes from one computer to another is the work of ROUTING protocol, which decides where the routed packet should go. We cannot use point-to-point protocol, like TCP and anything above it (such as HTTP), because there are too many point-to-point connections even for a single event. We can't

use IP either, since it also requires address-to-address communication. We can't even only use IP Multicast, due to the semantics gap between primitive multicast group and complex expressive event.

In unicast routing, there are two levels: intra-domain routing, whose standard protocol is OSPF, and inter-domain routing, whose standard protocol is BGP. In multicast routing, in addition to intra-domain multicast routing protocol, such as DVMRP, much current research is about inter-domain multicast routing, such as MASC/BGMP. This suggests us may be we should also include the concept of domain.

In traditional unicast routing, each router needs to main a routing table, which indicates using which neighbor for which destination. There may be a huge amount of destination hosts, which brings a scalability issue. Two methods solve this problem. One is the default router, a router sends every packet that it has no explicit route to the default router. Another one is address aggregation. Router can use network address to designate a large set of destinations. The success lies in the fact that address pattern reflects physical topology, and address-based routing exploits this pattern/topology. IP address is hierarchical, which means they are aggregatable. Unfortunately this is not the case in content-based routing, a complex mapping among subscription, content, and network address should be made, there's no direct relation between an event and address.

A related issue would be the topology/architecture of the event router network. Unicast routing permit any type of graph. This may be a luxury that content-based routing cannot afford. Multicast routing uses tree architecture (Core Based Tree) as their basis. Maybe this can serve as a basis for content-based routing. A group from USC argues generic graph will require too much routing entry, and they suggest a hierarchical configuration, each router has one parent. It participates in two multicast groups: one is his own, another is owned by its parent. (Here they use multicast as a transmission mechanism.) Router would forward each undecidable event toward root. (This is kind of default routing.) Cazarniga and Gryphon also look focusing on tree architecture. The shortcoming of tree architecture is longer latency of delivery (Shortcut can be used to alleviate) and less robust (Dynamic reconfiguration can be used to improve robustness).

There should be two interfaces. One is between publisher/subscriber and the network. This is the typical User-Network-Interface in communication community. Another one is between the routers inside the delivering network, a Network-Network-Interface. Here is the major protocol work.

In summary, Internet-scale event notification produces several challenges in event definition, service definition, architectural topology, and routing protocol. Some of them can find counterpart in traditional software and network research, others still need more exploration and evaluation.

Short Bibliography

- “A Hierarchical Proxy Architecture for Internet-scale Event Services”, Haobo Yu, Deborah Estrin, Ramesh Govindan, USC TR 99-704. This is the paper from USC ISI.
- “Information Flow Based Event Distribution Middleware”, Guruduth Banavar, Marc Kaplan, et.al. Middleware Workshop at the International Conference on Distributed Computing Systems 1999. One of several papers about Gryphon.