WebFilter: A High-throughput XML-based Publish and Subscribe System

João Pereira* INRIA Rocquencourt João.Pereira@inria.fr
Françoise Fabret INRIA Rocquencourt Francoise.Fabret@inria.fr
H. Arno Jacobsen Toronto University jacobsen@eecg.toronto.edu
François Llirbat INRIA Rocquencourt Francois.Llirbat@inria.fr
Dennis Shasha† Courant Institute of Mathematical Sciences
New York University shasha@cims.nyu.edu

1 Overview

Large-scale information dissemination systems for selective information distribution are gaining increasing importance. This is due to the fast proliferation of different communication infrastructures, e.g., Internet and wireless networks, and the increase in mobile computing potential. However, targeted information dissemination will only be effective if the distributed information is pertinent for the user. In this context, the publish and subscribe paradigm is a good interaction model. This interaction model enables publishers to publish information through the publish and subscribe system and provides subscribers with the ability to express their interest within the system in order to be notified afterwards of any publications matching their registered interests.

In the context of information dissemination, requirements for a publish and subscribe (pub/sub) system are threefold: First, the system must achieve interoperability between different devices and infrastructures; One solution is to use eXtensible Markup Language (XML) as the common data presentation format for publications. Second the system must support a (very) expressive filtering language that enables subscribers to accurately define their interests over publications. Finally, it has to be able to manage millions of users, manage high volatility of users’ interests, and filter large quantities of content per unit of time.

We have built a pub/sub system, called WebFilter, that aims at large-scale and high-throughput XML processing for selective information dissemination on the Internet and in wireless environments. The main characteristics of our system are the following:

1. A publication to our system may be any well-formed XML-document.
2. A subscription to our system is an XPath-expression [9] as defined by the W3C. XPath treats an XML document as a tree of nodes. XPath provides a flexible way to specify tree patterns that have to be matched by XML documents.
3. Scalable and extremely efficient matching, WebFilter can manage about 1 Million XPath-expressions and 70 XML-documents per second.
4. WebFilter is dynamically updatable, i.e., subscriptions may be efficiently inserted, updated, and deleted while the system is in operation.

A large number of information dissemination systems have been developed in the context of Message Oriented Middleware and Selective Dissemination of Information communities. In most of these systems, events consist of sets of (attribute, value) pairs and are filtered according to their attribute values [7, 6, 2, 3, 8]. Only a small number of systems [5, 1] supports XML filtering as WebFilter does. Compared to these systems, WebFilter achieves much better performance.

2 System description

The architecture of our system is shown in Figure 1. It processes XML-documents by translating them into
In the demonstration, a user can submit subscriptions that identify her interests in certain subjects by writing XPath expressions using the web interface of our system. For example, users can submit the following query: //real-estate/*/bedrooms[@number = 2].

In this subscription, the subscriber expresses interest in real-estate objects of unspecified kind (i.e., expressed with the wildcard operator "*".) with specifically constrained number of bedrooms. This XPath expression matches XML documents which contain a real-estate element followed by any other level in the document, followed by a bedrooms element with number attribute equal to 2.

3.2 Performance
We demonstrate the performance of WebFilter using the benchmark proposed in [1]. Our experiments show that our system can achieve high throughput even for large numbers of subscriptions. For instance, we have run an experiment where submitted XML documents have an average depth of 4.4 and an average number of 26.3 elements and subscriptions are path expressions having an average depth of 3.34 and contain one filter expression that defines an equality condition over an element attribute. This experiment has been executed on a single-CPU workstation with an i686 CPU at 500MHz and 256MB RAM operating under Linux. The experiment shows that our system can process 70 documents per second for one million of subscriptions.

Acknowledgements: We would like to thank Mehmet Altinol and Michael J. Franklin for making their XML filtering benchmark tool available to us.

References