XTreeNet: Scalable Overlay Networks for XML Content Dissemination and Querying (Synopsis)

William Fenner
AT&T Labs–Research
fenner@research.att.com

Michael Rabinovich
AT&T Labs–Research
misha@research.att.com

K. K. Ramakrishnan
AT&T Labs–Research
kkrama@research.att.com

Divesh Srivastava
AT&T Labs–Research
divesh@research.att.com

Yin Zhang
University of Texas, Austin
yzhang@cs.utexas.edu

Abstract

XML is becoming a ubiquitous format for information exchange on the Internet. To alleviate the problems of "whom to ask" and "whom to tell" when connecting XML information producers with consumers over the network, content-based querying and dissemination of information have been investigated in the literature. Our XTreeNet project unifies the publish/subscribe and query/response models with a single common XML aware overlay network for XML-based information producers and consumers. This integrated framework lends itself to a variety of applications.

1 Introduction

XML-aware Networks: Information is increasingly being created, exchanged and stored in the eXtensible Markup Language (XML). XML is suitable for this purpose because of its flexibility and self-describing nature: it is human readable, while at the same time it is convenient for machine processing. Examples of XML-based information include a large number and variety of electronic newspapers, technical journals, and healthcare databases.

These information producers are typically geographically dispersed across the network, as are the potential consumers of this information. Communicating XML-based information presents opportunities and challenges from a networking perspective. A network that can efficiently forward XML data can be advantageous, as it can offload the filtering of tremendous volumes of data from consumers. Or, it can reduce the load on the producer and the network, by avoiding producers having to broadcast information to individual consumers. In this synopsis, we argue that an XML-aware overlay network infrastructure is needed, both for performance and functionality reasons, to efficiently connect producers and consumers of this information.

Integrating Publish/Subscribe with Query/Response: As more and more information is becoming available electronically, with rapid globalization, both information consumers and information producers face increasing challenges. For information consumers, it is very difficult to determine "whom to ask" when it comes to getting the appropriate information available over the network. For information producers that would like their content to be made available to interested consumers, it is very hard to identify "whom to tell". The solution to both these problems is to support content-based dissemination and content-based querying of information, where the network itself is responsible for identifying "whom to ask" and "whom to tell".

For a consumer who is interested in a topic of long-standing interest, publish/subscribe overlay networks are very useful. The network facilitates the dissemination of the information by setting up the appropriate distribution trees based on the content and subscriber interests. In recent publish/subscribe overlay networks (see, e.g., ONYX [5]), subscriber interests are aggregated at each node in a producer-based distribution tree, and published information needs to be repeatedly matched against subscriber interests at each node along the overlay paths to relevant subscribers. Such approaches can be expensive computationally, potentially limiting the throughput of the system.

When the consumer is interested in a specific piece of information, or in a transient or a newly emerging topic, a database-style query/response model is more appropriate than a publish/subscribe model. The framework for querying these networked information producers has traditionally been quite different from the publish/subscribe model. Individual queries are posed to individual producers for processing. The only external support that may be available are search engines that help locate producers relevant to

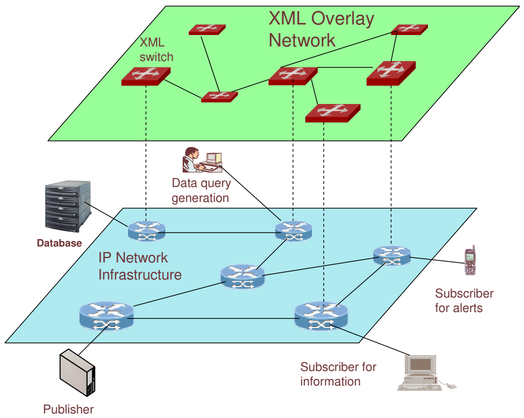

Figure 1. XTreeNet Overlay Network

user queries. However, this does not always lend itself to providing up-to-date real-time information to the query, because the location information at the search engine may be limited and out of date.

Information being disseminated electronically and in large scale from multiple producers has also naturally evolved over the years. For example, subscribers to electronic news publishers see "snippets" of information being presented to them. A subsequent request delivers the actual published information, for selected snippets. This naturally demands an environment that seamlessly integrates a publish/subscribe framework (presenting snippets of subscribed material from a myriad of producers) and a query/response framework (for responding to the specific information requests from subscribers).

XTreeNet Overlay Network: Our XTreeNet project (from "<u>X</u>ML <u>Tree</u>-based <u>Net</u>works") unifies the publish/subscribe and query/response models with a single common XML aware overlay network of XML routers that connect XML-based information producers and consumers (see Figure 1). XTreeNet is based on two key conceptual ideas:

- First, XTreeNet introduces the concept of a *content descriptor* (CD), used to describe the information that consumers want to receive and that producers generate. This permits producers and consumers to be *decoupled* from each other, which is important for scalability.

- Second, XTreeNet sets up a core-based distribution tree for each CD. Since there is no explicit matching of data against complex consumer interests, data and query forwarding are very efficient.

XTreeNet additionally enables such capabilities as duplicate elimination, relevance-score based filtering, and access control to be performed in the network. This integrated framework lends itself to a variety of applications, including one that we discuss in this synopsis—a "super newspaper" application that connects thousands of electronic newspapers to millions of readers across the network who want to get news on topics of interest from any and all of the publishers of the content.

2 Testbed Application and Capabilities

As a shared infrastructure, the network tries to find a sweetspot between being generic enough to support a large range of applications (as IP attempts to do) and providing critical functionality where appropriate. In the past, when the need for efficient application-level support within the network was desired for pervasive applications, we have seen the emergence of such network support. Examples include L7 switches, application-aware firewalls, and SSL accelerators. The introduction of application level support into the network is desirable when there are significant performance gains from offloading considerable amounts of duplicate effort at a large number of end-systems, or when functionality rightfully belongs in the network, such as the need for a trusted intermediary.

With XML becoming a ubiquitous format for data communication, XML-aware overlays will provide support to applications that communicate using this data format. We see the need for XML awareness in the network for both performance reasons (e.g., to offload filtering of data that is considered irrelevant to the end-system) and functionality reasons (e.g., access control).

2.1 Testbed Applications

Our XTreeNet overlay network infrastructure lends itself to a variety of applications. In particular, we plan to use the following two applications as a testbed for XTreeNet.

- One application is large scale news dissemination – a form of a "super newspaper". There are thousands of daily newspapers published electronically worldwide, reporting on a wide variety of topics of local, regional, national, and international interest. This news is of interest to millions of readers worldwide, each of whom may be interested in only a small set of topics (the stocks they own, their favorite sports teams), but wants to get news on these topics from any and all of the worldwide newspapers.

- A second application is "cooperative healthcare". There are thousands of hospitals, clinics, and health care providers who maintain records of patients, and the results of procedures carried out on patients. These records are of considerable interest to millions of patients and the medical doctors treating them, who may be interested in specific information (tests carried out on a specific patient in the last 2 years, techniques used to detect specific diseases), but want to get this information from all possible, relevant sources.

In each of these, and many other, applications, there is a concerted attempt at making information available as XML. For electronic newspapers, the Real Simple Syndication (RSS) XML format has become the dominant format for distributing news headlines on the Web. For healthcare, the Health Level 7 (HL7) consortium is standardizing XML formats for patient records and other aspects of health care, and health care information is becoming increasingly available in XML format.

In this synopsis, we consider the super newspaper application in more detail, focusing on the functionalities required and the utility of the XTreeNet infrastructure in providing these functionalities.

2.2 Content Descriptors

A fundamental question that arises in any producer/consumer model is the basis for matching data generated by producers with consumer interests. Typically, content-based publish/subscribe and query/response models permit consumer interests to be specified as (potentially complex) query filters over the data generated by producers. Identifying consumers interested in a data item is based on checking which of the consumer query filters matches the specific data item. While such an approach is appropriate in a centralized environment, it can be very expensive in a distributed overlay network, where published data would have to be *repeatedly matched* against subscriber query filters at multiple nodes in the overlay network.

In XTreeNet, we use the key concept of a *content descriptor* (CD) to alleviate this problem. Intuitively, CDs work by permitting each data item generated by a producer and each consumer query filter to be *independently* mapped to sets of CDs. A data item is then said to match a query filter only if their respective sets of CDs have at least one CD in common.

As examples, a CD could be an element of an ontological topic hierarchy, or an XML data path. A news data item can be associated with the CD set "/international/politics/mideast" and "/USA/business/energy/oil". A query filter can be of the form "/international/*/mideast", which can be mapped to the set of CDs "/international/politics/mideast", "/international/business/mideast", etc. The overlap between the sets of CDs means that the news data item matches the query filter. Often multiple hierarchies are of interest. For example, news data items can be simultaneously associated with a topic (in a topic hierarchy), as well as a geographic location (in the location hierarchy) of the producer of the news item. For this reason, our CDs support multiple hierarchies.

2.3 Information Publishers/Subscribers

A sports fan who wants to continually obtain information about a particular player, a particular team, etc., may not know all the relevant newspapers on any given day. For example, information about a player may be carried by newspapers in the player's home town, in the state where the player is playing, etc. In the super newspaper XTreeNet application, the fan could simply subscribe to information relating to the specific player. The subscription may also be very fine grained (e.g., only when the particular team wins).

Processing these types of fine grained, content-based, subscriptions requires some application layer function, which the XTreeNet overlay network provides. Without network support, either the subscriber must subscribe at each individual newspaper (clearly a non-scalable option) or the super newspaper provider must implement a "supersite" for harvesting all the news from individual publishers and processing all the subscribers. Although Google and Yahoo are demonstrating the feasibility of the latter approach, it creates significant traffic concentration and processing bottleneck problems. Replication of the servers across the network would only solve part of the problem since the servers would still need to receive and store all the published content. By offering this functionality in the network, XTreeNet avoids these problems and lowers the bar for new entrants into this space.

2.4 Query/Response

Subscriptions are not adequate in many scenarios, especially for breaking news; in such scenarios, an ad hoc XTreeNet query may be more appropriate. For example, when a large scale disaster strikes (such as the Asian Tsunami), it is quite difficult for friends (who may be geographically distant from the disaster) to get information from the locales close to the disaster. They may not know the names of the news publishers in that part of the world.

The super newspaper enables an ad hoc XTreeNet query to gather all the news from those parts of the world, and enables consumers to get the desired information without having to know specifically the names of the individual newspapers. If the topic of the ad hoc query becomes one of continuing interest (as in the Asian Tsunami case), the one time query can be replaced by a long term subscription.

2.5 Integration of the Models

The super newspaper application naturally requires an integration of the publish/subscribe and query/response models for information dissemination. In an electronic news distribution environment, subscribers are typically presented with "snippets" of the relevant items. With XTreeNet, snippets from a myriad of publishers can be filtered and presented. Unlike the same or slightly different piece of information from each publisher being presented multiple times to the subscriber, the XTreeNet overlay can deliver a single snippet, with the subscriber being confident of subsequently obtaining all the relevant information when he/she desires it. Distribution of snippets requires publish/subscribe functionality, and the subsequent request for specific information follows the query/response model. Thus, the integration of the publish/subscribe and query/response models for information retrieval and distribution is not only a desirable feature, it is an essential component of large scale information dissemination. XTreeNet recognizes and addresses this directly.

2.6 Duplicate Elimination and Top-K Filtering

A substantial benefit that XTreeNet provides is duplicate elimination, where only one copy of the document is delivered to the client. Without this, the burden for duplicate elimination falls on the client, which also imposes unnecessary load on the network for delivering the redundant copies that are discarded by the client anyway. In our sports fan example, the fan would naturally want the super newspaper application to remove duplicates, so that the same news published by multiple newspapers (e.g., when the source is a news agency such as Reuters or the Associated Press) is presented to the fan only once.

Another related functionality that can be provided by XTreeNet is relevance-score based filtering. In the publish/subscribe scenario, this functionality filters out any content with relevance scores lower than the top-K scores of documents previously delivered to the same clients. In the query/response case, the network delivers only the K most relevant responses to the query. By implementing this functionality in the network, unwanted content can be discarded early in the network.

2.7 Access Control

XTreeNet can also offer fine-grained access control, at the individual XML element level. While this functionality may not be relevant to our super newspaper application, it is very important in the cooperative healthcare application. For example, an XML document, containing patient information, may be delivered in full when requested by a medical doctor, and in an abridged form, without the sensitive medical history, to hospital administration.

Information producers routinely offer access control functionality and filter content sent to consumers according to their credentials. However, with a large and growing number of information producers in intranets, and with increasingly complex B2B interactions, managing access control rules across this variety of (sometimes administratively independent) information producers grows ever more problematic. The network, as a trusted intermediary, is a logical point to support access control in a systematic, coherent, and hard-to-bypass way.

3 Protocols

3.1 Overview

The key to the protocol framework is the need to support both a publish/subscribe as well as query/response model for information dissemination in a scalable manner. When every overlay router in the network has to perform matching and filtering of XML content from publishers, it becomes an impediment to making the framework scalable. From our past experience with packet networks, the fundamental building block for fast, scalable forwarding is to make the lookup at each node as simple as possible—such as a longest-prefix match or the lookup of a hash identifier. Our approach also recognizes that multicast offers a sound basis for minimizing the number of duplicate copies of a piece of data that traverse a link. However, a multicast framework such as IP multicast is limiting in scalability because of the global limit on the number of multicast groups supported.

The key idea we use is to exploit overlay multicast, with the flexibility of having a group for each distinct content descriptor (CD), so that we can have very fine grained distribution trees on an as-needed basis. CDs form the basis for a content-based multicast group that enjoins producers of information with a CD and the consumers interested in that CD. For every such CD, we conceptually construct two core-based multicast trees with a common core: one that leads to all of the producers, and one that leads to all of the consumers. We create a "coordinator" for each such CD, which is selected dynamically, based on the first arrival of the CD from a producer. The coordinator acts like a "core" in the core-based IP multicast framework. As soon as a CD (from a producer or from a consumer) hits the network, it is mapped into a single hash identifier at the first overlay router. This enables efficient mapping to the corresponding multicast trees for forwarding. Subsequent routers beyond the first only forward based on the hash identifier on the appropriate multicast tree. Since different coordinators are responsible for different CDs, XTreeNet reduces the likelihood of traffic concentration in the network.

We describe next an overview of the XTreeNet protocol that supports the publish/subscribe and query/response framework in an integrated manner. We refer to Figure 2 that shows a network of overlay XML routers R_i, and data nodes (publisher or subscriber end-systems) D_i. We assume that each XML router knows the overlay topology.

3.2 Constructing Trees to Publishers

A key characteristic of our protocol is the creation of a "coordinator" for each CD that anchors further distribution tree construction. The overlay XML router next to

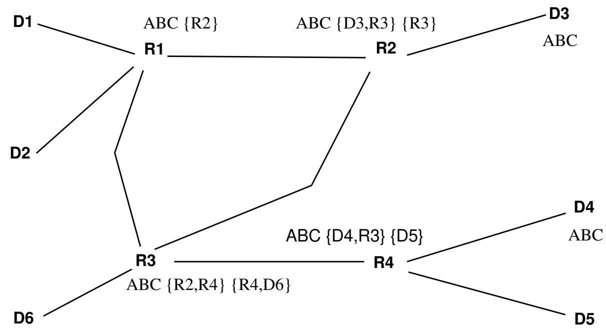

Figure 2. XTreeNet: Protocol Example

the first publisher that transmits a particular CD elects itself as the "coordinator" for the CD and floods that information throughout the overlay network (if there are parallel attempts by multiple coordinators to flood, the coordinator with the smallest ID wins).

So, e.g., if a data node D3 in Figure 2 produces a data item with CD = <ABC>, the overlay XML router R2 elects itself as the coordinator. The coordinator constructs a hash value based on the CD, and then floods to all the overlay router nodes in the network the tuple (hash ID, <ABC>, coordinator-id). The state stored in all the overlay XML routers includes the tuple (hash ID, <ABC>, publisher interface list), where the last item contains the list of interfaces towards the publishers for this CD, initially just the interface to the coordinator, for the non-coordinator routers and the interface to D3 for the coordinator.

If subsequently, another data node D4 publishes the same CD of <ABC>, the adjacent overlay router R4 sends a "publisher-join" to the interfaces on its publisher interface list. In our example, this means sending the join through the overlay topology towards the coordinator, R2. This goes to R3 to R2, along the shortest path of overlay hops. When the overlay XML router R3 receives the "publisher-join", it adds its link towards R4 to the list of "publisher interface list" for this CD. Similarly, overlay XML router R2 adds the link towards R3 in the "publisher interface list" for this CD. Thus, we have a tree of publishers formed. Figure 2 reflects the routing state at this stage, with the first bracketed list at each router representing the publisher interface list.

Subsequent CDs that are announced by other publishers follow the shortest path of overlay hops towards the coordinator, but do not need to progress all the way to coordinator once they hit an on-tree (of publishers) node. For example, if D6 published a document with <ABC>, its publisher-join would stop at R3 because the latter is an on-tree node. A router determines that it is an on-tree node if its publishers interface list contains more than one entry. The same state is maintained in all the overlay XML routers in the network, including the coordinator.

3.3 Query Processing

If an end-user on node D1 sends a query for the content descriptor <ABC>, the first overlay XML router, R1, adjacent to D1, processes the queried CD to derive a hash function (or potentially a set of hashes if the submitted query corresponds to multiple CDs) to match what was computed earlier based on the CD. With the hash, the overlay XML router R1 now knows the relevant tree, and forwards the message containing the (<hash>, <Query>) on the interfaces in the publisher interface list. Note that this list is guaranteed to include the interface towards the coordinator. In our example, the query is sent towards the coordinator R2. R2 will in turn send the query on its publisher interface list, which includes D3 and R3. R3 will forward the query to R4, and then to D4. Thus, the query will be delivered to all relevant publishers.

The main characteristic of our approach is that only the first overlay XML router R1 has to process the query. The remaining routers do not have to process the query, and just use the <hash> id to determine where to send the query – essentially like traditional multicast – along all the interfaces that are marked with "senders" for that hash id. The hash is key to avoiding the processing of queries at all the routers; instead it results in just a hash lookup. Hash collisions can be made unlikely by having a sufficient number of bits; an occasional collision only leads to redundant messages but does not compromise the system functionality.

Each data producer may have multiple documents that match the query, and may not have exposed the entire internal structure of these documents in the original CD announcements. Thus, it is useful for the data producer to process the entire query.

3.4 Constructing Trees to Subscribers

When a subscriber sends its subscription to its adjacent overlay XML router, its subscriber-join is multicast to publishers over the publisher tree. In the process, the subscriber-join also sets up a second logical tree (subscriber-tree), along which published data will flow. Thus, the routing state entry described in the previous subsection is augmented with a subscriber interface list. Each overlay XML router receiving a subscriber-join adds the interface from which the subscriber join was received to the subscriber interface list for the received CD hash. As with publisher-joins, the propagation of subscriber-join messages stops at overlay routers that are already on the subscriber tree (i.e., has a non-empty subscriber interface list). When a new publisher arrives on an interface of an XML router that already has an existing subscription for this CD, i.e., the overlay XML router is an on-subscriber tree node, then the subscriber join is sent to the overlay XML router adjacent to the publisher, thus extending the subscriber tree up to that publisher.

In the example of Figure 2, let us say that D5 wants to subscribe to updates to the content described by the CD = <ABC>. This subscriber join is sent to R4, the first overlay XML router. The state for the subscription tree is created at R4, which forwards along the publisher interface list, i.e., to overlay XML router R3. R3 forwards it (in addition to creating state for this subscription tree) to R2.

A subsequent subscription from node D6 for the same CD (an identical subscription to a previous one) will stop at R3 because it is an on-tree node for the subscription tree. R3 adds D6 to its subscriber interface list, but there is no need to propagate the subscription further, because all publication of content corresponding to CD = <ABC> will be forwarded to R3 and hence by R3 to D6. The routing state at this stage is shown in Figure 2, with the second bracketed lists representing the subscriber interface lists.

There are several other cases to be considered, including subscriptions with wildcards, subscription cancellation, dealing with duplicate detection, filtering based on relevance scores, and access control. We expect to provide a more complete description in a future detailed paper.

4 Related Work

A large body of work exists in the area of distributed content-based publish/subscribe systems, including suggestions to implement them on top of generic application-level multicast [2] or IP multicast [1]. Recently, proposals have appeared for ad hoc XML query routing as well [6].

Most closely related to our own work are XML publish/subscribe systems [3, 5]. ONYX, a system for XML content dissemination (Diao et al., [5]), uses source-based trees for publishing new data. Specifically, there is a dissemination tree rooted at each publisher. Each router maintains for each interface an aggregate subscription that summarizes all the subscriptions downstream along that interface. A published data item starts from the root (the publisher), and gets forwarded to all downstream interfaces whose corresponding aggregate subscriptions match the data item. Although the per-source dissemination tree used by ONYX is optimal, it is also very expensive to maintain especially when the number of publishers is large. Another important distinction between ONYX and XTreeNet is that in ONYX, every overlay router along the path needs to match aggregate subscriptions (XQuery operations, for ONYX) to data items, which can be fairly expensive. In XTreeNet, only the first hop (source overlay) router needs to map data items to CDs, whereas the intermediate routers only need to do forwarding. Chand and Felber [3] take a similar approach using sender based trees, while also providing support for subscription cancellation, which is not addressed in [5]. XTreeNet also supports subscription cancellation. Very recently, SemCast [7] was proposed. Like previous approaches, it aggregates subscriptions and matches a document to such aggregates. The difference is that the choice of the aggregate is made in a centralized manner using a cost-based model, and documents are routed through the network based on the subscription aggregates.

Commercially, DataPower [4] offers network appliances that accelerate XML parsing, validation, and encryption. Sarvega [8] also offers an appliance for XML subscription processing. Semandex [9] offers appliances that can be connected into a network, and used for routing a query to relevant data sources. However, these queries use a proprietary XML dialect for describing and querying content, and the queries are processed by each router en-route. XTreeNet protocols avoid XML processing at the intermediate routers.

5 Conclusion

XML is becoming the dominant format for information dissemination and querying on the web. In this synopsis, we make the case for XML-aware overlay routers in the network for scalability and functionality reasons. Fundamentally, our approach is a move towards content-based networking, without end-systems having to determine and maintain contexts to particular source of information, removing the problem of "whom to ask". It enables publishers to trust the network to deliver information securely, efficiently and just the right, desired content to the appropriate end-systems even when there are a tremendous number of subscribers, thus removing the problem of "whom to tell".

References

[1] G. Banavar, T. Chandra, B. Mukherjee, N. Nagarajarao, R. Strom, and D. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *Proc. of ICDCS*, 1999.

[2] A. Bozdog, R. van Renesse, and D. Dumitriu. Selectcast: a scalable and self-repairing multicast overlay routing facility. In *Proc. of ACM workshop on survivable and self-regenerative systems*, 2003.

[3] R. Chand and P. A. Felber. A scalable protocol for content-based routing in overlay networks. In *Proc. of Symposium on Network Computing and Applications*, 2003.

[4] http://www.datapower.com/.

[5] Y. Diao, S. Rizvi, and M. Franklin. Towards an Internet-scale XML dissemination service. In *Proc. of VLDB*, 2004.

[6] G. Koloniari and E. Pitoura. Content-based routing of path queries in peer-to-peer systems. In *Proc. of EDBT*, 2004.

[7] O. Papaemmanouil and U. Cetintemel. SemCast: Semantic multicast for content-based data dissemination. In *Proc. of ICDE*, 2005.

[8] http://www.sarvega.com/.

[9] http://www.semandex.com/.