Content-Based Routing with On-Demand Multicast

Eiko Yoneki and Jean Bacon
University of Cambridge Computer Laboratory
J J Thomson Avenue
Cambridge CB3 0FD, UK
{eiko.yoneki, jean.bacon}@cl.cam.ac.uk

Abstract

This paper presents a novel approach for messaging systems with content-based subscription in mobile ad-hoc networks. Dynamic construction of an event dissemination structure to route events from message producers to consumers is important to support rapidly changing multi-hop topologies over wireless links. The proposed approach uses summaries of content-based subscriptions to construct an event dissemination structure and Bloom filters are used to summarize subscriptions. We extend ODMRP (On-Demand Multicast Routing Protocol) to utilize this context from the middleware tier and construct an optimized dynamic dissemination mesh. ODMRP's mesh topology gives extra reliability compared to the tree structure in unstable mobile ad-hoc network environments.

1. Introduction

Event-based middleware became popular as asynchronous and multipoint communication is well suited for distributed computing applications. In distributed network environments such as P2P networks, it is attractive to construct event-based middleware over overlay networks to provide scalability. Most distributed event-based middleware contains three main elements: a producer who publishes events (messages), a consumer who subscribes his interests to the system, and an event broker with responsibility to deliver the matching events to the corresponding consumers. The first event-based middleware systems were based on the concepts of group (channel) or subject (topic) communication. These systems categorize events into pre-defined groups. In an attempt to overcome the limitation on subscription declarations, the content-based model [3],[14] has been introduced, which allows subscribers to use flexible querying languages to declare their interests with respect to event contents. Events are routed based on their content and consumer interests. A key challenge when building event-based middleware is efficient propagation of subscriptions among brokers. Brokers require this information to forward incoming events exclusively to interested consumers. Filtering out unrelated events can save significant overhead.

Mobile ad-hoc network (MANET) is a dynamic collection of nodes with rapidly changing multi-hop topologies that are composed of wireless links. The combination of mobile devices and ad-hoc networks allows the creation of highly dynamic, self-organizing, mobile peer-to-peer systems. In such systems, mobile hosts continuously change their physical location and establish peering relationships with each other based on proximity. Asynchronous communication is essential to support such MANET environments. There have been efforts to create efficient multicast communication for MANETs (see Section 2.2). Maintaining group membership and efficient delivery of the packets to all members is challenging. Nodes that are increasingly mobile have a limited transmission range; a source-to-destination path could pass through several intermediate nodes, leading to a dramatic increase in complexity. To achieve improved one-to-many communication systems in MANET environments, event-based middleware must be introduced.

We propose a content-based publish/subscribe system for MANETs, which integrates an extended ODMRP (On-Demand Multicast Routing Protocol) [4] and content-based subscriptions. ODMRP builds a routing table on demand, and it has performed well with regard to throughput and control packet overhead in simulation studies [11]. ODMRP supports optimized data dissemination mechanisms with context awareness including location, network topology, network ability (e.g., bandwidth and stability) and mobility. Dynamic construction of the event dissemination structure is essential to support messaging systems in MANET environments. To optimize construction of an event dissemination structure, the proposed system defines an interface to apply the context from a publish/subscribe system to ODMRP. The context is summarized subscriptions and notifications. The interface is generic to supply data to be attached to the ODMRP packet and indicate how to process them. Content-based subscriptions at a broker node are aggregated and summarized into a Bloom filter [2], and the event source broker node defines the multicast group by examining the propagated subscriptions. Context-awareness allows both middleware and network layer components to exploit information to provide an efficient and dynamic event routing mechanism for better performance.

This paper continues as follows: section 2 describes the background and related work, section 3 describes the proposed publish/subscribe system, and it concludes with section 4.

2. Background and Related Work

The messaging system in MANET should be self organized, because the topology of a mobile P2P system has to constantly adjust itself by discovering new communication links and also needs to be fully decentralized due to the lack of a central access point. In non-ad-hoc P2P networks, many prototype publish/subscribe systems have been reported, including Gryphon[8], JEDI [6], SCRIBE [16], HERMES [14], and SIENA [3]. SCRIBE is a topic-centric publish/subscribe messaging system, while Hermes [14] offers a scalable event broker system. XRoute [17] takes an approach to aggregate subscription to a tree pattern. However, in ad-hoc networks, much research currently focuses on general datagram routing in both unicast and multicast routing, but no definite solution to define publish/subscribe semantics using these protocols has been provided. Given the characteristics of MANET, the use of distributed hash tables in a messaging system to locate objects may not work in dynamic environments.

2.1. Content-Based Subscription and routing

Topic-based addressing is an abstraction of numeric network addressing schemes. With the content-based subscriptions used in SIENA and Gryphon, delivery depends only on message content, extending the capability of event notification with more expressive subscription filters. Common topic-based systems arrange topics in hierarchies, but a topic cannot have several super topics. Type-based subscription provides a natural approach to this if the language offers multiple sub-typing, thus avoiding explicit message classification through topics. This works well with typed languages, but it is complex to deploy this degree of serialization of objects. Moreover, mobile applications may not have the concept of objects or typing. Thus, the combination of hierarchical topics and high speed content filtering could be a more flexible approach for mobile applications. There are efforts to build content-based subscription with distributed hash tables by automatically organizing the content into several topics. Research is also ongoing to structure complex content-based data models [12] and reflection-based filters [7]. XRoute [17] proposes an approach for content-based routing of XML data in mesh-based overlay networks. We are currently working on efficient distribution of event matching tasks over the network and brokers by establishing the concept of ”approximate matching”.

The dynamic construction of event dissemination trees to route events from publishers to all interested subscribers is the most challenging task to support content-based subscription in distributed environments. JEDI proposes variations for event routing among its networked event servers, including the flooding and match-first approaches. With the hierarchical approach, event servers are organized in a tree. This approach may lead to a large routing table at the tree root. Routing strategies in SIENA use two classes of algorithm: advertisement forwarding and subscription forwarding. They prune the propagation tree by propagating only those paths that have not been covered by previous requests.

2.2. Multicast

Multicasting has traditionally been used as transport for messaging systems. However today’s multicast schemes are not scalable to support large groups. Thus, several Application-Level Multicast Routing Protocols (ALMRPs) have been designed. Most ALMRPs use tree routing for logarithmic scaling with respect to receiver numbers: as the node connectivity changes, the tree structure changes accordingly. Narada [13] and many subsequent designs can best be understood as two layered protocols: a protocol that maintains a mesh of hosts, and a multicast routing protocol on top of this mesh. Other examples are Bayeux/Tapestry [19] and CAN [15]. The multicast service model is less powerful than that of a content-based network, and there is currently no optimal way of using or adapting the multicast routing infrastructure to provide a content-based service.

For wireless networks, the most natural communication type is broadcasting. The dynamic topology of the network makes it difficult to maintain a multicast group, e.g., providing better bandwidth utilization and less host/router processing, and resolving unknown receiver addresses. There are several multicast routing protocols for MANET. DVMRP (Distance Vector Multicasting Routing Protocol) builds a source-Based tree. AODV (Ad-hoc On-Demand Distance Vector Routing Protocol) builds a core-based tree. CAMP(Core Assisted Mesh Protocol) builds a mesh interconnection of hosts. ODMRP applies an on-demand routing

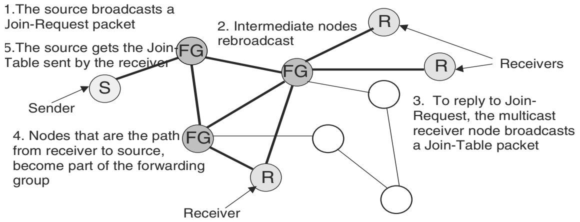

Figure 1. ODMRP Protocol

technique to avoid channel overhead and improve scalability (see Fig. 1). It will attempt to create a group of forwarding nodes between the source and the multicast receivers. These forwarding nodes re-broadcast any packet they receive to reach all interested multicast receivers. The multicast mesh is created through a reply-response phase that is repeated at intervals to keep the routes to the multicast receivers fresh. With the concept of forwarding group, only a subset of nodes forwards multicast packets (scoped flooding). ODMRP provides a richer connectivity among multicast members using a mesh-based approach. It supplies multiple routes for one particular destination, which helps in case of topology changes and node failure. ODMRP takes a soft-state approach to maintain multicast group members. Nodes need not send any explicit control message to leave the group. It can work with any unicast protocol. See more details of ODMRP in [4].

2.3. Bloom filters

Bloom filters [2] are compact data structures for probabilistic representation of a set in order to support membership queries. Each filter provides a constant-space approximate representation of the set. Errors between the actual set and the Bloom filter representation always take the form of false positives to inclusion tests. A Bloom filter is a bit vector, and items are inserted by setting the bits corresponding to a number of independent hash functions. Bloom filters are used in the proposed system to summarize content-based subscriptions.

Constructing and updating Bloom filters: Consider a set A = $a_1, a_2, ...a_n$ of n elements. Bloom filters describe membership information of A using a bit vector V of length m. For this, k hash functions, $h_1, h_2, ...h_k$ with $h_i : X \to 1...m$, are used as described in Fig. 2. Filters can be made incrementally as new elements are added to a set, the corresponding positions are computed through the hash functions and bits are set in the filter.

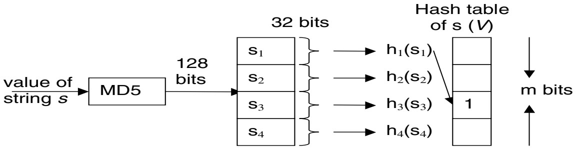

Figure 2. Bloom Filter

3. The Proposed Publish/Subscribe System

The proposed publish/subscribe system follows the basic model of an event brokering system. In MANET environments, it is best to create a routing table on demand (see [5]). Thus, ODMRP is chosen for data dissemination. The MANET environment is dynamic, and the multi-hop routing has to consider many contexts from physical constraints, location, and mobility. Many contexts belong to the network, which are outside the scope of middleware. On the other hand, the semantic contexts from upper layers should be used for building an efficient communication by the network layer component. Here, there is a necessity to exchange contexts among applications, the middleware tier, and the network layer component to build an optimized data dissemination structure. Thus, content-based subscriptions are used when ODMRP builds the routing table to disseminate the events. Note that it is necessary to develop data structures and algorithms that can introduce efficient subscription propagation and event matching/filtering. Naturally XML is a good candidate, although it lacks typing. Research is ongoing on subsumption of XML types [10]. Thus, events are defined in XML format with XML schema, and XPath is used as a subscription language. The subscriptions are tightly linked to the corresponding event data structure in this system, and summarized subscriptions are created using Bloom filters. Furthermore, Bloom filters are used to digest publishing events and event advertisements. An event source broker node determines the subscription belonging to the group by the propagated subscriptions. Brokers can propagate any changes in subscriptions to ODMRP's periodic membership refreshing mechanism.

3.1. XML-based Typed Event (Message)

The event is defined in XML format with XML schema, and the root element name identifies the event type. The XML schema for the event consists of a set of typed elements. Each element contains a type and a name. The element's name is a simple string, while the value can be in any range defined by the corresponding type. Fig. 3 shows the schema example named 'CD' and Fig. 4 shows example messages. Events themselves can be carried in byte stream or compressed format in ODMRP packets.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema attributeFormDefault="unqualified"
    elementFormDefault="qualified"
    xmlns="http://www.cl.cam.ac.uk/~ey204/lib/"
    xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:simpleType name="categoryType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="jazz"/>
            <xs:enumeration value="classic"/>
            <xs:enumeration value="pop"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="priceType">
        <xs:restriction base="xs:float">
            <xs:maxInclusive value="25.00"/>
            <xs:minInclusive value="10.00"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="category" type="categoryType"/>
    <xs:element name="composer" type="xs:string"/>
    <xs:element name="price" type="priceType"/>
    <xs:attribute name="id" type="xs:int"/>
    <xs:attribute name="timestamp" type="xs:date"/>
    <xs:element name="CD">
        <xs:complexType>
            <xs:sequence>
                <xs:element ref="category"/>
                <xs:element ref="composer"/>
                <xs:element ref="price"/>
            </xs:sequence>
            <xs:attribute ref="id"/>
            <xs:attribute ref="timestamp"/>
        </xs:complexType>
    </xs:element>
</xs:schema>
```

Figure 3. XML Schema Definition for Type CD

```
<?xml version="1.0" encoding="UTF-8"?>
<CD id="001" timestamp="1999-02-27T12:00:00.000-00:00"
       xmlns="http://www.cl.cam.ac.uk/~ey204/lib/">
    <category>jazz</category>
    <composer>Evans</composer>
    <price>18.00</price>
</CD>
----------
<CD id="002" timestamp="1003-02-27T12:00:00.000-00:00"..>
    <category>jazz</category>
    <composer>Davis</composer>
    <price>10.00</price>
</CD>
----------
<CD id="003" timestamp="1055-02-27T12:00:00.000-00:00"..>
    <category>classic</category>
    <composer>Bach</composer>
    <price>22.00</price>
</CD>
```

Figure 4. Event Instance for Type CD

3.2. Subscription Language

Most event systems support a subscription language that allows a subscriber to express its information need. The resulting filtering expression is then used by the brokers to determine whether a particular event notification is of interest to a subscriber. If the language is very expressive, efficient matching of notifications will be difficult. However, if the language does not support rich constructs, its applicability is limited. A subset of XPath [1] is used as a filter-specification language (see examples in Fig. 5). Some XPath expressions will be transformed to the simplified and unified formats, and complex expressions are limited.

```
1. /CD[category="jazz" and price<"20" and price>"15"]
2. /CD[category="pop"][composer="Harrison"]
3. /CD[category="jazz" and price<"20" and price>"10"]
4. /CD[category!="jazz"] will be transformed to
             /CD[category=“pop” and “classic”]
5. /CD[composer="Davis"]|/CD[category="pop"]
    will be transformed to two subscriptions
```

Figure 5. Subscription Filter for Type CD

3.3. Summary Structure in Bloom Filters

Given the constrained MANET environments, it is mandatory to aggregate the set of subscriptions to a compact set of content specifications. The proposed subscription summary structure in Bloom filters is based on the one described in [18], extended to take advantage of XML typed events by XML schema and XPath subscription language. A subscription summary from XPath consists of four data structures described below, and a broker's subscription summary is an array of these data to keep the summarized per-broker subscription information. As an option, a broker aggregates subscription information, when it serves many subscribers or the broker acts on the second level of source node.

Event Type Name (ETN): is the hashed value of the root element name in the XML schema that is an identifier of the event type.

Element Association List (EAL): EALs store information about the elements and attributes in XML schema and actual values that belong together in a subscription. An EAL consists of an array of bits with a constant number of columns and a variable number of rows for subscriptions. Columns represent the ordered set of supported elements and attributes defined in XML schema, and the rows represent the unique sets of subscriptions. Redundant EALs are eliminated. Each row in an EAL includes associated AEC and SEC subsequently. An example of the subscriptions 1 to 3 in Fig. 5 is shown in Fig. 6.

Element or Attribute position in the array						
Element or Attribute	CD	id	timestamp	category	composer	price
Position	1	2	3	4	5	6

EAL for Subscriptions (1-3) on a broker						
Position of the array	1	2	3	4	5	6
Subscription 1	1	0	0	1	0	1
Subscription 2	1	0	0	1	1	0
Subscription 3	1	0	0	1	0	1

Figure 6. Element Association List (EAL)

Arithmetic Element Constraint (AEC): AEC holds the constraints of each arithmetic attribute of a subscription. The first array consists of two columns and a variable number of rows. Each row represents non-overlapping sub-ranges of values specified in subscriptions for the specific attribute. The second array is used when an arithmetic constraint has an equality operator for a value that is not in the existing sub-ranges. An example is shown in Fig. 7.

AEC for price		
Range	*min*	*max*
Sub-range	10.00	20.00
Equality		

Figure 7. Arithmetic Element Constraint (AEC)

String Element Constraint (SEC): SEC contains information about the constraints in string type. For each different string type element/attribute, which appears in at least one subscription, a broker implements a SEC structure using three bit vectors (SEC_L, SEC_R, SEC_X), as Bloom filters. For containment operations, the specified string value is divided into two substrings, 'left' and 'right', defined relative to the position of the operator '*'. After the string value is divided into the two substrings, the left (right) substring is hashed and placed in the $SEC_L(SEC_R)$ filter for the specific string. If the constraint specified a prefix or suffix operation, the specified string value is hashed, and the result is inserted in the SEC_X filter. In case of equality without a containment operation, the whole string is considered as if a suffix operation had been performed, and the hashed result is placed in SEC_X.

3.4. Subscription Aggregation over Networks

The broker nodes that keep Bloom filters have the correct bits set for the subscription to be recognized as receiver nodes over the network. The event source broker node defines the multicast group with the propagated subscriptions.

Currently the least constrained subscription is used for multicast group setting. If the group covers a coarse-grain subscription, more noise would be delivered to the broker, while, if it covers a fine-grain subscription, many groups would be created. Although, when a node has high mobility, setting a fine-grain subscription may prevent the overhead of group membership maintenance. Because of the dynamic MANET environment, a crisp and deterministic setting may not produce better performance than the probabilistic approach. It is challenging to define the balance between multicast groups and subscriptions, and the current approach will be refined by evaluating the experiment results. A couple of approaches into solve this problem are a part of a current ongoing project: (i) aggregate subscriptions to hierarchical patterns including the encoding mechanism; (ii) split/combine subscriptions on the group using content graphs.

ODMRP maintains the group membership with periodic flooding (see Section 3.6), and subscriptions are also up-to-date using this mechanism. If there is any change in subscriptions, brokers maintain the group setting, as well as splitting and combining the groups. The group splitting operation is in an experimental stage, and no detail is reported in this paper. In Fig. 8, in total four subscriptions are registered. When an event is published, the multicast group will be set with the least subscription among the four subscriptions and the published event.

```
Subscriptions on a Broker A
1. /CD[category="jazz" and price<"20" and price>"15"]
2. /CD[category="pop"][composer="Harrison"]
3. /CD[category="jazz" and price<"20" and price>"10"]

Subscription on a Broker B
4. /CD[category="jazz"][composer="Evans"]

Published Event
5. /CD[category="jazz"][composer="Evans"][price="12"]

Established Multicast Group
6. /CD[category="jazz"]
```

Figure 8. Subscriptions and Group Setting

3.5. Routing Strategy

A broker can reside in an independent node or in the same node where publishers/subscribers reside. The routing between the broker and the client can be achieved by the original ODMRP. This paper focuses on the routing between the broker publishing an event and the brokers subscribing to it. In general, there are two approaches to disseminate events to the corresponding subscribers. The first is flooding message by broadcast, followed by filtering performed at the broker. The second is match-first that requires a precomputed destination list, which is broadcasted to all brokers, followed by routing using the routing list. Flooding protocol is simple but may lead to network congestion. Match-first is not scalable, because the routing table must be shared by all brokers, and preprocessing may not work well in MANET environments. The third possibility is a distributed-approach by the brokers, where the brokers examine the message content and forward messages using the routing table. This is bandwidth and space efficient, but establishing and maintaining routing tables can be complex.

MANET environment's dynamic network condition may cause frequent reconfiguration of routing table. Therefore, our proposal is an integrated approach of the above third way with extending ODMRP.

3.6. Extended ODMRP

The protocol of extended ODMRP is described below.

Join Request (JR) Operation:

- A sender node wishing to send multicast packets periodically broadcasts a Join Request (JR) over the network (see Fig. 9).
- The summary of the event to be sent in the Bloom filter expression is attached to the JR packet.
- A node receiving a non-duplicate JR stores the upstream node ID and rebroadcasts the packet.
- Optionally, a Join Request operation provides 'advertisement' mode, which sends out a special event to establish the route before sending out the whole data. This can be deployed when the network is stable and pre-setting routing is beneficial.

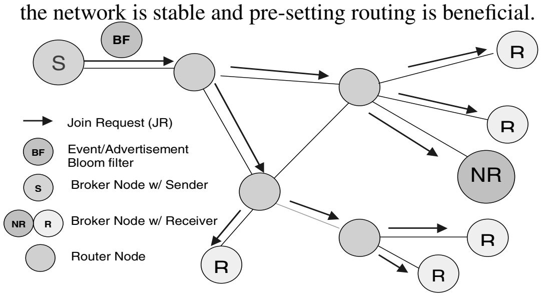
S
BF
R
R
NR
Join Request (JR)
BF Event/Advertisement Bloom filter
S Broker Node w/ Sender
NR R Broker Node w/ Receiver
Router Node
R
R
R

Figure 9. Join Request Operation

JoinTable (JT) Operation:

- A multicast receiver, getting the JR, creates or updates the source entry in its member table (see Fig. 10). During this operation, the Bloom filter in JR is also compared with the local summary of subscription Bloom filters.
- JoinTable (JT) are broadcasted periodically. The subscription information is attached to the JT packet.
- The source node updates subscription information for the group according to the received subscription information in the JT packet.

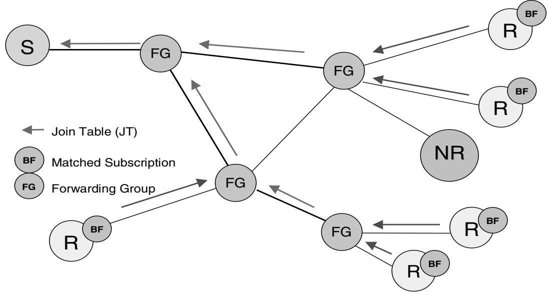

Figure 10. JoinTable Operation

Build Mesh: An intermediate node (router node), receiving the JT, compares its ID with the entries of the forwarding group table. If there is a match, it is a member of the forwarding group (see Fig. 10). It sets the FG_Flag and broadcasts its JT. This process creates a mesh between all forwarding group members.

Data Forwarding: After the group establishment and route construction process, a multicast source can transmit packets to receivers via selected routes and forwarding groups. When receiving a multicast data packet, a node forwards it ONLY IF it is not a duplicate and the setting of the FG_Flag for the multicast group has not expired. This whole operation minimizes the traffic overhead and prevents sending packets through stale routes.

Route and Subscription Maintenance: No explicit control message is required to join or leave the group. When the source node leaves the group, it stops sending a *join request* packet, and the receiver node avoids sending back a *join table* packet to leave the group. Forwarding group nodes demote to non-forwarding group nodes if no *join request* is received within the time out period. For maintaining the group ODMRP requires periodic flooding of *join request* to refresh routes and group memberships. Receivers broadcast *join table* as a reply to the *join request* or subscription change on the node. This flooding of packets often causes the broadcasting storm problem. Optimizing the refresh interval for ODMRP is critical for performance and reliability. The ODMRP scheme can utilize mobility prediction to adapt the route refresh interval. This scheme uses the location and mobility information provided by a system such as GPS (Global Positioning System) to determine how long routes will remain valid. The route refresh interval can then be adapted.

3.7. Prototype Development

A Java-based prototype of the proposed content-based publish/subscribe system is currently under development. The extended ODMRP has been implemented in Java as application level multicast, because a MANET routing multicast protocol is not yet integrated to the network layer in most commercially available mobile devices. A network simulator JNS (Java Network Simulator) [9] is extended to support the extended ODMRP, and mobility simulation is added.

4. Future Work and Conclusions

We introduced a novel publish/subscribe system in mobile ad-hoc networks. In this system, summaries of content-based subscriptions with Bloom filters are used to determine the subscribing broker nodes. An extended ODMRP utilizes summarized subscriptions and constructs an optimized event dissemination mesh. We will continue our work in the following aspects: (i) building hierarchical subscription summaries with Bloom filters including encoding of hierarchy implication; (ii) establishing an efficient event matching algorithm using approximate and fuzzy approach in order to represent content-based subscription in a graphes or set; (iii) defining subtyping mechanism of XML schema.

Performance evaluation within the network simulator is in progress. Evaluation requires many different aspects such as overhead of extended part of ODMRP, complexity of subscription, efficiency of global subscription aggregation, reliability of different mobility patterns, compared with original ODMRP, multicast group size, network traffic load, number of subscribers, false positive rate by Bloom filters etc. We will study the experimental results and combine the above research work to improve the publish/subscribe system in unstably connected network environments.

Acknowledgment. We thank Jon Crowcroft (University of Cambridge) for valuable comments.

References

[1] Berglund, A. et al. XML Path Language (XPath) 2.0. Working Draft. In *W3C, http://www.w3c.org/TR/xpath20/*, 2001.

[2] Bloom, B. Space/time Trade-offs in Hash Coding with Allowable Errors. In *CACM, 13(7)*, 1970.

[3] Carzaniga, A. et al. Achieving scalability and expressiveness in an Internet-scale event notification service. In *Proc. of 19th ACM Symposium on Principles of Distributed Computing*, 2000.

[4] Chiang, C. et al. On-Demand Multicast Routing Protocol. In *Proc. of IEEE WCNC*, 1999.

[5] Corson, S. et al. Mobile ad hoc networking(MANET): Routing protocol performance issues and evaluation considerations. In *RFC 2501*, 1999.

[6] Cugola, G. et al. Exploiting an event-based infrastructure to develop complex distributed systems. In *Int. Conf. on Software Engineering*, 1998.

[7] Eugster, P.Th. et al. The Many Faces of Publish/Subscribe. *Technical Report TR-DSC-2001-04, EPFL*, 2001.

[8] IBM. Gryphon: Publish/Subscribe over Public Networks. In *http://researchweb.watson.ibm.com/gryphon/gryphon.html*.

[9] Java Network Simulator (JNS). In *http://http://jns.sourceforge.net*.

[10] Kyper, G. et al. Subsumption for XML Types. In *Proceeding of ICDT*, 2001.

[11] Lee, S. et al. A Performance Comparison Study of Ad Hoc Wireless Multicast Protocols. In *IEEE INFOCOM*, 2000.

[12] Muhl, G. et al. Filter Similarities in Content-Based Publish/Subscribe Systems. In *Proc. of ARCS*, 2002.

[13] The Narada Event Brokering System . *http://grids.ucs.narada.edu/ptliupages/projects/narada*.

[14] Pietzuch, P. et al. Hermes: A distributed event-based middleware architecture. *Proc. of DEBS*, 2002.

[15] Ratnasamy, S. et al. Application-Level Multicast using Content-Addressable Networks. In *Proc. of NGC*, 2001.

[16] Rowstron, A. et al. SCRIBE: The design of a large-scale event notification infrastructure. In *Int. Workshop of NGC*, 2001.

[17] Snoeren, A. et al. Mesh Based Content Routing using XML. *Proc. of SOSP*, 2001.

[18] Triantafillou, Y. et al. Subsciption Summaries for Scalability and Efficiency in Publish/Subscribe Systems. In *Proc. of DEBS*, 2002.

[19] Zhuang, S. et al. Bayeux: An architecture for scalable and fault-tolerant wide-area data dissemination. In *Proc. of NOSSDAV*, 2001.