Report Number: WUCSE-2007-6

2007

Context-Aware Publish Subscribe in Mobile ad Hoc Networks

Davide Frey and Gruia-Catalin Roman

The publish-subscribe communication paradigm is enjoying increasing popularity thanks to its ability to simplify the development of complex distributed applications. However, existing solutions in the publish-subscribe domain address only part of the challenges associated with the development of applications in dynamic scenarios such as mobile ad hoc networks. Mobile applications must be able to assist users in a variety of situations, responding not only to their inputs but also to the characteristics of the environment in which they operate. In this paper, we address these challenges by extending the publish-subscribe paradigm with the ability to manage and exploit context... Read complete abstract on page 2.

Context-Aware Publish Subscribe in Mobile ad Hoc Networks

Davide Frey and Gruia-Catalin Roman

Complete Abstract:

The publish-subscribe communication paradigm is enjoying increasing popularity thanks to its ability to simplify the development of complex distributed applications. However, existing solutions in the publish-subscribe domain address only part of the challenges associated with the development of applications in dynamic scenarios such as mobile ad hoc networks. Mobile applications must be able to assist users in a variety of situations, responding not only to their inputs but also to the characteristics of the environment in which they operate. In this paper, we address these challenges by extending the publish-subscribe paradigm with the ability to manage and exploit context information when matching events against subscriptions. We present our extension in terms of a formal model of context-aware publish-subscribe. We propose a solution for its implementation in MANETs; and finally we validate our approach by means of extensive simulations.

2007-6

Context-Aware Publish Subscribe in Mobile ad Hoc Networks

Authors: Davide Frey and Gruia-Catalin Roman

Corresponding Author: frey@wustl.edu

Abstract: The publish-subscribe communication paradigm is enjoying increasing popularity thanks to its ability to simplify the development of complex distributed applications. However, existing solutions in the publish-subscribe domain address only part of the challenges associated with the development of applications in dynamic scenarios such as mobile ad hoc networks. Mobile applications must be able to assist users in a variety of situations, responding not only to their inputs but also to the characteristics of the environment in which they operate. In this paper, we address these challenges by extending the publish-subscribe paradigm with the ability to manage and exploit context information when matching events against subscriptions. We present our extension in terms of a formal model of context-aware publish-subscribe. We propose a solution for its implementation in MANETs; and finally we validate our approach by means of extensive simulations.

Type of Report: Other

Context-Aware Publish Subscribe in Mobile ad Hoc Networks

Davide Frey and Gruia-Catalin Roman

Department of Computer Science and Engineering
Washington University in St. Louis
{frey,roman}@cse.wustl.edu

Abstract. The publish-subscribe communication paradigm is enjoying increasing popularity thanks to its ability to simplify the development of complex distributed applications. However, existing solutions in the publish-subscribe domain address only part of the challenges associated with the development of applications in dynamic scenarios such as mobile ad hoc networks. Mobile applications must be able to assist users in a variety of situations, responding not only to their inputs but also to the characteristics of the environment in which they operate. In this paper, we address these challenges by extending the publish-subscribe paradigm with the ability to manage and exploit context information when matching events against subscriptions. We present our extension in terms of a formal model of context-aware publish-subscribe. We propose a solution for its implementation in MANETs; and finally we validate our approach by means of extensive simulations.

1 Introduction

Publish-Subscribe has emerged as a communication paradigm able to facilitate the development of complex distributed applications in open network environments. The strong decoupling it introduces between communication parties enables applications to *publish* information without being aware of the identities of potential receivers or even of their existence. Similarly, it enables receivers to issue *subscriptions* that express their interests in messages with a given content regardless of the identity of their publishers. These characteristics make the paradigm particularly suited to scenarios where the set of communicating parties is subject to frequent changes as in Mobile Ad Hoc Networks (MANETs).

Consider the problem of disseminating traffic information in a network of vehicles. Messages need to be routed to vehicles regardless of their identity and based on the interests expressed by their drivers. A vehicle moving on a freeway might, for example, be interested in messages announcing traffic accidents or slowdowns between its current location and its intended exit. A vehicle approaching its destination might want to be notified about the availability of new parking spots in its vicinity. Similarly, a vehicle running out of fuel will request information about available gas stations. The publish-subscribe paradigm is naturally suited to this type of scenarios. Vehicles witnessing an accident, leaving their parking spots, or observing other relevant events can *publish* this information without having to know if there are any vehicles interested in receiving it.

The appropriateness of the publish-subscribe paradigm for mobile scenarios has motivated researchers to investigate routing techniques for the dissemination of events and subscriptions over dynamic network topologies. However, these techniques solve only part of the issues related to programming applications in mobile networks. Mobile applications are required to assist users in a variety of situations, responding not only to their inputs but also to the characteristics of the environment in which they operate. In the above example, a vehicle needs to be notified of available parking spots only when it approaches its destination and not while it is still on the freeway. Likewise, the interest in a specific parking spot increases or decreases depending on its location, or the presence of other vehicles also looking for a parking place in its vicinity. Similarly, vehicles that enter a freeway after an accident has occurred should be notified about the accident even if the accident happened some time ago. Traditional publish-subscribe middleware offers only limited support for these aspects: for example events are not generally associated with a notion of persistence and only propagate instantaneously through the network. This forces developers to deal with this and other contextual aspects at the application level preventing the middleware from exploiting context information to optimize the dissemination of events.

In this paper, we address these limitations and develop a new kind of middleware that integrates the publish-subscribe paradigm with the requirements of context-aware mobile applications. Such an undertaking poses significant intellectual and technical challenges. Managing context at the middleware level requires a richer set of primitives than those available in current publish-subscribe implementations. Our middleware extends the publish-subscribe API and enriches events and subscriptions with notions of space, time, and more generally with the context information associated with publishers and subscribers. Publishers can thus constrain the diffusion of events by specifying that each event is relevant and/or visible only in a given context. In addition, they can exploit the time dimension and define persistent events that should remain available for a specified time after their publication. Similarly, subscribers can subscribe to events that are relevant in specified context domains and originate at publishers belonging to a particular context.

The paper is structured as follows. Section 2 presents our model of context-aware publish-subscribe. Section 3 proposes our context-aware routing protocol for its implementation in MANETs. Section 4 evaluates its performance by means of simulation. Section 5 places our work in the context of related efforts, and Section 6 concludes the paper.

2 Bringing Context into Publish-Subscribe

We introduce our extension to the publish-subscribe paradigm in the form of a basic formalization. This allows us to present our notion of context and discuss how it may be integrated into the publish-subscribe communication paradigm.

2.1 Basic Definitions

We assume a universe consisting of hosts that move freely through the physical space $\mathcal{S}$. Each host is characterized by a unique identifier $h \in H$, a vector of attribute-value pairs $a \in A$, and its location in space, $\lambda \in \mathcal{S}$. For the time being, we ignore the aspects related to communication between hosts and return to them in Section 3 in the description of our routing and matching protocol.

System Configuration. At every instant in time, we can describe the configuration of the system in terms of the distribution of the hosts in the physical space and of values assigned to the attribute vectors associated with each host. We model this information by defining a *system configuration* as a pair consisting of two functions: the *spatial distribution* and the *system state.* The former expresses the geographical aspects of the system configuration by associating each host with its current geographical location. The latter captures its non-geographical aspects and associates each host with its current vector of attribute-value pairs.

In mathematical terms, we represent the spatial distribution and the system state, respectively, as two functions $f_s \in \mathcal{C}_s$ and $f_c \in \mathcal{C}_c$, where

$$\mathcal{C}_s = (H \to \mathcal{S}) \text{ and } \mathcal{C}_c = (H \to A).$$

Based on these definitions we model the system configuration as a pair $c = (f_s, f_c)$ where

$$c \in \mathcal{C} = \mathcal{C}_s \times \mathcal{C}_c.$$

Configuration Function. Both the geographical and non-geographical aspects of a system's configuration continuously vary through time due to the movement of hosts and due to changes in the hosts' attribute values. To model this dynamic aspect, we consider a linear notion of time with values from the set T. This allows us to capture the history of the system's evolution over time by means of a *configuration function* that associates each time instant $t \in T$ with the corresponding system configuration.

$$\mathcal{K} : T \to \mathcal{C}$$

2.2 Context Specifications

Informally, we can define context to include those aspects of the state of the environment that can affect a particular entity, henceforth called the *reference host.* In an environment like a mobile ad hoc network, it is natural to define context as *a set of hosts and their associated properties that are of interest to a given reference host, i.e., hosts that can affect its behavior, can communicate with it, or can carry out activities on its behalf.* This notion of context is reasonable in a MANET because, in the absence of any fixed infrastructure or dedicated servers, all information must be associated with one or more mobile hosts.

The hosts associated with a particular context can rarely be specified by explicit enumeration because it is often impossible to know in advance which hosts may be of interest to a given reference host, or even which hosts are in the system at a particular point in time. Thus, we introduce the notion of *context specification*. Context specifications allow a reference host to identify the hosts that are part of some context in terms of the properties they must have as individuals or as a group. *Individual properties* are defined on a host's location and attribute values. The properties of a group, henceforth called *relational properties*, relate the location and attributes of multiple hosts in the same system configuration. An individual property, for example, may express the fact that a host is "on the freeway and is traveling faster than 55mph". A relational property may instead identify the hosts that are "traveling faster than the hosts around them." The context defined by a given context specification is inherently dynamic. The set of hosts that are part of some the context may vary as a result of changes in their locations, in their attribute values, and in those of other hosts in the system configuration. We model a context specification as a function that selects a group of hosts of interest from a given configuration. In this paper, we focus on context specifications with static properties that identify dynamic sets of hosts. However, in the most general case, the properties may also change over time; thus, the function, denoted by α, assumes as arguments a time instant and a system configuration, i.e.,

$$\alpha(t, \mathcal{K}(t)) \in \Lambda = (T \times \mathcal{C} \rightarrow \mathcal{P}(H))$$

where the symbol $\mathcal{P}(X)$ denotes the powerset of set X.

2.3 Bringing Context into Events and Subscriptions

Traditionally, publish-subscribe communication is achieved by means of event notification messages that propagate through the network to reach all matching subscribers. Publishers define the content of each event at publication time, while subscribers define subscription filters that operate on this content. The middleware determines which subscribers should receive a given event by means of a matching process that applies the filters associated with existing subscriptions to the content of every newly published event. Given a universe E of possible events, we model a specific event as a member of this set (i.e., $e \in E$) and a subscription as the set of events (i.e., $\tilde{e} \in \mathcal{P}(E)$) of interest to the subscriber. Formally, this reduces matching to a simple membership test, $e \in \tilde{e}$.

As we pointed out in Section 1, the publish-subscribe communication model exhibits several limitations that hamper its applicability in context-aware applications for mobile network scenarios. In the remainder of this section, we remove these limitations and add several new features that allow publish-subscribe middleware to respond not only to the interests of subscribers as they change over time, but also to the evolving context in which subscriptions and events exist.

- We unify the treatment of events and subscriptions by allowing both of them to remain available in the system for an arbitrary time until their *expiration*.
- We allow events and subscriptions to exist within a limited scope, which we call *context of relevance* in the case of events and *context of interest* in the case of subscriptions.

- We enable publishers and subscribers to use context specifications to restrict the identity of their communication parties based on their current context by defining a *publication* and a *subscription domains*.

Persistent Events and Subscriptions. The first characteristic we introduce in our context-aware publish-subscribe model is the ability to define events that may remain available in the system for an arbitrary time after their publication. To achieve this, we have each publisher associate an expiration time $\tau_e \geq 0$ with each of its event notifications. This implies that, at each time instant τ, an event is active and may be matched by a subscription only if its expiration time τ_e has not elapsed.

The notion of expiration time allows us to distinguish between *instantaneous* and *persistent* events. Instantaneous events correspond to those available in traditional publish-subscribe middleware and are characterized by an expiration time that coincides with their publication time. Persistent events, on the other hand, are those for which expiration occurs later than the time of publication. Persistent events enrich the paradigm with the ability to maintain state in event notifications by extending their relevance over an arbitrarily long period of time. This allows the middleware to address situations like the freeway scenario mentioned above, in which the event notifying vehicles of the accident should remain available after the accident has occurred.

Following the same pattern as with event notifications, we also introduce an expiration time $\tau_s \in T$ in the case of subscriptions. According to intuition, a subscription is active and available for matching at a given time instant if its expiration time has not elapsed.

Contextual Relevance and Interest. The second characteristic we introduce in our model is the ability for publishers to associate their events with information about the context that may be affected by them. Consider the example in Figure 1a: an accident happens on the freeway at mile 32 as indicated in the figure. One of the cars involved in the accident publishes a persistent event to inform oncoming vehicles of the danger. Ideally, the event should propagate towards vehicles that are approaching the accident site and not to those that are already past the accident or traveling in the opposite direction. To make this possible, the publisher associates the event notification with a *context of relevance*. The context of relevance represents the context that, according to the publisher, will be affected by the event or in which the event will be relevant: in this case the vehicles that are traveling east between miles 20 and 32. The set of hosts that constitute the context of relevance is dynamic and thus cannot be computed once for all by the publisher and encoded in the content of the event. The context of relevance may, in fact, vary over time due the mobility of hosts or due to changes in the values of their attributes. We address these dynamic aspects by modeling the context of relevance as a context specification $r \in \Lambda$. The context of relevance expresses the opinion of the publisher regarding the context that may be affected by the event or in which the event should be considered relevant. This means that subscribers outside an event's context of relevance may still request that the event be delivered to them.

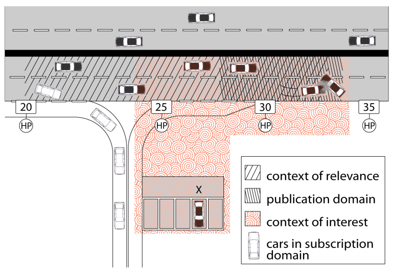
Consider again the above scenario. A short while after the accident, car X leaves its parking lot and intends to travel along the freeway where the accident happened. The middleware, should enable car X to issue a subscription that matches the events that may affect the road it intends to take. To make this possible, we allow subscribers to associate a *context of interest* with their subscriptions. Similar to the context of relevance, the context of interest is described by a context specification $\tilde{r} \in \Lambda$ and identifies a set of hosts that may change over time due. An event's context of relevance matches a subscription's context of interest if the sets of hosts they identify have a non-empty intersection. In Figure 1a, the context of interest of car X's subscription consists of the hosts in the area identified by the dashed contour. This allows subscriber X to receive the information about the accident from the hosts in the dashed area and decide for a different route.

Constraining the sets of publishers and subscribers. The notions of relevance and of interest allow publishers and subscribers to associate a contextual scope to events and subscriptions. However, they do not allow them to restrict the identity of their communication parties based on their current contexts. Consider again the above scenario: the highway patrol has several stations along the freeway, each responsible for accidents that happen in a specific section. To detect accidents, each station subscribes to events regarding the freeway. However, it must also be able to specify that the publishers of these events should be located in the section it is responsible for. In this case the information about the accident should be received and handled by the station at mile 30 and not by the one at mile 25, regardless of the event's domain of relevance. To make this possible, each station associates its subscription with a *publication domain*: a context specification that, for each time instant and configuration, identifies the publishers whose events may be matched by a given subscription. In the above example, the station at mile 30 specifies that it wants to receive events published by cars involved in accidents between miles 28 and 33.

In a similar manner, publishers may want to restrict the identity of possibly matching subscribers based on their own context information. After intervening on the scene, the highway patrol attempts to speed up traffic by forcing vehicles with less than two occupants to exit at the last available exit before the accident. To inform vehicles of this decision it publishes a persistent event, stating that it should reach only the cars with only one occupant. To make this possible, the highway patrol associates a further context specification with its event: the *subscription domain.* This context specification states that the event may be received by subscribers that have a matching subscription filter, only if they are in the specified context (in this case, if they have only one occupant).

As with all other context specifications, we model the publication and subscription domains as two functions: respectively $\tilde{p} \in \Lambda$ and $\tilde{s} \in \Lambda$ that associate each time instant and configuration with a set of publishers or subscribers.

Events, Subscriptions, and Matching. The extensions we just defined allow us to enrich the publish-subscribe model with the notions of context-aware events, subscriptions, and matching. We define a context-aware *event* — henceforth



(a) Accident Scenario

Event:

$p \in H$	publisher's identifier
$e \in E$	content of the event
$\tau_e \in T$	expiration of event
$r \in \Lambda$	context of relevance
$\tilde{s} \in \Lambda$	subscription domain

Subscription:

$s \in H$	subscriber's identifier
$\tilde{e} \in \mathcal{P}(E)$	filter on content of events
$\tau_s \in T$	expiration of subscription
$\tilde{r} \in \Lambda$	context of interest
$\tilde{p} \in \Lambda$	publication domain

(b) Event and Subscription Format

Fig. 1: Application Scenario and Event-Subscription Model

called event — as a tuple $[p, e, \tau_e, r, \tilde{s}]$. As summarized in Figure 1b, the tuple includes the identity of the publisher, the content of the event, its expiration time, the context of relevance and the subscription domain. In a similar manner, we define a context-aware *subscription* — henceforth called subscription — as a tuple $[s, \tilde{e}, \tau_s, \tilde{r}, \tilde{p}]$. As summarized in Figure 1b, this tuple consists of the identity of the subscriber, the subscription filter, the expiration time of the subscription, its context of interest, and its publication domain.

The definitions of context-aware events and subscriptions allow us to model the concept of *context-aware matching* as a direct extension of the matching operation for traditional publish-subscribe. As in the traditional case, the goal of matching is to determine whether a given event should be delivered to a given subscriber. However, our context-aware matching process must also take into account the current configuration of the system. Let us assume that each context specification in a given event-subscription pair is evaluated at time instant τ, and let $c = \mathcal{K}(\tau)$ denote the system configuration at that instant. Then we can state that a given event matches a given subscription if and only if the following five conditions are satisfied.

– (i) *standard matching*: the event content matches the subscription filter as in standard content-based publish-subscribe.
$$e \in \tilde{e}$$

– (ii) *lifetime validity*: the event and the subscription have not expired.
$$\tau \leq \tau_e \wedge \tau \leq \tau_s$$

– (iii) *interest-and-relevance overlap*: the context of relevance of the event and the context of interest of the subscription intersect each other.
$$r(\tau, c) \cap \tilde{r}(\tau, c) \neq \emptyset$$

– (iv) *publication domain matching*: the publisher is part of the context identified by the subscriber's publication domain.
$$p \in \tilde{p}(\tau, c)$$

– (v) *subscription domain matching*: the subscriber is part of the context identified by the event's subscription domain.
$$s \in \tilde{s}(\tau, c)$$

3 Routing and Matching in Context-Aware Publish-Subscribe

The definition of matching presented in Section 2.3 forms the basis for our routing and matching protocol supporting the context-aware publish-subscribe paradigm. The protocol is based on the idea that persistent events and subscriptions should be maintained by the hosts in the contexts of relevance and of interest until their expiration time. For a given event-subscription pair, the matching process is initiated by the hosts in the intersection of these two contexts, by evaluating conditions (i), (ii), and (iii): standard content-based matching, event and subscription lifetime, and the overlap between the contexts of relevance and of interest. If this evaluation is successful, further processing depends on whether the subscription includes a the publication domain, i.e., evaluation of condition (iv). If this is the case, the host that is carrying out the matching must contact the publisher. Otherwise it simply forwards the event to the subscriber for the evaluation of the subscription domain, i.e., condition (v), and for possible delivery to the application.

3.1 Assumptions and Requirements

To manage the complexity of context-aware matching, we built our protocol over a geocast routing service [17] and we assume that the contexts of relevance and of interest are always associated with some geographical component that identifies a region of the physical space. Moreover, we constrain the relational properties associated with the contexts of relevance and of interest to operate on groups of hosts within a one-hop neighborhood and on an application-defined subset of attributes, henceforth called *key attributes*. Hosts exchange periodic beacons that contain information about their current location, their location at the time their previous beacon was sent, and the current values of their *key attributes*. This allows relational properties to be evaluated locally by the hosts in the contexts of relevance and of interest, without issuing a query for each matching operation. In the case of the publication and subscription domains, on the other hand, relational properties may also exploit non-key attributes and may refer to groups of hosts beyond a one-hop range. To achieve this, we assume the use of a query dissemination service like the one in [22].

Finally, it is worth noting, that all the context specifications in events and subscriptions refer to the system configuration when matching occurs. If publishers or subscribers want matching to operate on their state at publish or subscribe time, they can store this information as part of their events or subscriptions at the time they are issued.

3.2 Main Protocol Operation

The protocol exploits four data structures maintained by each network host (h) for the management of events and subscriptions.

- *Event table*: stores the unexpired events that have a geographical component of the context of relevance that includes host h.

- *Subscription table*: stores the unexpired subscriptions that have a geographical component of the context of interest that includes host h.
- *Local event table*: stores the unexpired events issued by host h.
- *Local subscription table*: stores the unexpired subscriptions issued by host h.

In addition, each host maintains a neighbor table that records the location and key attributes of neighboring hosts. This information, gathered with periodic beacons, is used in the evaluation of relational properties. Finally, the geocast protocol records the identifiers of recently received messages to prevent the transmission of duplicates.

Matching Subscription Interest against Event Relevance. We begin our description by considering the publication of an event. The publisher first stores the event in its local event table. Then, it uses the geocast service to forward the event to the geographical region associated with its context of relevance. Each host in this region reacts to the receipt of the event with the following steps. First, if the event is persistent, the host stores it in its event table. This is done even if the non-geographical properties of its context of relevance are not currently satisfied. The dynamic nature of attribute values can in fact cause these properties to be satisfied at a later time. Second, the host carries out the first three steps of the context-aware matching process. First, it verifies whether the current values of its own attributes satisfy the individual properties associated with the context of relevance, and whether the key attributes of the hosts in its neighbor table satisfy the corresponding relational properties. If this is the case, it attempts to match the event against each unexpired subscription in its subscription table. This involves evaluating the subscription filter of each subscription against the content of the event and evaluating if the attributes of the host and the key attributes of the hosts in its communication range satisfy the properties associated with the subscription's context of interest. If these first steps of the matching process are successful, the host takes action to deliver the event to the subscriber. In the accident scenario of Figure 1a, the hosts in the highlighted region will forward the event about the accident to the subscribers that expressed an interest in traffic information regarding the stretch of highway affected by the accident.

Delivering Events To Subscribers. For each event and subscription, the above matching process occurs at the intersection of the geographical regions associated with the contexts of relevance and of interest. Let us refer to this intersection as the *matching area*. A host that detects a positive match between an event and a subscription, prepares a *matched-event* message that encodes both the event and the information on how to reach the subscriber as shown in Figure 2 and discussed in greater detail in the following. Different matched event messages generated by different hosts in the matching area for the same event-subscription pair are indistinguishable to the geocast protocol, which can therefore reduce the number of redundant messages by dropping multiple copies and by passively listening for other node's transmissions before forwarding a packet.

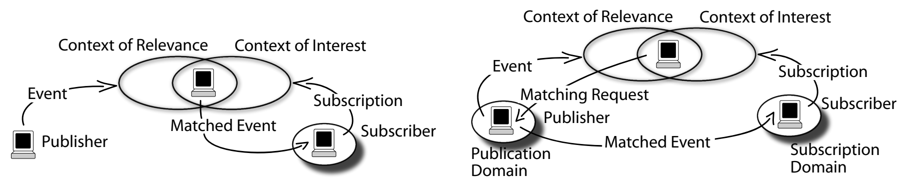
Matching Subscriptions against Persistent Events. Subscriptions are disseminated with a similar mechanism as events. A subscriber that issues a new subscription first stores it in its local table; then it forwards it to the hosts in the region associated with its context of interest. When a host in the region receives the subscription, it first stores it in its subscription table, and then it checks if its attributes and the key attributes of the hosts in its neighbor table satisfy the properties specified by the subscription's context of interest. If this is the case, it attempts to match the subscription against the unexpired persistent events contained in its event table. As in the above case, this involves evaluating whether, for each event, the conditions specified by the context of relevance are satisfied, and whether the subscription filter is satisfied by the content of the event. If these matching steps are successful, the event is forwarded to the subscriber as described above.

Managing the publication and subscription domains. The protocol we just described exploits the contexts of relevance and of interest to direct events towards matching subscribers. In the following, we show how it can also evaluate the subscription and publication domains of events and subscriptions.

Subscription Domain. Let us consider an event with a subscription domain that is matched against a subscription without a publication domain. The subscription domain is evaluated by the subscriber upon receipt of the matched-event message. The subscriber uses its current location, its attribute values, and those in its neighbor table to evaluate individual properties and the relational properties referring to hosts in its one-hop neighborhood. If this first evaluation is successful, it issues a query to retrieve information about the attribute values of hosts beyond the one-hop range. If the results of the query satisfy the relational properties of the subscription domain, the event is delivered to the application. Figure 2a shows the messages exchanged by the protocol in this situation.

Publication Domain. The protocol is slightly more complex when the publication domain is specified. A host that completes the first three steps of the matching process must in fact communicate with the publisher to retrieve information about the current configuration. To achieve this, it sends a *matching-request* message to the publisher. The message contains the identifier of the event being matched, the specification of the publication domain, and information on how to reach the subscriber. The publisher reacts to the matching-request by evaluating the publication domain and, in case of a positive match, by sending a matched-event message to the subscriber using the reachability region specified by the matching request. A schematic view of this behavior is shown in Figure 2b.

Managing Host Mobility. So far, we have described our protocol by ignoring a key aspect of the system, host mobility. In the following we return to this issue and describe our mechanism to handle changes in host location. There are two aspects of our protocol that are affected by the ability of hosts to move: the ability to reach publishers and subscribers with matching-requests and matched-event messages during the matching process, and the maintenance of events and subscriptions in the regions of relevance and of interest.



(a) (b)

Fig. 2: Routing without (a) or with (b) a publication domain.

Mobility of Publishers and Subscribers. To address the first issue, publishers and subscribers label their events and subscriptions with information about how they can be reached. Specifically they associate each event or subscription with a *reachability region*, that is a geographical region computed on the basis of their current location and predicted movement. When routing a matching-request or matched-event message to a publisher or subscriber, a hosts uses the geocast service to deliver it to all the hosts in the reachability region. The description of the reachability region of the subscriber is also included in each matching-request sent to the publisher of a possibly matching event.

In principle, the reachability region should cover all the possible locations of a host throughout the lifetime of a persistent event or subscription. This, however, would result in exceedingly large regions for events and subscriptions with non-trivial lifetimes. Therefore, publishers and subscribers are allowed to update their events and subscriptions with new reachability regions. Updates are sent both periodically and whenever a publisher or subscriber gets within a specified *safe-distance* from the boundary of its reachability region.

Mobility in the Regions of Relevance and of Interest. To maintain events and subscriptions in dynamic regions of relevance and of interest, we combine the above updates to events and subscriptions with a reactive approach. Specifically the hosts in the regions of relevance and of interest monitor their neighbors to determine if they have just entered the region associated with one of their events or subscriptions. To achieve this, nodes include, in each of their beacons, their current location and the one at the time they sent the previous beacon. Using the information about the previous and the new position, a host computes which neighbors have just entered one of the regions associated with the events and subscriptions in its tables and broadcasts a *digest* message containing information about them. Finally, to improve reliability, publishers and subscribers may send each event and subscription to a region that is slightly larger than the actual region of relevance or of interest.

4 Simulation

We evaluated the performance of our protocol by means of a detailed simulation study based on OmNet++ [28] a popular open-source simulation system. We model the physical and MAC layers using Omnet++'s mobility framework.

Specifically, we adopted an 802.11 MAC over which we placed a custom implementation of a receiver-based geocast protocol that provides the ability to route messages to a region of the physical space.

4.1 Simulation Setup

We consider a reference scenario consisting of 100 hosts in a $1000m \times 1000m$ area. The movement of hosts follows the random way-point mobility model, with a minimum speed of 1m/s, a pause time of 0s, and a maximum speed of 10m/s. For each measurement we executed 10 simulation runs of 450s each.

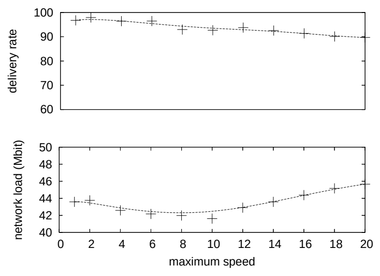
We model contextual attributes by associating each host with a randomly chosen integer value. According to the discussion in Section 3.1, the contexts of relevance and of interest are always associated with a geographical region, represented by a $100m \times 100m$ rectangle. For the publication and subscription domains we consider the worst case scenario by using only non-geographical context specifications. Specifically, each is configured to match 50% of the available attribute values.

Publishers and subscribers each constitute 20% of the hosts in the network. Each subscriber issues a subscription every 50s: the subscription is either instantaneous or has a 50s lifetime. Similarly each publisher generates an event every 10s that is either instantaneous or has a lifetime of 10s. Moreover, 50% of the subscriptions are associated with a publication domain. The size of reachability regions of publishers and subscribers is set to $50m \times 50m$. Events and subscriptions are refreshed every 5s or whenever the publisher or subscribers gets within 10m of the region's boundary. In the absence of other traffic, hosts exchange beacon messages every 5s. Finally, to improve reliability, event and subscription updates are sent to regions that are $60m$ larger than their actual destinations.

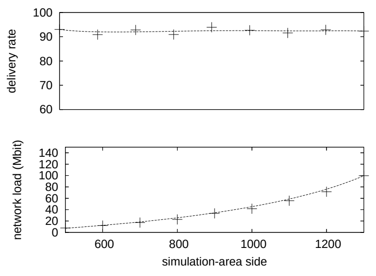
4.2 Protocol Performance

We consider two performance metrics in the evaluation of our protocol: delivery rate and communication cost. To evaluate the former, we compare our protocol against an ideal protocol that instantaneously matches events against subscriptions based on information about the entire system configuration. The delivery rate is the ratio between the number of events successfully delivered by our protocol and those that would be delivered by such an ideal system. The communication cost, on the other hand, is measured as the number of Mbits transmitted at the physical layer. To evaluate the impact of the application scenario on these two metrics, we individually vary four of the above parameters: the speed of hosts, the size of the network, its density, and the percentage of subscriptions that require the evaluation of the publication domain.

Impact of Host Speed. We first consider the performance of our protocol with respect to variable host speed. The plots in Figure 3a show the results obtained with hosts that move with a minimum speed of 1m/s and a maximum speed ranging from 1m/s to 20m/s. The top plot confirms the expectation that the delivery rate should decrease with increasing speed. This is due to the combination



(a) host speed



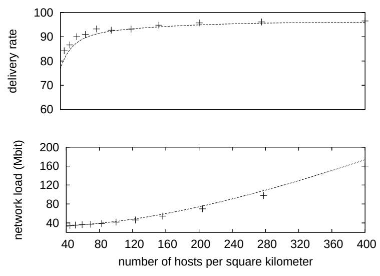
(b) system scale

Fig. 3: Impact of host speed and system scale over delivery rate and communication cost.

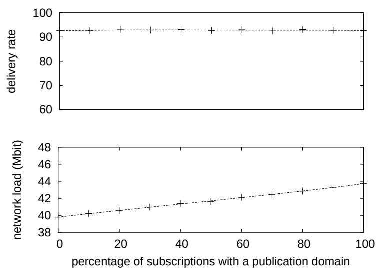
of two phenomena. First the accuracy of the geocast protocol naturally decreases with speed. Second, the higher the speed, the more likely hosts are to move out of the regions of relevance and of interest. In general, a host that enters either region will receive the corresponding event or subscription in a digest message from one of the hosts already in the region. However, if there is no other host in range that can communicate this information, the new host will be unable to retrieve the event or subscription until the next refresh message.

The bottom plot in Figure 3a shows the effect of speed on the communication cost of the protocol. The decrease in cost when the maximum speed increases from 1m/s to 10m/s is a result of the decrease in delivery rate associated with high mobility. When speed increases even further, such a decrease is balanced by an increase in control traffic: digest messages, and refresh messages for event and subscriptions. It is worth noting that the increase remains limited (42Mbit to 46Mbit) even with at the maximum speed of 20m/s. In particular, we verified that flooding events in the same scenario would result in a cost ranging from 83Mbit to 94Mbit without the ability to manage the contextual aspects of events and subscriptions.

Impact of System Scale. Next, we evaluate the scalability of our protocol by varying the size of the network with a constant host density of 100 hosts per square kilometer. Results are depicted in Figure 3b. The top plot shows that the delivery rate of our protocol is largely independent of system scale. An increased scale at a constant host density simply results in a larger number of hops to reach the hosts in the matching area and to deliver matching events to subscribers. Our geocast-based protocol is able to address this increase without significantly decreasing its ability to deliver events. Moreover, the bottom plot in Figure 3b shows that network traffic increases as the cube of the side of the simulation area. This can be easily explained by observing that the number of events and subscriptions in the system grows linearly with the number of publishers and subscribers and thus quadratically with the simulation area. This, together with the fact that each event and subscription must travel a path that is approximately linear with the number of hosts, yields the cubic relationship



(a) host density



(b) publication domain

Fig. 4: Impact of host density and of the evaluation of the publication domain on delivery rate and communication cost.

shown in Figure 3b. In real-world applications, scalability should be further improved since the density of events and subscription is likely to be lower.

Impact of Host Density. Our next experiment analyzes the behavior of our protocol with different host densities. We vary density by changing the simulation area while employing a fixed number of 100 hosts. Specifically, we simulated areas yielding host densities ranging from 44 to 400 hosts per square kilometer. The results are shown in Figure 4a. The top plot highlights the good performance of our protocol in terms of delivery rate for host densities higher than 60 hosts per square kilometer. With lower densities, network connectivity may not be guaranteed at all times and the geocast protocol is more likely to experience routing errors due to local minima. Mechanisms to address this, such as perimeter routing or store-&-forward approaches, may improve performance at low densities. However, their evaluation is outside the scope of this paper.

The bottom plot in Figure 4a shows the behavior of the protocol in terms of network traffic. The increase in host density determines the presence of a larger number of hosts in the regions associated with the contexts of relevance and of interest. This causes an increase in the number of receivers for each event, which translates into the increase in network traffic evidenced by the figure.

Impact of the Publication Domain. The last scenario parameter we consider is the percentage of subscriptions that require the evaluation of the publication domain. As described in Section 3.2, the evaluation of this context specification requires publishers to be contacted after their events have been matched in the intersection of the contexts of relevance and of interest. The data depicted in the bottom plot of Figure 4b confirms the intuition that this process has an impact on the communication cost of the protocol. Matching-request messages account for almost 10% of the overall network traffic when the publication domain must be evaluated for every pair of event and subscription. The top plot in Figure 4b, on the other hand, shows that the delivery rate is almost unaffected by this parameter, despite the longer distance traveled by events before reaching matching subscribers.

5 Related Work

Recent advances in mobile computing infrastructures have led to increasing interest in context-aware applications that can respond not only to user inputs but also to the characteristics of the environment in which they operate. To support these applications, the research community has proposed several middleware solutions that aim to facilitate software development [11, 20, 16, 25].

In particular, researchers have followed the success of publish-subscribe middleware for large-scale wired networks [4, 2, 23, 24, 5, 14, 10, 21, 13, 15, 26, 27]and have investigated solutions for publish-subscribe communication in environments such as MANETs. The solutions in [30, 18, 29, 1, 8, 7] take a first step in the development of protocols that implement the publish-subscribe model in a mobile environment. In this paper, we continue this effort and extend publish-subscribe with the ability to manage and react to context at the middleware level.

Such capabilities are available only in very basic forms in existing middleware. The work in [9] proposes a location-aware extension to the publish-subscribe model in which events and subscriptions can be associated with geographical scopes that resemble our publication and subscription domains. Nonetheless, the management of location information is only one of the needs of context-aware applications. Moreover, while the authors list MANETs as a possible application scenario, they only consider a general-purpose tree-based solution that is not backed up by a performance evaluation. Scalable Timed Events And Mobility (STEAM) [19] ties the locations of publishers with that of subscribers by proposing a proximity-based event model for MANETs. Events are distributed only in a limited geographical area centered around the publisher's location. This results in a limited model that cannot easily represent scenarios like those we consider in this paper. The same limitation affects the work in [12]. While its authors propose a notion of persistent events and subscriptions, their matching model requires the publisher and the subscriber of a matching event-subscription pair to be within a region centered around the other.

In [6], events are associated with a location of occurrence and subscriptions with a geographical predicate. Publishers are therefore unable to control the dissemination of their events. Moreover, the system exploits a centralized spatial matching engine that is clearly unsuitable for the MANET scenarios we target. Finally, Fulcrum [3] tackles the definition of context-aware publish-subscribe from a different angle and presents a system for large-scale context-aware publish-subscribe based on an open-implementation approach. Clients may specify user-defined strategies to implement complex filters that match combinations of events occurring at different locations in the network. Its current implementation, though, relies on a static backbone of brokers which may not always be available in scenarios such as MANETs.

6 Conclusions

Publish-subscribe has emerged as a communication paradigm able to facilitate the development of complex reactive applications in open network environments.

Yet, current systems still offer only partial support for the management of context in mobile applications. In this paper, we proposed an extension to the publish-subscribe paradigm that enables the middleware to factor context information into events and subscriptions. This gives publishers and subscribers the ability to control their diffusion and to restrict the identities of their communication parties. We support this model with a protocol for the dissemination of context-aware events and subscriptions in MANETs. We based this initial protocol on geocast to aid the management of the geographical components of context, but we are also planning to investigate alternative routing approaches. Our simulation analysis shows that our protocol achieves high delivery rates in the presence of mobility while exhibiting good scalability properties. This suggests we can expect even better performance in many real-world scenarios.

References

1. R. Baldoni, R. Beraldi, G. Cugola, M. Migliavacca, and L. Querzoni. Structureless content-based routing in mobile ad hoc networks. In *Proc. of ICPS*, Santorini (Greece), July 2005. IEEE Computer Society Press.
2. G. Banavar, T. Chandra, B. Mukherjee, J. Nagarajarao, R. Strom, and D. Sturman. An efficient multicast protocol for content-based publish-subscribe systems. In *Proc. of the 19th Int. Conf. on Distributed Computing Systems*, Austin, TX, USA, 1999. IEEE Computer Society Press.
3. Robert T. Boyer and William G. Griswold. Fulcrum - an open-implementation approach to internet-scale context-aware publish/subscribe. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05) - Track 9*. IEEE Computer Society, 2005.
4. A. Carzaniga, D.S. Rosenblum, and A.L. Wolf. Design and evaluation of a wide-area event notification service. *ACM TOCS*, 19(3):332–383, August 2001.
5. R. Chand and P.A. Felber. XNet: a reliable content based publish subscribe system. In *Proc. of the 23rd Symp. on Reliable Distributed Systems*, Florianópolis, Brazil, Oct 2004. IEEE Computer Society Press.
6. X. Chen, Y. Chen, and F. Rao. An efficient spatial publish/subscribe system for intelligent location-based services. In *Proceedings of the 2nd international workshop on Distributed Event-Based systems*, New York, NY, USA, 2003. ACM Press.
7. Y. Chen and K. Schwan. Opportunistic overlays: Efficient content delivery in mobile ad hoc networks. In *Proc. of the 6th ACM/IFIP/USENIX Int. Middleware Conf.*, LNCS 3790, pages 354–374, Grenoble, France, November 2005. Springer.
8. P. Costa and G. P. Picco. Semi-probabilistic Content-based Publish-subscribe. In *Proc. of the 25th Int. Conf. on Distributed Computing Systems (ICDCS05)*, pages 575–585, Columbus (OH, USA), June 2005. IEEE Computer Society Press.
9. G. Cugola and J. E. Munoz de Cote. On introducing location awareness in publish-subscribe middleware. In *ICDCSW '05: Proceedings of the Fourth International Workshop on Distributed Event-Based Systems (DEBS) (ICDCSW'05)*, pages 377–382, Washington, DC, USA, 2005. IEEE Computer Society.
10. G. Cugola, E. Di Nitto, and A. Fuggetta. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Trans. on Software Engineering*, 27(9):827–850, September 2001.
11. A. Dey and G. Abowd. The context toolkit: Aiding the development of contextaware applications, 1999.

12. Patrick Th. Eugster, Beno&#238;t Garbinato, and Adrian Holzer. Location-based publish/subscribe. In *NCA '05: Proceedings of the Fourth IEEE International Symposium on Network Computing and Applications*, pages 279–282, Washington, DC, USA, 2005. IEEE Computer Society.
13. F. Fabret, H.A. Jacobsen, F. Llirbat, J. Pereira, K.A. Ross, and D. Shasha. Filtering algorithms and implementation for very fast publish/subscribe systems. *ACM SIGMOD Record*, 30(2):115–126, 2001.
14. L. Fiege, G. Mühl, and F.C. Gärtner. Modular event-based systems. *Knowledge Engineering Review*, 17(4):359–388, 2002.
15. R.E. Gruber, B. Krishnamurthy, and E. Panagos. The architecture of the READY event notification service. In P. Dasgupta, editor, *Proc. of the ICDCS Workshop on Electronic Commerce and Web-Based Applications*, Austin, TX, USA, May 1999. IEEE Computer Society Press.
16. Christine Julien and Gruia-Catalin Roman. Egospaces: Facilitating rapid development of context-aware mobile applications. *IEEE TSE*, 32(5):281–298, 2006.
17. C. Maihfer. A survey on geocast routing protocols. *IEEE Communications Surveys and Tutorials*, 6(2), 2nd quarter issue 2004.
18. R. Meier and V. Cahill. STEAM: Event-Based Middleware for Wireless Ad Hoc Network. In *Proc. of the* 22^{nd} *Int. Conf. on Distributed Computing Systems*, pages 639–644, Vienna, Austria, 2002. IEEE Computer Society.
19. R. Meier and V. Cahill. Steam: Event-based middleware for wireless ad hoc networks, 2002.
20. Amy L. Murphy, Gian Pietro Picco, and Gruia-Catalin Roman. Lime: A coordination model and middleware supporting mobility of hosts and agents. *ACM Trans. Softw. Eng. Methodol.*, 15(3):279–328, 2006.
21. S. Pallickara and G. Fox. NaradaBrokering: A Distributed Middleware Framework and Architecture for Enabling Durable Peer-to-Peer Grids. In *Proc. of the 4th ACM/IFIP/USENIX Int. Middleware Conf.*, pages 41–61, Rio de Janeiro, Brazil, June 2003. ACM Press.
22. Jamie Payton. *A Query-Centric Approach to Supporting the Development of Context-Aware Applications for Mobile Ad Hoc Networks.* PhD thesis, Washington University in St. Louis, 2006. Technical Report WUCSE-2006-49.
23. P.R. Pietzuch and J.M. Bacon. Hermes: A distributed event-based middleware architecture. In *Proc. of the* 1^{st} *Int. Workshop on Distributed Event-Based Systems (DEBS)*, Vienna, Austria, July 2002. IEEE Computer Society Press.
24. P.R. Pietzuch and J.M. Bacon. Peer-to-peer overlay broker networks in an event-based middleware. In *Proc. of the* 2^{nd} *Int. Workshop on Distributed Event-Based Systems (DEBS)*, San Diego, CA, USA, June 2003. ACM Press.
25. Manuel Rom&#225;n, Christopher Hess, Renato Cerqueira, Anand Ranganathan, Roy H. Campbell, and Klara Nahrstedt. A middleware infrastructure for active spaces. *IEEE Pervasive Computing*, 1(4):74–83, 2002.
26. B. Segall, D. Arnold, J. Boot, M. Henderson, and T. Phelps. Content Based Routing with Elvin4. In *Proc. of AUUG2K*, Canberra, Australia, June 2000.
27. TIBCO Inc. TIBCO Rendezvous. `www.tibco.com`.
28. A. Varga. OMNeT++ Web page, 2003. `www.omnetpp.org`.
29. E. Yoneki and J. Bacon. An adaptive approach to content-based subscription in mobile ad hoc networks. In *Proc. of the 2nd IEEE Annual Conference on Pervasive Computing and Communications Workshops*, Orlando, FL, March 2004.
30. H. Zhou and S. Singh. Content-based multicast for mobile ad hoc networks. In *Proc. of the* 1^{st} *Annual Workshop on Mobile Ad Hoc Networking and Computing (Mobihoc 2000)*, Boston, MA, Aug 2000. ACM Press.