Reducing Latency in Rendezvous-Based Publish-Subscribe Systems for Wireless *Ad Hoc* Networks*

Nuno Carvalho
University of Lisbon
nunomrc@di.fc.ul.pt

Filipe Araujo
University of Lisbon
filipius@di.fc.ul.pt

Luís Rodrigues
University of Lisbon
ler@di.fc.ul.pt

Abstract

To ensure decoupling between publishers and subscribers, most publish-subscribe systems route notifications through intermediate message brokers. A byproduct of this practice is that notifications often follow suboptimal paths that are much longer than a direct path. Hence, in this paper, we propose a publish-subscribe architecture called GeoRendezvous *which aims to reduce the latency experienced by end clients in the delivery of notifications. We base our system on a position-based distributed hash table (DHT) that supports rendezvous points where the interests of publishers and subscribers match. Leveraging from previous work, we replicate the rendezvous points to give multiple choices of paths to the subscribers. We show that in this way, the subscriber is able to achieve latencies comparable to a direct publisher-subscriber path without breaking the decoupling assumptions of the publish-subscribe model. Additionally, we show that scalability is one of the most prominent features of* GeoRendezvous*, as the number of rendezvous points scales with the network size.*

1 Introduction

When compared to remote invocations, the weak coupling of event-based communication in general and of the publish-subscribe communication model in particular offers a number of advantages to create modular applications. To ensure decoupling, publish-subscribe clients typically interact through means of intermediate decentralized message brokers, which often form an overlay network [8, 7, 15, 17]. This often raises a dilemma to the designer: the system can disseminate advertisements and subscriptions all over the overlay, thus creating a lot of traffic. In alternative, the system may try to match publishers and subscribers in randomly placed rendezvous points, thus using suboptimal paths. In particular, most publish-subscribe systems cannot ensure timely delivery of notifications as the path that goes from the publisher to the subscriber is fixed by the overlay.

*This work was partially supported by the LaSIGE and by the FCT project P-SON POSC/EIA/60941/2004 via POSI and FEDER funds.

In this paper, we propose *GeoRendezvous*, which is based on the ideas presented in Hermes [15] and also leverages on systems like SCRIBE [17] or IndiQoS [6]. In *GeoRendezvous*, we focus on minimizing the latency experienced by clients of a publish-subscribe system in a wireless *ad hoc* network. *GeoRendezvous* runs atop of a wireless distributed hash table called "Cell Hash Routing" [1] (CHR) that we use to spread a number of rendezvous points. The hash function deterministically fixes the location of these rendezvous points, which can be accessed by all the publishers and subscribers of a given type. By having multiple path options, subscribers can choose the shortest path. One of the fundamental characteristics of *GeoRendezvous* is the use of a position-based DHT which maps the rendezvous points into fixed positions of the space. This allows the subscribers to take advantage of positional information to make hints on the rendezvous that will best serve them, thus keeping the signaling costs under control. In this way, subscribers concern only with their own position and pick a better rendezvous node as they move. Interestingly, we show that one of the strongest features of *GeoRendezvous* is its scalability. For all the network sizes we have tested, the number of rendezvous nodes necessary to achieve a given performance is independent of the network size.

The rest of the paper is organized as follows. Section 2 presents *GeoRendezvous*. Section 3 presents a theoretical analysis that describes the behavior of the *GeoRendezvous* system, which is experimentally evaluated in Section 4. Section 5 presents related work. Finally, Section 6 concludes the paper.

2 The *GeoRendezvous* Architecture

The *GeoRendezvous* architecture is a topic-based publish-subscribe middleware targeted for wireless networks. *GeoRendezvous* uses an underlying distributed hash table (DHT) to match advertisements and subscriptions.

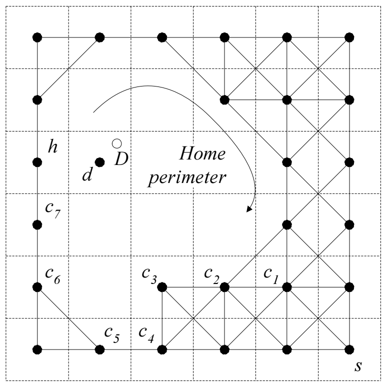


Figure 1. Routing in the CHR DHT

Given the identification of a type, we use a hash function to *deterministically* get a position in space, which corresponds to a rendezvous point. Then, the DHT ensures that there is at least one node responsible for this rendezvous point. In fact, in our particular implementation, there is an entire cluster of nodes inside a given geographical cell, called "rendezvous cell", that is responsible for that point. The rendezvous cell must store the subscriptions to that topic and forward each of the notifications it gets to the interested subscribers. The main contribution of *GeoRendezvous* is the use of a position-based DHT to support the publish-subscribe system. In a position-based DHT, nodes self organize according to their geographical positions and the keys hash to physical locations. This allows *GeoRendezvous* clients to use positional information to considerably reduce traffic and signaling costs.

If we used only one rendezvous cell, we would have no options concerning the choice of paths and, as a consequence, we could not improve the latency metric. On the other hand, even in a wired system, it is not feasible to collect network-wide information about advertisements, subscriptions and QoS (see for instance [9]) to select paths based on QoS constraints. This problem is exacerbated in a wireless system. Therefore, to deal with the inherent lack of information in *GeoRendezvous*, we replicate the rendezvous cells, in a way that is similar to *IndiQoS* [6]. This creates several different routing options, thus enabling the system to improve latency with only a limited impact on the signaling cost and state. In the following sections we detail the *GeoRendezvous* architecture, starting by the supporting DHT.

2.1 The Cell Hash Routing DHT

The Cell Hash Routing (CHR) is a cluster-based DHT [1]. In CHR, the space is divided into equally sized squares or cells and each cell acts as a virtual node that represents all the real nodes that are inside it. This virtual node is located in the precise center of the cell. Whenever a cell is populated, there is a virtual node representing it. The size of the cells must be chosen in a way that maximizes the probability that nodes inside a given cell can listen to all the nodes in any of the eight neighboring cells. On the other hand, the size of the cell cannot be made so small or no gain will result from clustering. Figure 1 depicts the CHR architecture. One crucial aspect of CHR is that it must be possible to unambiguously determine the center of the cell corresponding to any point in space. To do this, it suffices that the nodes agree on the size of the cells and on some arbitrary origin of space. One interesting aspect of CHR is that nodes do not need to have a precise notion of location. It suffices for them to know their cell and to reach at least one neighbor in each of the populated adjacent cells.

To route messages in CHR, we use a variation of the Greedy Perimeter Stateless Routing algorithm [13], GPSR, initially proposed in [4]. In CHR, our routing algorithm works as follows. Assume that some node S in cell s is forwarding a message to destination D in cell d. In this case, node S will consider itself to be in the center of cell s and consider its routing graph to have a shape like the one shown in Figure 1, i.e., all the edges are directed to the center of the populated neighboring cells. If S wants to route to neighboring cell c_1, it picks a random node from c_1, say C_1, and sends the message to this node. Cell x is represented by its central point and $N(x)$ is the set of populated cells that are adjacent to x. When using the greedy mode, S will select as the next hop, the cell c_1 such that $c_1 \in N(s) \wedge ||c_1 d|| < ||sd|| \wedge \nexists k \in N(s)\ ||kd|| < ||c_1 d||$. This means that S will send the message to neighboring node C_1, of cell c_1 that minimizes the distance to the destination, as long as this cell is closer to the destination. Otherwise, if a cell is a local minimum, e.g. c_3, node C_3 must send the message in perimeter mode to contour the face in the direction where the line $c_3 d$ lies. Then, as soon as the message reaches some cell h, such that $||hd|| < ||c_3 d||$, the message leaves the perimeter mode and reenters the greedy mode. If the message reaches the same node twice, say H, cell h, in perimeter mode without having passed through a closer neighbor of d, this means that i) cell d is empty and ii) cell d is inside the face contoured by the message (called "home perimeter") and therefore, h is the proxy destination of the message, which we call "home cell"[1]. Consider the example of Figure 1. To route a message to point D in space, starting from some node inside cell s, the message uses greedy routing to do the path $s - c_1 - c_2 - c_3$. Then, c_3 is a local minimum and the packet enters perimeter mode, going around the face through cells $c_4 - c_5 - c_6 - c_7$. Upon

[1]As explained in [1], if cell d is empty, nodes in cell h may under certain circumstances assume that h is the home cell of d before they send the message in the home perimeter.

arrival at cell c_7, the packet reenters greedy mode and the node holding it sends the packet to cell h, which is a local minimum. Cell h is, in fact, the home cell of d and the packet will contour the entire home perimeter until reaching h again. Araújo *et al.* proved in [1] that routing always converges in CHR, given that no edges exist between non adjacent cells. A similar routing scheme was first proposed in [16], for a non-clustered environment.

2.2 Basic Operation of *GeoRendezvous*

GeoRendezvous uses the hash function of the DHT to output pseudo-arbitrary positions in space, given the unique identification of a topic. Hence, consider that a client wants to subscribe for the topic "cars". It must compute the hash of "cars", $hash$("cars"), which will provide a pseudo-random position in space, say $(138, 144)$. This means that the rendezvous point of the topic "cars" is $(138, 144)$. In the case of CHR, this will define a unique rendezvous cell. All the nodes of that cell can be made responsible for that point (optionally, to conserve resources, keys inside a cell can be divided by the nodes). Note that *GeoRendezvous* can work with any DHT that hashes keys to positions in space and that uses position-based routing, namely with Geographical Hash Table [16] (GHT). However, we use CHR, because it scales better with node density than unclustered approaches [1] and, additionally, it is easy to take advantage of redundancy of nodes inside each cell, as more than one node can take care of a key.

Consider that there is some publisher P willing to notify subscribers on the topic "cars" and some subscriber S_i interested in those notifications. P will send an advertisement to the rendezvous cell of the topic "cars". On the other hand, S_i must also send a subscription to the same rendezvous cell. When there is a match, all the notifications that get to the rendezvous cell are forwarded to S_i. To reduce traffic, when a subscription matches an advertisement, a node in the rendezvous cell becomes responsible for forwarding the subscription to P. The rendezvous cell only needs to do this for the first subscriber S_1. To enable *GeoRendezvous* to react to changes in the topology and in the parties, publishers and subscribers periodically repeat their advertisements and subscriptions. Clients that fail to renew their advertisements/subscriptions are removed from the system.

2.3 Replication of the Rendezvous Points

The latency in the architecture that we have described so far is determined by the single path publisher-rendezvous cell-subscriber. To have more options we simply use more than a single rendezvous point. For each rendezvous point i, we provide i as an additional parameter to hash with the same key (i.e., the topic name) to independent positions of

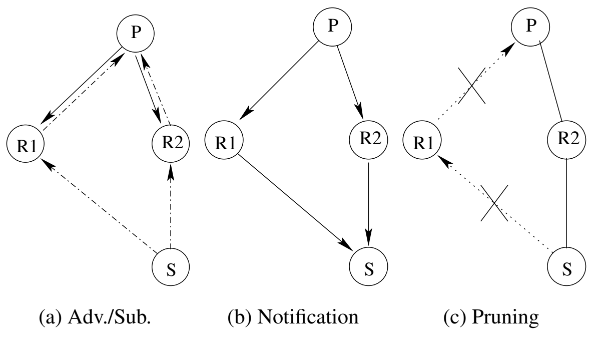


Figure 2. *GeoRendezvous* **architecture**

the space. Hence, assume that we have n rendezvous points and that there is a publisher P willing to notify subscribers on the topic "cars". This publisher will send an advertisement to each of the n rendezvous points. The subscriber also sends a subscription to all the n rendezvous nodes. Then, when some node P sends a notification, a subscriber will receive the notification n times. It must prune $n - 1$ paths, by selecting the single path to the rendezvous cell that has the shortest number of hops and removing all the others (it can break ties in many different ways, like comparing distance to the rendezvous point or selecting the path where the first notification arrived). This soft-wires a publisher and a subscriber through just a single rendezvous cell.

To reduce traffic, when there are no other subscribers interested, the rendezvous cells forward the prune messages toward the publisher. This ensures that the publisher will only send notifications through paths where it actually has receivers. However, it should be noticed that advertisements are never pruned, i.e., they remain in the rendezvous cells for the life of the publisher. Like in the single rendezvous node, clients must resend their messages periodically to refresh the state at the rendezvous cells. Figure 2 illustrates these interactions in the *GeoRendezvous* architecture.

2.4 The Use of Position to Optimize *GeoRendezvous*

One of the most interesting aspects of using geographical position as part of the system is that it becomes easier for subscribers to make hints on whether some of the rendezvous points are good or not. Hence, we considerably reduce the signaling cost associated with each subscription by limiting the subscriber to select only a subset of the rendezvous points that are geographically close to it. In other words, instead of letting the subscriber search for the best rendezvous among n points, with all the associated costs, the subscriber selects just the m rendezvous points that are physically closer to it. Since position bears a close connection to topology in wireless networks, we expect this optimization to have only a small impact on performance in exchange of significant savings in signaling traffic, especially for large values of n.

3 Theoretical Analysis

In this section, we show that under certain conditions that are likely to hold in a wireless environment with a uniform distribution of nodes, the probability of having a nicely placed rendezvous does not decrease below a certain threshold, despite the number of nodes of the network. $d(A, B)$ is the number of hops of the shortest path between A and B. $N(X, r)$ is the set of nodes within r hops of node X. The total number of nodes of the network is N. The diameter of the network is L. $|N(X, L)| = N$. Random variable l represents the minimal number of hops between arbitrary pairs of nodes, being $E[l]$ its expectation.

Additionally, we assume that $i)$ the network follows a growth bounded model (e.g. [12]), where $|N(X, 2r)| \leq \Delta|N(X, r)|$, for a constant Δ; and $ii)$ $\exists\epsilon > 0 \mid \forall L,\ E[l]/L > \epsilon$.

Theorem 3.1 *Assume that $l = d(A, B)$ and $l_r = d(A, R) + d(R, B)$, where R is a randomly placed rendezvous node. $\forall k, \exists\rho > 0 \mid P(l_r \leq l + 2kl) > \rho$. This means that for any k, the probability of having a path through the rendezvous node not exceeding $l + 2kl$ is bounded by below by a constant ρ, i.e., it does not vanish.*

Proof 3.1 *Given two different distances r_1 and r_2, such that $r_2 > r_1$, we have the following relation:*

$$\frac{|N(X, r_2)|}{|N(X, r_1)|} \leq \Delta^{\lceil \log_2 \frac{r_2}{r_1} \rceil}$$

To demonstrate our theorem, we pick any node M in the optimal path between A and B. For any node $Y \in N(M, kl)$, we have $d(A, Y) + d(B, Y) \leq l + 2kl$. For a given l and L, we define $r(l, L)$ to be the ratio of nodes inside $N(M, kl)$ to the total number of nodes in the network. From the growth bounded assumption:

$$r(l, L) = \frac{|N(M, kl)|}{|N(M, L)|} \geq \frac{1}{\Delta^{\lceil \log_2 \frac{L}{kl} \rceil}}$$

This is a lower bound for the probability that a rendezvous node will ensure a path of at most $l + 2kl$, for each path length l. Additionally, the function of τ, $\Delta^{\log_2 \frac{k\tau}{L}}$ is convex if $\Delta > 2$ in the interval $(0, +\infty)$, concave if $\Delta < 2$ and linear if $\Delta = 2$. Therefore, considering all the possible path lengths and using Jensen's inequality, it follows that (we omit the cases where $\Delta \leq 2$ as they trivially follow):

$$\begin{aligned} P(l_r \leq l + 2kl) &\geq \sum_{\tau=1}^{L} r(\tau, L) P(l = \tau) \\ &\geq \frac{1}{\Delta} \sum_{\tau=1}^{L} \Delta^{\log_2 \frac{k\tau}{L}} P(l = \tau) \\ &\geq \frac{1}{\Delta} \Delta^{\log_2 \frac{k}{L} \sum_{\tau=1}^{L} \tau P(l=\tau)} \\ &= \Delta^{\log_2 k \frac{E[l]}{L} - 1} > \Delta^{\log_2 k\epsilon - 1} = \rho \end{aligned}$$

□

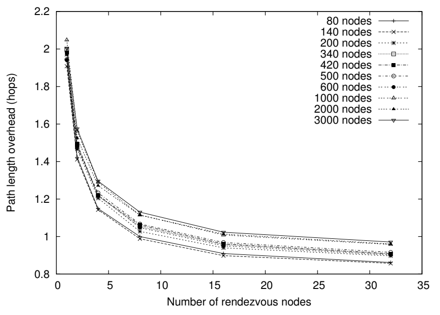
It directly follows from this result that if we have more than one rendezvous to place, the probability of having a path length through one of the rendezvous R_i such that $d(A, R_i) + d(R_i, B) < l + 2kl$ is also bounded by below. This shows that in a larger network, longer paths between arbitrary pairs of nodes balance the increased difficulty of placing a rendezvous in a favorable position.

4 Experimental Evaluation

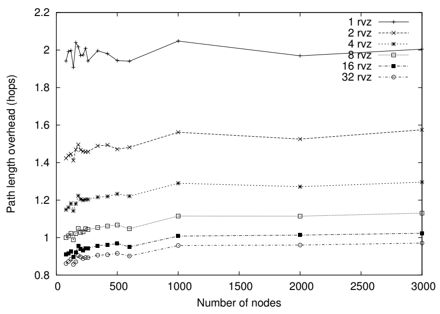
We now demonstrate the effectiveness of *GeoRendezvous*. We show that, with less than 8 rendezvous cells, *GeoRendezvous* can consistently achieve latencies similar to those of a direct path. Then, we show that using positional information, we can reduce the number of rendezvous cells to only 2 for a 600-node network. Finally, we take a configuration where a publisher sends a notification to randomly placed subscribers to compare the cost of sending notifications down a tree with multiple rendezvous cells, versus the cost of using a single rendezvous cell. We simulated networks with 80 to 3000 static uniformly distributed nodes (we did not consider packet collisions) inside a square. We assumed that communication range of nodes is a unit disk ray with an average network degree of 45 neighbors per node. All the results that we depict here result from an average of 20 different uniformly distributed publisher-subscriber paths in 200 different networks.

In Figure 3(a), we show the relative overhead of routing through a variable number of rendezvous cells, compared with a direct path from the publisher to the subscriber. We can see that the benefits of using more than 8 rendezvous cells are limited. Additionally, we can see in Figure 3(b) that performance is dictated by the number of rendezvous cells and it is quite stable and largely independent of the network size. This confirms the theoretical results of Section 3 and matches the wired case [6]. We believe that this independence of the network size is one of the most interesting aspects of replicating the rendezvous cells.

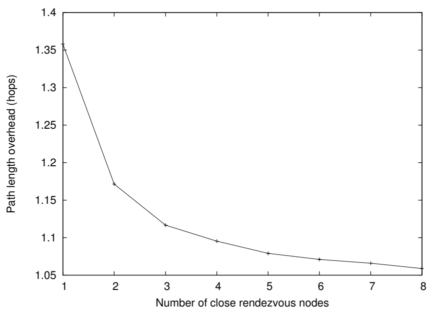
As we can see from Figure 2, there is a signaling cost associated with each rendezvous point. We assume a stable



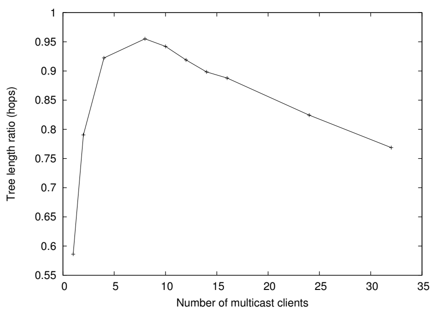
(a) Overhead per rendezvous node (vs. direct path)



(b) Overhead per network size (vs. direct path)



(c) Overhead per nearby rendezvous cell



(d) Length ratio

Figure 3. Results

setting, where a publisher is already notifying all the rendezvous. For each rendezvous that is not selected by the subscriber, we have the following unnecessary messages: a subscription, a notification and a pruning message (which unlike Figure 2 does not need to be forwarded to the publisher). For n rendezvous, this makes $3(n-1)$ unnecessary messages. To reduce n, we tried the optimization described in Section 2.4. We used a network with 600 nodes and set the total number of rendezvous cells to 8. Then each subscriber selected from 1 to (all the) 8 rendezvous by ascending order of physical distance. In Figure 3(c), we can see that with only 2 rendezvous cells, we stay close to the average direct publisher-subscriber path (and also close to the result achieved by 8 rendezvous cells in a 600-node network). The reader should confront this curve with the one depicted in Figure 3(a).

Another important question is whether the notification trees with multiple rendezvous cells generate more traffic than their counterparts with only one rendezvous cell. We depict the overhead of having 8 rendezvous cells (of which the clients only try the 2 closest) versus a tree with a single rendezvous cell in Figure 3(d) for a 600-node network. We uniformly distributed between 1 and 32 subscribers. Results are quite interesting. With less than 8 subscribers, it is better to have multiple rendezvous, because the publisher does not need to send a notification to them all (ratios < 1). On the other hand, as the number of subscribers approaches 8, the cost of the multiple-rendezvous tree grows to become nearly the same as in a single-rendezvous tree. Finally, when the number of subscribers grows, having 8 rendezvous cells becomes increasingly better than having only 1. The shorter path to the subscribers clearly pays for the burden of replicating the notifications in the publisher and every new subscriber helps to amortize this cost.

5 Related work

A limitation of most existing architectures that support publish-subscribe communication, even in wired networks, is their limited support for the negotiation or enforcement of Quality of Service (QoS) parameters (such as latency). This observation applies both to models such as the CORBA Event Service [14], the Java Message Service [10] and to systems, such as CEA (Cambridge Event Architecture) [2], SIENA (Scalable Internet Event Notification Architectures) [7], Combined Broadcast and Content-Based [8] (CBCB) or Hermes [15]. This is a significant drawback, because QoS features are important for applications [5, 3]. In particular, latency arguably remains the most important as well as difficult QoS parameter to optimize in a network.

Systems like CEA directly connect publishers and subscribers and do not ensure the adequate decoupling required by all the applications. One way to overcome this problem is through means of intermediate message brokers. For example, the Java Message Service [10] is a model that uses a broker that is conceptually centralized which brings a number of problems like scalability or tolerance to faults. This motivates many authors to follow decentralized approaches, like SIENA [7]. To preclude the flooding of control information, systems like Hermes [15] use a rendezvous node on top of a DHT. In [6], we proposed a type-based publish-subscribe system that uses more than one rendezvous node to ensure QoS requirements to the clients. Unfortunately, none of these systems is applicable to wireless networks, because they rely on overlays that need the (wired) IP protocol. Additionally, none of these solutions could take advantage of positional information.

There are also publish-subscribe systems built on wireless networks. One example is the Pronto [19] system which is a serverless JMS client for mobile applications that takes into account problems like resource constraints, network characteristics and data optimization. In a serverless implementation based on IP-multicast, JMS behaves in a decentralized way, which makes the system more fault tolerant. However, a publisher acts as a temporary server. In [11] Huang *et. al.* build publish-subscribe trees on top of a wireless network. The algorithm builds a spanning tree that maximizes a metric that takes into account network characteristics and limited device resources. These solutions compromise some decoupling properties of the publish-subscribe interaction. There are also multicast routing protocols that use multi-core trees [18, 20]. One common property of these algorithms is that they use a knowledge comparable to that of distance-vector protocols and, therefore, scale worse than position-based routing approaches.

6 Conclusions

In this paper we presented and evaluated a topic-based publish-subscribe architecture called *GeoRendezvous*. *GeoRendezvous* explores multiple rendezvous nodes and takes advantage of positional information to reduce the end-to-end latency experimented by the clients. We showed, by analysis and experimentally that our system can effectively reduce the latency, when compared to a single rendezvous node system, without compromising the scalability. In particular we obtained the following results: i) using positional information, we can get latencies close to a direct publisher-subscriber path using only 3 additional messages (in a stable scenario), ii) the probability of randomly getting a good place for a rendezvous point does not vanish with the network size and, for the network sizes we tested, it seems to be nearly constant; and iii) with the use of position, we can create a tree with multiple rendezvous nodes (cores) that is shorter than a tree with a single randomly placed core.

References

[1] F. Araújo, L. Rodrigues, J. Kaiser, C. Liu, and C. Mitidieri. CHR: a distributed hash table for wireless ad hoc networks. In *The 25th IEEE Int'l Conference on Distributed Computing Systems Workshops (DEBS '02)*, Columbus, Ohio, USA, June 2005.

[2] J. Bacon, K. Moody, J. Bates, R. Hayton, C. Ma, A. McNeil, O. Seidel, and M. Spiteri. Generic support for distributed applications. *IEEE Computer*, Mar. 2000.

[3] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An architecture for differenciated services, December 1998. RFC 2475.

[4] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in *ad hoc* wireless networks. In *Int'l Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIALM)*, pages 48–55, 1999.

[5] R. Braden, D. Clark, and S. Shenker. Integrated services in the internet architecture: an overview, June 1994. RFC 1633.

[6] N. Carvalho, F. Araújo, and L. Rodrigues. Scalable QoS-based event routing in publish-subscribe systems. In *The 4th IEEE Int'l Conference on Network Computing and Applications (NCA '05)*, Cambridge, MA, USA, July 2005.

[7] A. Carzaniga, D. Rosenblum, and A. Wolf. Design and evaluation of a wide-area event notification service. *ACM Transactions on Computer Systems*, 19(3):332–383, Aug. 2001.

[8] A. Carzaniga, M. J. Rutherford, and A. L. Wolf. A routing scheme for content-based networking. In *IEEE INFOCOM 2004*, Hong Kong, China, Mar. 2004.

[9] Y. Goto, M. Ohta, and K. Araki. Path QoS collection for stable hop-by-hop QoS routing. In *The Seventh Annual Conference of the Internet Society (INET '97)*, 1997.

[10] M. Hapner, R. Burridge, R. Sharma, J. Fialli, and K. Stout. *Java Message Service*. Sun Microsystems, April 2002.

[11] Y. Huang and H. Garcia-Molina. Publish/Subscribe Tree Construction in Wireless Ad-Hoc Networks. In *The 4th Int'l Conference on Mobile Data Management (MDM)*, volume 2574 of *LNCS*, pages 122–140, London, UK, 2003.

[12] D. R. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *STOC*, pages 741–750, 2002.

[13] B. Karp and H. T. Kung. GPRS: Greedy perimeter stateless routing for wireless networks. In *ACM/IEEE Int'l Conference on Mobile Computing and Networking*, 2000.

[14] OMG. *Event Service Specification*. Object Management Group, Mar. 2001.

[15] P. Pietzuch and J. Bacon. Hermes: A distributed event-based middleware architecture. In *22nd IEEE Int'l Conference on Distributed Computing Systems Workshops (DEBS '02)*, 2002.

[16] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. GHT: A geographic hash table for data-centric storage in sensornets. In *First ACM Int'l Workshop on Wireless Sensor Networks and Applications (WSNA)*, Atlanta, Georgia, September 2002.

[17] A. Rowstron, A.-M. Kermarrec, M. Castro, and P. Druschel. SCRIBE: The design of a large-scale event notification infrastructure. In *Networked Group Communication*, pages 30–43, 2001.

[18] C. Shields and J. J. Garcia-Luna-Aceves. The ordered core based tree protocol. In *Int'l Conference on Computer Communication (INFOCOM)*, pages 884–891, 1997.

[19] E. Yoneki and J. Bacon. Pronto: MobileGateway with publish-subscribe paradigm over wireless network, 2003.

[20] D. Zappala and A. Fabbri. An evaluation of shared multicast trees with multiple active cores. *LNCS*, 2093, 2001.