Extended ZRP: a Routing Layer Based Service Discovery Protocol for Mobile Ad Hoc Networks*

Christopher N. Ververidis and George C. Polyzos
Department of Computer Science
Athens University of Economics and Business
Athens 10434, Greece
{chris, polyzos}@aueb.gr

Abstract

Service discovery in Mobile Ad Hoc Networks is an essential process in order for these networks to be self-configurable with zero or minimal administration overhead. In this paper we argue that Service Discovery can be greatly enhanced in terms of efficiency (regarding service discoverability and energy consumption), by piggybacking service information into routing layer messages. Thus, service discovery does not generate additional messages and a node requesting a service, in addition to discovering that service, it is simultaneously informed of the route to the service provider. We extended the Zone Routing Protocol in order to encapsulate service information in its routing messages. Extensive simulations demonstrate the superiority of this routing layer-based service discovery scheme over that of a similar, but application layer based service discovery scheme.

1. Introduction

Much research has been devoted to Service Discovery in static networks, applied mostly to the Internet. The emergence of wireless communications and mobile computing devices has created the need for developing service discovery protocols and architectures targeted to mobile environments. Especially, the proliferation of Mobile Ad-Hoc Networks (MANETs) has introduced new requirements to service discovery due to the nature and inherent characteristics of these networks.

MANETs are extremely dynamic due to the mobility of their nodes, the wireless channel's adverse conditions and the energy limitations of small, mobile devices. The great majority of service discovery protocols developed for MANETs deal with the above issues at the application layer. In this paper we argue that by implementing service discovery at the routing layer, instead of the application layer, the resulting communication and energy consumption overheads are significantly reduced. Our approach is to implement service discovery in the routing layer by piggybacking the service information into the routing protocol control messages, thus enabling the devices to acquire both service and routing information simultaneously. This way a node requesting a service (henceforth called service requestor) in addition to discovering the service, it is also informed of the route to the service provider at the same time.

In this paper, we propose the piggybacking of service information in routing messages, in order to decrease communication overhead and save battery power. This way, besides these savings, we can also achieve smooth service discovery adaptation to severe network conditions (e.g. network partitions). Smooth adaptation occurs because service availability is tightly coupled with route availability to serving nodes. Hence when all routes towards a node fail, this is immediately translated to a loss of service availability for the services that this node provides. We demonstrate the benefits of our approach (i.e. routing layer supported service discovery) versus traditional application based service discovery, by extending the Zone Routing Protocol (ZRP), which is a hybrid routing protocol (i.e. proactive for a number of hops around a node called the node's zone, and reactive for requests outside this zone), so that it is

* This research is supported by project "Mobile Services" (EP-1221-06), funded by the research program "Pythagoras–Support for Research Groups at the AUEB," which is co-financed by the Ministry of National Education and Religious Affairs of Greece and the European Union, through the program "EPEAEK II."

capable of encapsulating service information in its messages.

The remainder of this paper is organized as follows. In section II we provide the essential background on service discovery by presenting the most significant research results. In section III we present our approach of routing layer based service discovery, and in section IV we provide simulation results along with their analysis. Finally in section V we provide our conclusions and refer to our future research directions.

2. Related Work

Significant academic and industrial research has led to the development of a variety of protocols, platforms and architectures for service discovery such as JINI [1], Salutation [2], UPnP [3], UDDI [4], Bluetooths' SDP [5] and SLP [6]. All these approaches, except SDP, are mainly targeted towards the discovery of services in fixed infrastructure networks. They are mostly centralized approaches that assume that reliable communication can be provided by the underlying network. Most of these approaches utilize nodes acting as (central) service directories-repositories, where service providers register the services they offer. Service requestors submit their queries to these 'special nodes' in order to discover services and information about the nodes that actually host these services. It is clear that such assumptions are not consistent with MANETs' inherent features due to their volatile nature.

This has motivated some recent approaches in the field, namely Allia [7], GSD [8], DEAPspace [9], Konark [10] and SANDMAN [11]. These approaches were developed with pervasive computing environments in mind, and are briefly presented in the next paragraphs. One aspect of the discovery approach which we consider significant and we pay particular attention to is energy consumption.

Allia is an agent based service discovery protocol, centered on peer-to-peer caching of service information. Every node in the network periodically broadcasts service advertisements. Nodes with similar types of services form alliances by caching each other's services. So, when a node receives a service request, which it cannot fulfill (doesn't have an appropriate service), it checks whether it has cached information about other nodes (allies) that offer similar services. In case such information is indeed cached, this node sends back the appropriate reply. If there is no cached information, then, depending on its policy, the node either broadcasts this request to the other nodes in its vicinity or forwards it to the members of its alliance. When a node caches service information sent by another node, then this node automatically becomes a member of the caching node's alliance. Allia uses Unique Universal Identifiers (UUIDs) for services, which should be a-priori known to all nodes. However, Allia is entirely agent based and hence it is too demanding in terms of computational power and resources in general. It also does not address energy consumption, and no related measurements or metrics are provided.

Another approach is the Group-based Service Discovery Protocol (GSD). GSD is also based in peer-to-peer caching of service advertisements and selective forwarding of service requests. GSD generates fewer messages compared to a simple broadcasting scheme, since service requests are not broadcast but instead forwarded only to those nodes that have already cached information about similar services. However, GSD uses DAML-based service descriptions in the advertisement messages (instead of simple UUIDs) and performs semantic matching, thus increasing energy consumption.

Similarly to GSD, Konark is a distributed service discovery protocol based on peer-to-peer caching of service information. In Konark, every node maintains a service registry, where it stores information about its own services and also about services that other nodes provide. This registry is actually a tree-structure with a number of levels that represent service classification. Upon receiving a service advertisement, a node updates its registry by classifying that service under the appropriate leaf of its tree. Service advertisements are in an XML-like language (similar to WSDL but smaller), hence allowing semantic matching, leading to increased energy consumption, but more precise resolutions. Konark uses multicasting for service requests and unicasting for service replies; hence it is more efficient than simple broadcasting schemes in terms of messaging overhead.

DEAPspace employs a periodic broadcast scheme for service advertisements. Each node sends the full list of services that it is aware of in its one-hop vicinity. Hence DEAPspace is targeted to smaller networks than Konark. In DEAPspace each node listens to its neighbors' broadcasts. In case the node doesn't find its own services in these messages, it schedules a broadcast sooner than usual, informing all the others about its presence and the services it can provide. In contrast to the aforementioned approaches, DEAPspace deals with the problem of energy consumption explicitly, by forcing weak nodes to go into idle mode during pauses between (the periodic) broadcasts.

SANDMAN, like DEAPSpace, is another service discovery protocol that implements power savings. This is done by grouping nodes with similar mobility patterns into clusters; in each cluster, one of the nodes (called clusterhead) stays awake permanently and

answers discovery requests. The rest of the nodes periodically wake up to provide the actual services and also inform the clusterhead about their presence and services. The clusterheads are re-elected periodically to avoid draining a single node's battery. Simulation results show energy savings up to 40% for low numbers of service requests. Increasing the size of a cluster can attain even higher savings. However, this results in a dramatic increase of the average interaction latency due to the fact that a requesting node has to wait the sleeping node to wake up in order to interact with its services.

It is clear from the above discussion that only the latter two approaches take into account energy consumption and provide related metrics and comparisons. A key difference of our approach from those is that we do not expect or allow the nodes to go into sleep mode, since we target environments where continuous communication is necessary. The other aforementioned approaches do not provide specific results regarding energy consumption. Our approach explicitly deals with power savings resulting from a routing layer supported service discovery scheme by modeling, simulating and recording energy consumption. Finally in [12] an architecture called CARD is developed, using a zone-based protocol for service discovery and energy measures are provided for its performance. However, the focus of CARD is more on out-of-zone resource discovery and is specific to short-term transactions only. Also the provided measurements do not take into account MAC layer issues, like collisions. In our case measurements are the result of simulating the whole protocol stack from the application layer down to the physical layer. What differentiates our work is that we use ZRP both for service and route discovery for all kinds of transactions and especially for intra-zone transactions. In the next section we present our approach in detail and justify our design decisions.

3. Routing Layer Supported Service Discovery

Our motivation for adding routing layer support for service discovery stems from the fact that any service discovery protocol implemented above the routing layer will always require the existence of some kind of routing protocol for its own use. Hence, two message-producing processes must coexist: the first one communicates service information among service providers and service requestors; the second one communicates routing information among them. As a result, a node is forced to perform multiple times the battery-draining operation of receiving and transmitting (control) packets. Our approach exploits the capability of acquiring service information along with routing information (from the same message) by piggybacking service information onto routing messages. This way, redundant transmissions of service discovery packets at the application layer are avoided and energy is saved.

The idea of providing routing layer support for service discovery was first introduced by Koodli and Perkins in [13]. They argue that for proactively routed MANETs, a service reply extension added to topology updating routing messages is enough for providing both service discovery and route discovery concurrently. In reactively (or on-demand) routed MANETs, the service discovery process follows the traditional route discovery process by using its message formats for route requests (RREQ packets) and route replies (RREP packets) extended to carry also a service request or reply respectively. However, as far as we know, no experimental assessment of Koodli's and Perkins' proposal has been published until now.

As stated in the introduction we have extended the Zone Routing Protocol (ZRP) [14] so that it provides service discovery functionality. ZRP was selected because: (a) it is ideal for environments where local information-either routing or service information-is of particular interest, as it provides discovery (through the notion of zones described further on) in a fast and energy efficient way and (b) it is scalable, as it intelligently propagates information to distant nodes by avoiding flooding. In this paper we present experimental results using extensions on the proactive part of ZRP, while our current and future work is focused on extending its reactive part as well. Next, we describe the basic operation of ZRP and the extensions we have introduced in order to enhance it with service discovery capabilities.

ZRP

We proceed to describe the ZRP's structure and operation. ZRP actually consists of three sub-protocols, namely:

- The Neighbor Discovery Protocol (NDP), through which every node periodically broadcasts a "hello" message to denote its presence.
- The Intra Zone Routing Protocol (IARP), which is responsible for proactively maintaining route records for nodes located inside a node's routing zone (for example records for nodes located up to 2-hops away). This is depicted in fig.1 where nodes B to H are inside the routing zone of node A; hence node A is proactively aware of all the routes to these nodes through IARP.
- The Inter Zone Routing Protocol (IERP), which is responsible for reactively creating route records for

nodes located outside a node's routing zone (e.g. records for nodes located further than 2-hops away).

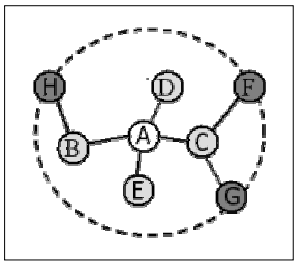

Fig.1: ZRP 2-hop zone

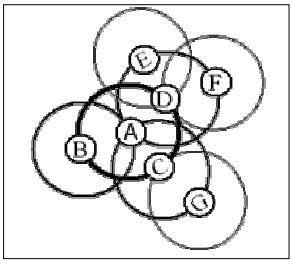

Fig.2: ZRP bordercasting

In ZRP, a node in search of a route towards a node outside its zone, unicasts the route request only to nodes located at the borders of its zone. This method is called bordercasting and is depicted in Figure 2. The border nodes check their IARP tables to find if the requested node is included in their respective routing zones; if not they also bordercast the request to their own border-nodes. When the requested node is found, a reply is unicasted back to the node that initiated the request. This way, global flooding is avoided and distant resources are discovered in an efficient and scalable manner.

E-ZRP

In order to add service discovery capabilities to ZRP we embedded an extra field in NDP "hello" messages for storing service IDs. We used the concept of Unique Universal Identifiers (UUIDs) instead of service descriptions, keeping packet lengths small for the routing messages and minimizing the effects on the network (the bigger the messages the larger the delays and the possibility of transmission errors). Such an approach implies that all nodes know a-priori the mappings between services offered in the MANET and UUIDs. This is a common assumption and is justified by the fact that most MANETs are deployed for certain purposes where there is lack of fixed communication infrastructure (e.g. a battlefield or a spot of physical disaster). In such environments, the roles of every participating node are concrete and can be easily classified in types of services. For example, in a battlefield one node may offer radar information to the rest, while another one may offer critical mission update information. In the case of a disaster such as an earthquake, an on-site relief team usually consists of members having different missions (e.g. one may be able to provide information about trapped people under ruins, another may provide information about terrain stability, and others may try to find and provide valuable structural information about the collapsed buildings etc.). In such environments the mapping of services to UUIDs is more than sufficient for service discovery. Semantic matching of rich service descriptions is of no particular use in these cases, not to mention that these techniques lead to increased energy consumption (a scarce and valuable resource in the above scenarios). Thus, by extending "hello" messages with service UUIDs, a node is able to denote both its presence and the services it provides.

ZRP was further extended in order to include service information in every routing entry of the IARP routing messages and tables. IARP listens to information gathered from NDP messages, updates its table and then periodically broadcasts its table to its neighbors. A node broadcasting this IARP update packets sets the TTL (Time To Live) field in these packets equal to its routing zone diameter, so that they will be dropped at border nodes. This way each node knows the routes to all the nodes in its zone and also the services that these nodes offer; thus adding the service discovery capability to the proactive part of ZRP.

The extended version of ZRP we implemented (henceforth called E-ZRP) is capable of providing routing layer support for proactive service discovery. In the following section we present our simulation results from applying E-ZRP in multiple scenarios.

4. Performance Analysis of E-ZRP

Our simulations were conducted using the Qualnet Simulator [15], which has a ZRP module. A basic assumption in our simulations is that each node hosts a unique service, which can be provided to other nodes. This was done for simplicity and in order to facilitate the analysis of the results. At the physical and data-link layer we used the IEEE 802.11b protocol.

As previously stated our goal was to compare E-ZRP with a traditional flooding scheme for service discovery. We conducted 4 sets of experiments. In the first 2 sets the parameter settings for configuring both protocols where chosen to be identical, so that a fair comparison between the two approaches (i.e. application layer and routing layer service discovery) is feasible. In the last 2 sets we modified this parameters so as to “favor” application layer service discovery by employing larger update intervals compared to these used in the routing layer, hence minimizing the produced overhead as much as possible. Table 1 summarizes the settings for the first 2 sets of experiments. The IARP Zone Radius is equal to the flooding radius; this implies that restricted area flooding is performed, as opposed to global flooding. The broadcast interval is used by IARP in order for a node to send at regular time interval all the information it has (zone routing information in the original ZRP, zone routing and service information in

E-ZRP) to neighboring nodes. The same interval is used in Flooding as well, with the difference that flooding messages are much shorter containing only a node's own service UUID and no routing information or other nodes' service UUIDs. The IARP deletion interval and the Service deletion interval, define the time after which a node erases records that haven't been updated.

Table 1: Protocol Settings

IARP Zone Radius	3 hops
IARP broadcast interval	10 seconds
IARP deletion interval	40 seconds
Flooding Radius	3 hops
Flooding broadcast interval	10 seconds
Service deletion interval	40 seconds

In our first set of experiments, the two approaches are tested in a static context (i.e. nodes do not move). In order to facilitate the analysis, we designed a "chain topology," where nodes are placed in a row, each one of them having exactly one neighbor to the left and one to the right (except from the first and the last node of the chain). We conducted several experiments, altering each time the number of the participating nodes. Each experiment had duration of 1000 seconds (simulation time). The results of these experiments are presented in Fig. 3 and Fig. 4.

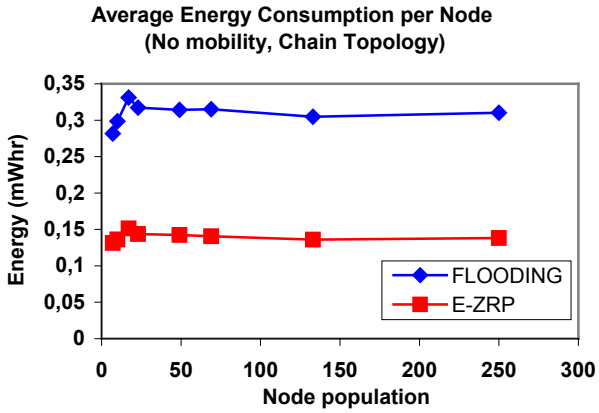

Fig. 3: Average Energy Consumption per node in a static context (E-ZRP versus Flooding).

Fig. 3 clearly shows that the energy consumption for E-ZRP is almost always 50% less than that for Flooding, irrespectively of the number of participating nodes. This happens because in the Flooding experiments, ZRP is also used at the routing layer to actually route packets. So, in the case of the Flooding scheme there are two processes creating messages: one at the application layer for service discovery and another one in the routing layer for route discovery. This application layer overhead in messages, leads to the observed dramatic difference of energy consumption between the two schemes. Also, it is evident for both approaches that energy consumption remains almost the same irrespectively of the node population. This is explained by the fact that the average number of every node's neighbors remains the same . In this static chain topology, every node exchanges information only with those nodes located inside its zone, and so energy consumption remains almost constant.

Fig. 4 depicts the average number of services discovered per node. What is worth noting is that a node using E-ZRP is able to discover on the average almost the same number of services, as compared to Flooding. The restricted-area Flooding scheme employed, performs slightly better than E-ZRP because flooding packets (containing information about 1 service only) are shorter than IARP packets (containing information for all services provided in a node's zone) and hence are less susceptible to transmission errors. On the average (over every node population) we get only 9,2% less services discovered when using E-ZRP, which is negligible as compared to the achieved energy savings of 47% on the average.

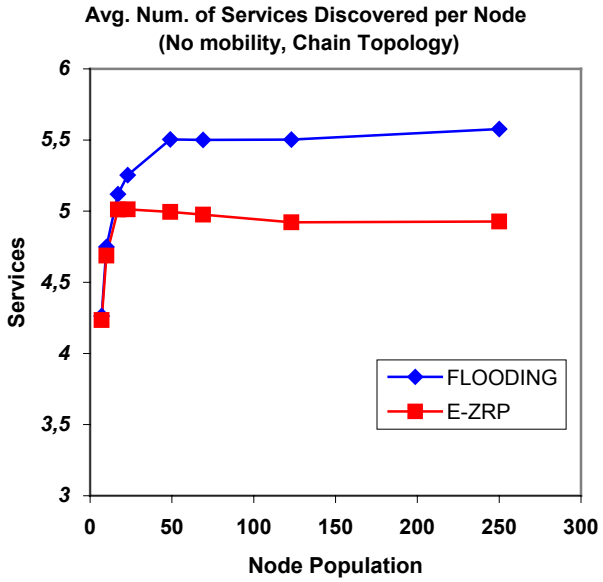

Fig. 4: Average Number of Services Discovered per node proactively in a static context (E-ZRP versus Flooding).

Considering the above results, it is clear that E-ZRP is more efficient than Flooding when there is no node mobility and both protocols have the same parameter settings (especially their update interval). In the following paragraphs we also test the two schemes under mobility conditions.

In the second set of experiments, the two service discovery schemes are tested in a mobile context (i.e. nodes do move). It is important to note that for stability reasons the density is kept fixed when varying the number of nodes (node population) by resizing the terrain in which they are allowed to move. Every node in the simulated scenarios uses the random waypoint model with the following parameters:

- Minimum Speed = 0meters/second (m/s)
- Pause Time = 30 seconds.
- Maximum Speed takes the following values: 0.5m/s, 1m/s, 2m/s, 5m/s, 7.5m/s, 10m/s and 12.5m/s in order to test service discovery and energy consumption under different speeds.

Fig. 5 and Fig. 6 depict the results for service discovery and energy consumption respectively in this mobile context. Each spot in the diagrams represents an average value obtained by running the experiment over 8 different randomly chosen node populations (spanning from 10 nodes to 250 nodes).

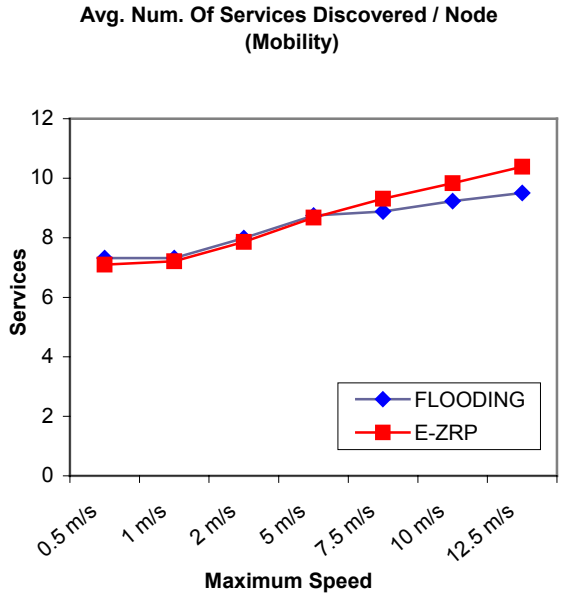

Fig. 5: Average Number of Services Discovered per node proactively in a mobile context (E-ZRP versus Flooding). The random way point mobility model is used with pause time 30 seconds and minimum speed 0m/s.

Regarding service discoverability (Fig.5) the two protocols give almost identical results. We observe that both protocols perform better when speed increases (this means that each node will meet more nodes throughout its lifetime), with E-ZRP being better only when the maximum speed is set at 7.5m/s or more, hence giving 2% more services on average (across all speeds). The main reason is that in E-ZRP, IARP packets contain much more information about available services in a node's zone, compared to flooding packets that only contain information about the service that their sender provides. Hence, when speed increases and successful packet transmissions are decreased (nodes remain much less time in each others transmission range), one IARP packet that successfully reaches a node is much more informative than several Flooding packets that may reach this node. As expected, energy consumption (Fig. 6) follows the same pattern (i.e. it increases when speed increases), which is explained by the fact that every node meets more nodes when moving at higher speeds; hence more bytes are received, leading to increased energy consumption. Energy consumption is on average 45% less for E-ZRP compared to flooding (across all speeds).

The above simulation results prove the superiority of the routing layer service discovery approach compared to a traditional application layer service discovery when both layers work with identical parameter settings. This superiority is expressed in terms of significantly improved energy efficiency in both mobile and static environments with almost the same number of services discovered.

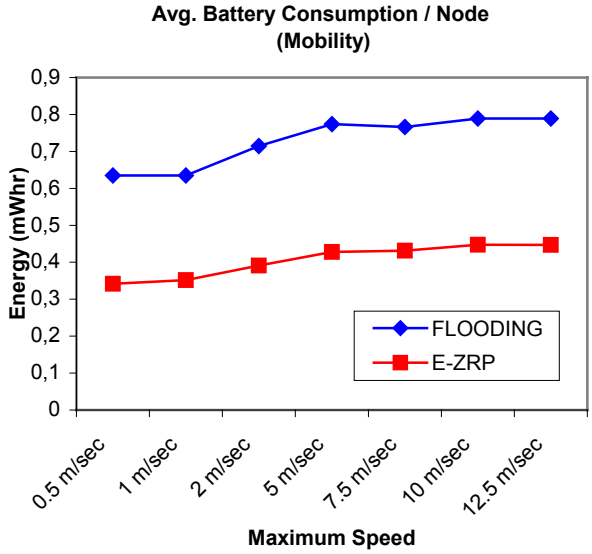

Fig. 6: Average Amount of Energy Consumed per node in a mobile context (E-ZRP versus Flooding). The random way point mobility model is used with pause time 30 seconds and minimum speed 0m/s.

The last 2 sets of experiments were conducted in order to investigate the performance of the application layer service discovery versus the routing layer service discovery scheme, when the update intervals used at

the application layer are larger. In these cases the application layer sends messages in larger time intervals and hence decreases the energy consumption. However this comes at the cost of decreased capability of discovering services. The purpose of these experiments was to show the optimal configuration of an application based service discovery scheme (based on updates in a restricted zone), so that service discoverability is equal or better to that achieved by a routing layer based approach. Table 2 summarizes these new settings. Note that service deletion interval will always be 4 times the broadcast interval for fairness reasons.

Table 2: Protocol Settings

	Flooding broadcast interval	Service deletion interval
A	200 seconds	800 seconds
B	160 seconds	640 seconds
C	80 seconds	320 seconds
D	40 seconds	160 seconds
E	20 seconds	80 seconds
F	15 seconds	60 seconds
G	10 seconds	40 seconds

So, in our third set of experiments, the two approaches are again tested in a static context with a "chain node topology," Since in the previous similar experiments we showed that results do not vary much over different network sizes, we conducted experiments, over a network with 250 participating nodes. Each experiment had duration of 1000 seconds (simulation time). The results of these experiments are presented in Fig. 7. Each point on the blue curve corresponds to different parameter settings for the update and service deletion intervals (those presented in Table 2) for the flooding protocol. The vertical and horizontal blue dotted lines denote the energy consumption and the number of services discovered respectively, for E-ZRP with a broadcast interval of 10 seconds. It is evident that the application layer service discovery approach (flooding) may perform better than the routing layer approach in terms of service discoverability for broadcast intervals lower than 40 seconds. However, this comes at the cost of energy consumption, which is increased 30% or more compared to the routing layer approach with the original broadcast interval of 10 seconds. This is again explained by the fact that the messages of the application layer scheme are much shorter (in order to be more economic) and hence less informative than those of the routing layer approach. So, service discoverability is reduced by reducing the number of broadcasted messages (bigger intervals means less messages transmitted, hence every node receives less information about services).

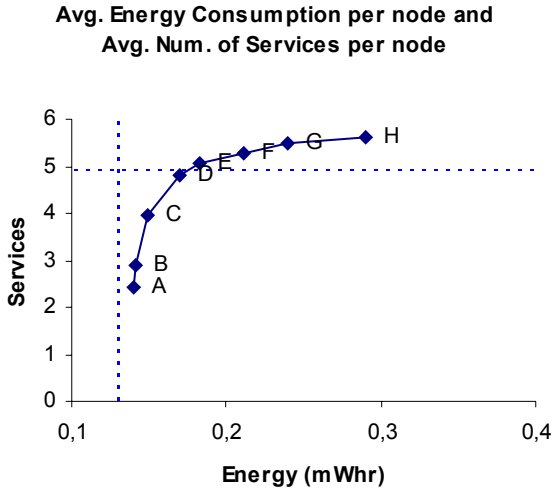

Fig. 7: Relating Average Energy Consumption per node to Average Number of Services discovered per node without mobility.

In the fourth set of experiments, the two approaches are tested in a mobile context. All the parameters (e.g. regarding node mobility) besides flooding broadcast interval and service deletion interval are the same as those used at the second set of experiments analyzed in previous paragraphs.

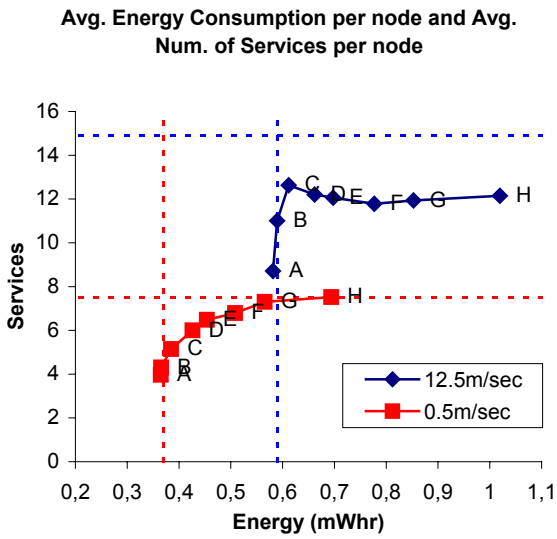

Fig. 8: Relating Average Energy Consumption per node to Average Number of Services discovered per node for high and low mobility.

Fig. 8 depicts the results for service discovery and energy consumption respectively in this mobile context. Each experiment was run over a network of 250 nodes (density remains fixed as in previous experiments. We study 2 extreme cases of mobility. The first case is for low mobility, where nodes move according to the random waypoint mobility model with minimum speed 0m/s, maximum speed 0,5m/s and pause time 30 seconds. The second case is for high mobility, where the mobility parameter of maximum speed changes to 12,5m/s. Each point on both curves corresponds to different parameter settings for the update and service deletion intervals (those presented in Table 2) for the flooding protocol.

As it is shown from fig. 8 the application layer based service discovery scheme reaches its optimal performance in terms of energy consumption (compared to the routing layer approach) when the broadcast interval is equal or more than 160 seconds saving 3% more power but discovering 43% less services for low mobility cases and 22% less services for high mobility cases.

5. Conclusions and Future Work

Existing application layer service discovery architectures suffer from redundant packet transmissions in their effort to discover routes towards the services (in the sense that control messages for information discovery are required at both the network and application layers). In the literature it has been proposed that service discovery may be integrated with routing, thus allowing nodes to find available services and routes to them simultaneously. This way fewer messages are broadcast onto the network. However, no experimental assessment of energy savings from employing a routing layer service discovery scheme versus a traditional application layer-based service discovery scheme has been presented. In this paper, we presented a new routing layer architecture that integrates service discovery functionality with an existing routing protocol. Our main focus was on demonstrating the impact of such an approach, on energy consumption. We have experimentally shown that our implementation consistently outperforms an application-layer service discovery scheme based on restricted-area flooding in terms of energy consumption, both in static and mobile environments. Our proposed protocol (E-ZRP) leads to significantly smaller energy consumption (approximately 50% less), but also, in certain cases, it achieves higher service discoverability. It was also shown that 'favoring' the application layer based service discovery protocol with larger flooding intervals (in order to become more economic in terms of energy consumption (savings of 3%)), had a detrimental effect in service discoverability reducing it by 22% or more compared to the proposed routing layer based approach.

In our future work we will enhance our approach of providing routing layer support for service discovery by extending the reactive part of ZRP. We also plan to conduct extensive simulations in order to test the reactive service discovery capabilities of E-ZRP by employing the appropriate simulation scenarios. Finally we will further investigate the impact of node density on E-ZRP's performance.

REFERENCES

[1] Sun Microsystems, *JINI Architecture Specification*, Nov. 1999.

[2] Salutation Consortium, "Salutation Architecture Specification," http://www.salutation.org/ specordr.htm, 1999.

[3] Microsoft Corporation, "Universal Plug and Play: Background", http://www.upnp.org/ resources/UPnPbkgnd.htm, 1999.

[4] Universal Description Discovery and Integration Platform, http://www.uddi.org/pubs/ Iru_UDDI_Technical_White_Paper.pdf, Sept. 2000.

[5] "Specification of the Bluetooth System," http://www.bluetooth.com, December 1999.

[6] E. Guttman, C. Perkins, J. Veizades, and M. Day. "Service Location Protocol, Version 2," IETF RFC 2608, June 1999.

[7] O. Ratsimor, D. Chakraborty, S. Tolia, D. Kushraj, A. Kunjithapatham, G. Gupta, A. Joshi, T. Finin, "Allia: Alliance-based Service Discovery for Ad-Hoc Environments," in Proc. ACM Mobile Commerce Workshop, September 2002.

[8] D. Chakraborty and A. Joshi, "GSD: A novel group-based service discovery protocol for MANETS", In IEEE Conference on Mobile and Wireless Communications Networks, Stockholm, Sweden, September 2002.

[9] M. Nidd, "Service Discovery in DEAPspace", IEEE Personal Communications, August 2001, pp. 39-45.

[10] S. Helal, N. Desai, V. Verma, and C. Lee, "Konark - A Service Discovery and Delivery Protocol for Ad-Hoc Networks", in Proceedings of the 3rd IEEE Conference on Wireless Communication Networks (WCNC), New Orleans, Louisiana, March 2003.

[11] G. Schiele, C. Becker and K. Rothermel,"Energy-Efficient Cluster-based Service Discovery for Ubiquitous Computing," in Proceedings of the 11th ACM SIGOPS European Workshop, Leuven, Belgium, September 2004.

[12] A. Helmy, S. Garg, N. Nahata and P.Pamu, “CARD: A contact-based Architecture for Resource Discovery in Wireless Ad Hoc Networks,” Mobile Networks and Applications (MONET), Vol 10, pp.99-113, 2005.

[13] R. Koodli and C. E. Perkins, “Service discovery in on-demand ad hoc networks,” IETF Internet Draft, draft-koodli-manet-servicediscovery-00.txt, October 2002.

[14] Z.J. Haas, M.R Pearlman, P. Samar, “The Zone Routing Protocol (ZRP) for Ad Hoc Networks,” IETF Internet Draft, draft-ietf-manet-zone-zrp-04.txt, July 2002.

[15] Scalable Network Technologies Inc., “Qualnet Simulator,” http://www.scalable-networks.com