

# Resource Discovery in Activity-Based Sensor Networks

Doina Bucur and Jakob E. Bardram

Centre for Pervasive Healthcare

Department of Computer Science, University of Aarhus

{doina,bardram}@daimi.au.dk

**Abstract**— This paper proposes a service discovery protocol for sensor networks that is specifically tailored for use in human-centered pervasive environments. It uses the high-level concept of *computational activities* (as logical bundles of data and resources) to give sensors in *Activity-Based Sensor Networks* (ABSNs) knowledge about their usage even at the network layer. ABSN redesigns classical network-level service discovery protocols to include and use this logical structuring of the network for a more practically applicable service discovery scheme. Noting that in practical settings activity-based sensor patches are localized, ABSN designs a completely distributed, hybrid discovery protocol which is proactive in a neighbourhood zone and reactive outside, tailored so that any query among the sensors of one activity is routed through the network with minimum overhead, guided by the bounds of that activity. ABSN enhances the generic *Extended Zone Routing Protocol* with logical sensor grouping and greatly lowers network overhead during the process of discovery, while keeping discovery latency close to optimal.

**Index Terms**— Sensor networks, resource discovery, Activity-Based Computing, pervasive healthcare, Activity-Based Sensor Networks

## I. INTRODUCTION

WIRELESS ad hoc sensor networks for medical purposes are playing an increasing role within healthcare. Body Sensor Networks (BSN) are being designed for prophylactic and follow-up monitoring of patients in e.g. their homes, during hospitalization, and in emergencies. For example, a wide range of medical sensor networks has been proposed for post-operative monitoring [1], heart monitoring [7], and follow up monitoring of e.g. strokes [11]. In the European PalCom project [20] and in the Code Blue project in Boston [5], medical sensors are being developed for patient monitoring in emergencies.

Core research questions in such wireless ad-hoc sensor networks include low-level data routing and service discovery protocols, i.e. the way to most efficiently - in terms of response time, network bandwidth and power consumption - route data in the network and to discover and access services within the network. Extensive research within efficient data routing in wireless ad-hoc networks has been done already. However, limited research has been done within the field of resource discovery in such networks, and most of this previous research either builds on service discovery protocols from a heavyweight IP-based network infrastructure, or is purely theoretical. Furthermore, most of this work does not take into account any knowledge about the usage of the network, and

hence becomes completely general and detached from the application domain.

In this paper, we propose to use the concept of *Activity-Based Computing* [2] for data routing and service discovery in wireless, ad hoc sensor networks. The core idea in activity-based computing is to help users organize computational services, data and resources in logical bundles that match their work activity. We call these bundles *computational activities*, or simply *activities*. For example, a healthcare activity would be the monitoring of a patient, which can take the concrete form of the prophylactic monitoring of Mr. Hansen for congestive heart failure by monitoring a combination of parameters like blood pressure, ECG, weight, and pulse. Similarly, in an accident, a monitoring activity for each victim could be created by bundling sensors monitoring respiration, pulse, oxygen saturation, temperature, and blood pressure. It is our experience that in medical settings it is practical to logically bundle services, data, and other resources around an activity, which is typically tied to a specific patient.

The core idea in *Activity-Based Sensor Networks* (ABSNS) is to utilize the clustering of sensors into logical activities, and then base data routing and service discovery on each sensor's knowledge of activity membership. Our hypothesis is that data routing and service lookup are more efficient in this protocol design than in a general uniform scheme, since services and their data are mostly relevant within the bounds of each patient activity.

### A. ABSN Requirements and Solutions

A trademark usage case for ABSN is the emergency or hospital healthcare setting; sensors are deployed as either body monitoring sensors on patients or as context sensors in the environment, and are dynamically and explicitly activity-grouped to patients or objects by nurses. An activity ID on each node groups sensors in activity clusters. While patients are moved to ambulances or other locations in the hospital, their body sensors

- record medical data into memory, and use the knowledge about belonging to a certain activity in order to efficiently aggregate the data history for individual patients and make simple diagnoses,
- and discover the gateways to an infrastructure network (if any), so that data queries from the infrastructure network

towards the patient sensors and the answers to the queries could be routed to destination.

In the example above, the ABSN discovery protocol allows low-latency data aggregation within the bounds of a patient's sensor bundle, by locally caching on sensors the neighbouring of same-activity routes and services. For the same purpose, it uses the knowledge about the patient sensor bundles being relatively localized, in order to limit the packet broadcast overhead in the network. This is different from the majority of the ad hoc protocols in the literature – which do not differentiate between nodes in the network – in the fact that ABSN allows the application-level logic to be used at the networking layer, in order to make the network protocols more applicable and efficient in practice.

An ABSN is an ad hoc network that has no reliance on any infrastructure; in addition, our design considers gateways (and other such network points of presence) as simple services, and treats them in the same fashion as other resources, such as the sensing capabilities of the network nodes.

The network topology that ABSN expects to cover is the typical case encountered in home, organization and accident settings:

- The sensors are often deployed in dense, relatively localized and connected patches following a person's or object's location.
- There is a logical structuring of the sensors based on the activity they are a part of; we call this logical grouping an *activity cluster*, AC; a typical activity cluster is formed by the set of body sensors on a patient or by the sensors monitoring the number of people present in a ward room.
- Interaction among sensors is, more often than not, bounded inside an activity cluster, but network-wide discovery and data exchange is of no less importance.
- There is a high degree of mobility involving entire activity clusters at a time and sensor unavailability is often a problem (in the example above, sensors might fall off patients while the patients are being moved, and the protocol is required to signal the change to the application layer).

ABSN answers these requirements with a completely distributed discovery scheme, that relies on no directories, but on limited, individual caching of routes and service information on nodes. It employs proactive route and resource discovery within an activity cluster, ensuring low latency for all intra-AC communication; to keep network overhead down, discovery outside an AC is reactive.

As is the case in sensor networks, separate software layers for routing and resource discovery are redundant (as argued by [21], [13]). This has led us to follow EZRP's way ([21]) in embedding routing and discovery into one protocol.

We state that, despite the extensive research in ad hoc routing and the very limited research in service discovery in sensor-like networks, it is the binding of such low-level protocols to high-level human activity concepts that makes sensor networks protocols more applicable in pervasive health-

care environments than generic routing or service discovery protocols

Our main contribution is thus the porting of ABC concepts [2] into pervasive sensor networks for the purpose of redesigning classical networking protocols to make sensors fit for usage in human-centered pervasive settings. Also, to the best of our knowledge, this is the first attempt to practically employ such low-resourced embedded systems like sensors in the field of service discovery in ad hoc environments.

We implemented ABSN on Moteiv's Tmote Sky, TelosB sensors, and tested the protocol on a real-code simulator.

The rest of this paper is organized as follows: section II gives an overview of other service discovery protocols for ad hoc and transient networks, section III gives a detailed description of ABSN, while section IV makes an analysis of the performance parameters of discovery in ABSN. At last, section V gives the evaluation results and section VI summarizes and concludes.

## II. RELATED WORK

In general, for any networked environment, device mobility implies losing connectivity with configured environments. *Service discovery* software then enables a mobile user to take advantage of resources at any new location. For pervasive environments, in which autonomous computing devices interact to achieve intelligence, service discovery protocols permit the adaptation of devices to network composition and context.

“Classical” service discovery protocols for pervasive environments - like Jini, UPnP and others - are tailored for resource-rich, stable-network enterprise-like environments, and as such are not always applicable in pervasive computing, yet they do provide basic protocol design reference points. They rely on stable, LAN-like network connections, do not need to solve network-layer routing issues (since they use IP), are usually centralized - using one or more directories spanning the network - and can employ sophisticated service semantics.

Unlike LAN devices, sensors form unstable ad hoc network connections and are used for extremely mobile applications. Such Mobile Ad hoc Networks (MANETs) are autonomous systems of intermittent, energy-bounded nodes collaborating in the absence of any centralized support. The network diameter is large compared to a node's range, any data transport is multihop, and the nodes do not have a priori knowledge of the topology of the network. Because of the network diameter-to-wireless range ratio, routing and power efficiency are central issues, and each node has to act as a router.

Thus, when trying to port “classical” service discovery protocols to MANETs, the problems arising are that in MANETs mobility and resource limitations disallow static directories, the underlying network is not stable enough to allow centralized, registration-oriented protocols and the protocols cannot be heavyweight, with respect to bandwidth and power usage.

When designing both routing and discovery protocols for large-scale ad hoc sensor networks, important decisions to take include the discovery scheme (be it proactive, reactive

or a combination of these), the network range of the protocol (multihop or singlehop), and the schemes to achieve power efficiency (by putting nodes to sleep, or just by the optimization of the computation and transmission/reception times).

#### A. Service Discovery in Ad Hoc Environments

A review of service discovery protocols for transient environments is to be found in Table I. Although most protocols are intended for ad hoc, but still resource-rich wireless networks, and the others are only theoretical, simulated proposals, both categories have provided good hints and test results that helped the design of the sensor network protocol in ABSN.

*DEAPspace* [14] redesigns the proactive approach of the 802.11-related wireless LAN protocols, so that instead of having the access points each broadcasting a beacon periodically, they each simply cache all the beacons they hear and then take turns in broadcasting a list of all the network's services (just like gossiping protocols). This decreases the transmitting frequency, while increasing the size of the transmission, thus giving devices longer idle intervals and consuming less bandwidth with cutting down on transmitted headers. The singlehop target network and the bare unsophistication of this protocol simply make it unuseful for large sensor networks.

*SANDMAN* [17] is the only approach other than [14] to design for discovery protocols centered around energy-efficiency. It is based on the observation that, although mobile, there exist groups of devices with similar mobility patterns. These will form a cluster with a dynamically reelected cluster head, that will answer service requests on behalf of the cluster nodes, which can sleep if service requests are not present. In *SANDMAN*'s case protocol functionality comes secondary to the efficiency of power consumption, and discovery in large networks is not of main interest. ABSN chooses the opposite approach, targeting at protocol functionality first.

Among the heavyweight IP-based protocols, Cheng's protocol [4] experiments with using On-Demand Multicast Routing Protocol over IP for service discovery, with multicast groups being formed by a service and all its possible clients. The protocol implements both a push variant (with services advertising themselves in multicast packets, so that clients can explicitly join) and a pull variant (that has clients query a well-known service query multicast address).

In the same category, *Konark* [9] and *Sailhan's* protocol [16] take opposite approaches. The former chooses a completely distributed, but unscalable approach, having the service providers advertise their services across the whole multihop network. The latter upgrades a set of well-positioned service providers in the network to the role of directories, which discover the services in their vicinity and actively advertise them to the other directories. Then, clients only need to query the local directory which will either solve it locally, or will ask the other directories.

While the push variant in [4] suffers from the overhead of multicast group joining for every used service, and the pull variant can be costly in large environments, the use of multicast groups for grouping services and clients can be of

future interest. For scalability and limited resources reasons, neither of [9], [16] can be ported in sensor networks.

*Allia* [15] finds an original solution for service discovery in mobile e-commerce: transitory participants advertise local services in their vicinity, the advertisements are forwarded within a certain hop diameter and the devices which hear these might passively cache them. The set of devices that a certain node chooses to cache services of forms this node's *alliance*; a request will go to other alliances only if a service is not found in the cache, and a node only knows the participants of its own alliance, but is blind as to which other alliances it might be considered a member of. The hidden intelligence of the protocol lies in its set of policies (based on application preferences and context) that rule the frequency of advertisements, whether or not an advertisement is cached and the diameter of the alliance. While still being IP-based and too heavyweight a protocol for our purposes, *Allia* was a solution to follow in our design, both because it recognizes the optimal hybrid proactive/reactive approach for discovery, and because it employs application-level policies over the discovery protocol.

Among the lightweight ad hoc discovery protocols, *GSD* [3] adapts the sophistication of [15] to use in actual MANETs, by replacing policy-based alliances with broadcast vicinities of a certain diameter, so that the network is now uniform, a situation common in generic ad hoc networks. *CARD* [10] keeps with the hybrid approach of proactive local and reactive remote discovery, but is best-effort and only valid as a routing protocol for short flows of data, trading off the optimality of the shortest path method for heavy savings in certain settings. It employs contact nodes instead of ZRP's [8] bordercast, and concentrates on the effort of electing and maintaining the contact nodes, while not placing importance on the use of vicinity information. The network uniformity in [3] and the limited validity of [10] distances both protocols from our approach: ABSN is intended to make efficient use of a logical structuring of the network, and to be a general service discovery protocol for pervasive environments.

*EZRP* [21] recognizes that, in an ad hoc network, service availability is tightly linked to route determination to that service: finding the address of a service is redundantly followed by finding a route to that address. Because of this, the solution employs the piggybacking of service information in routing packets (an idea first found in [13]). It extends ZRP [8], to include service information and has been the basis ABSN was built on, as detailed in section III.

### III. ABSN DESIGN

This section gives a detailed description of both Zone Routing Protocol (ZRP, [8]) and Extended Zone Routing Protocol (EZRP, [21]) serving as basis for ABSN.

#### A. ZRP and EZRP

ZRP is a MANET routing protocol whose guidelines fit a sensor network's setting by being flat, fully distributed and only limitedly proactive. While the advantage of proactive

|                       | Network topology and transport protocol                | Storage of service info   | Discovery policy   | Software layer                           | Implementation and/or testing  |
|-----------------------|--|---|--|--|--|
| DEAPspace (2001) [14] | Singlehop, short range 802.11-like transient network   | Fully decentralized: all nodes cache all the services available in the network            | Proactive: each node advertises all services in the network (gossip-like)                            | -  | Simulation   |
| Allia (2002) [15]     | IP network over Bluetooth                              | Fully decentralized: each node might cache the services in its vicinity based on a policy | Proactive: a node advertises its services and is included in alliances; reactive for remote services | IP middleware                            | Applied in IP mobile commerce on laptops and iPAQs, over Bluetooth (footprint: ~600k, heap: ~800k) |
| GSD (2002) [3]        | As [15]  | As [15]   | As [15] w/o alliances  | -  | Simulation (Glomosim)  |
| Cheng's (2002) [4]    | Multicast IP network, independent on lower layers      | None, discovery is on-demand; only caching of already discovered services                 | Mainly reactive, with a variant of proactive updated services advertises                             | IP routing, piggybacked on ODMRP         | -  |
| Konark (2003) [9]     | Multicast IP network, independent on the network layer | Fully decentralized: all nodes in the network cache all services                          | Both proactive (services advertise themselves) and reactive  | IP middleware                            | Applied in IP mobile commerce, using HTTP micro-servers on iPAQs and phones                        |
| SANDMAN (2004) [17]   | Any, only a model                                      | Decentralized is nodes awake; centralized, with cluster heads, if nodes sleep             | Reactive   | -  | Simulation (ns-2)  |
| Sailhan's (2005) [16] | IP network, independent on lower layers                | Centralized, distributed: dynamically elected directories form a network backbone         | Proactive among directories inside a node's zone, reactive outside it                                | -  | Simulation   |
| EZRP (2005) [21]      | Multihop, sensor-like network                          | Fully decentralized: all nodes cache services in their zone                               | Proactive inside a node's zone, reactive outside it  | Routing, embedded in ZRP                 | Simulation (Qualnet)   |
| CARD (2005) [10]      | Multihop, sensor-like network                          | Fully decentralized: all nodes cache services in their vicinity                           | Proactive inside a node's zone, reactive outside it  | Routing (not a general routing protocol) | Simulation (ns-2)  |

TABLE I  
A REVIEW OVER SERVICE DISCOVERY PROTOCOLS IN AD HOC NETWORKS

route discovery schemes is that routes within the network are continuously refreshed - so that a query for a route is answered without latency - they also constantly add overhead on network bandwidth and on a node's cache memory. Reactive protocols do not determine routes before an explicit query, and require a global flood search procedure with long delays and heavy-weight traffic at each query.

ZRP chooses a hybrid proactive and reactive approach and delimits a zone of a certain number of hops around each node (so that, overall, the zones are heavily overlapped), and limits the proactive procedure to this zone. For out-of-zone discovery, instead of plainly broadcasting of the query throughout the network, the *bordercast* message distribution scheme directs queries from a source node towards the edges of the network by only forwarding the query to the nodes on the border of the source node's zone. It is important to note that, despite grouping the nodes into zones, ZRP is not a hierarchical protocol, but remains a flat one, since each node has a zone, so that the grain size of this zoning is one node.

The ZRP and its subprotocols' IETF drafts do not impose specific protocols for the proactive and reactive discovery of routes, and they even give guidelines specific to routing over the heavyweight IP layer. We choose to adapt these drafts' guidelines for use in resource-poor sensor networks, as described in subsection III-B.

EZRP extends ZRP for use in service discovery, simply by adding a service ID to the hello messages used by neighbours to let them know each other. This way, neighbouring nodes find the others' service IDs, together with their simple presence. Furthermore, nodes periodically broadcast their set of neighbours and their service IDs throughout their zones, so that intra-zone routes are extended with service information.

A view over an EZRP network is given in figure 1. We will denote the ZRP zones by the term *network cluster*, NC, for easily relating them to the ACs. In figure 1, if a link-state protocol is used intra-NC (for example, a simplified OSPF), node A will receive hello packets from its direct neighbours (also containing a service ID field) and will periodically transmit advertisements throughout the NC, announcing its list of neighbours and their services. When node A's NC converges (all nodes have a consistent view upon the network, from a routing point of view), A will have in its local cache a complete view over its NC, route- and service-wise. In order for A to discover services and routes for nodes D or X, bordercast is employed.

### B. ABSN Discovery Design

As stated in section I, the reflection of high-level activities into the sensor network is the logical grouping of the sensors into *activity clusters*, AC, which are deployed in multihop,

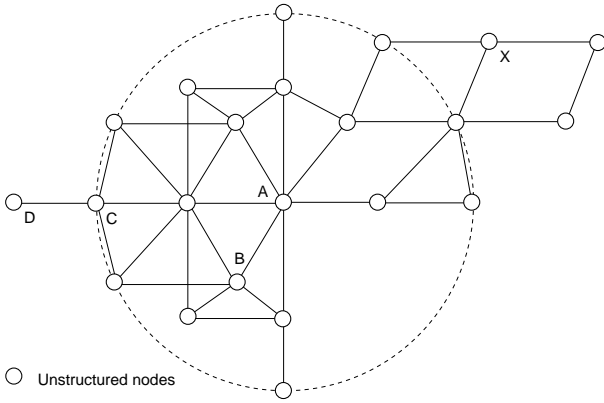


Fig. 1. A view over an EZRP discovery network: the nodes are not logically grouped and each node keeps a zone (NC) of a 2-hop radius (for this example) centered at itself. The NC for node A is drawn.

overlapping patches throughout the network. Since data communication is more often than not bound inside an AC, we call for a proactive discovery scheme for intra-AC routes and services and a reactive discovery scheme inter-AC.

Although this might immediately recall EZRP (subsection III-A) as a solution, there is no possibility of identifying ACs with the network clusters in ZRP in an one-to-one fashion. In ZRP, NCs are sets of nodes reachable within a certain radius (number of hops) from any central node and are flatly distributed across the network (for every node in the network there is a NC, and nodes are logically homogeneous). On the other hand, ACs are unique sets of nodes (for example, there is only one AC with patient Hansen's ID) and are deployed in irregular, possibly overlapping and dense geometric patterns around the network.

A view over a basic activity-based network is given in figure 2. Activity clusters are deployed in a relatively localized, connected fashion throughout the network; they might overlap in the same area, so that any node might have in its range other nodes belonging to activities different than its own.

The main requirements imposed over the ABSN design are protocol scalability in large networks and low latency at intra-AC discovery: service and route discovery is expected to work throughout the network, yet optimized in such a way that discovery latency is low if the requestor and the service belong to the same activity.

ABSN superimposes the logical AC grouping over the flat EZRP network topology, as in figure 3. To achieve scalability, every node in the network still keeps a NC zone centered at itself, in order to limit the proactiveness of the protocol. Since the nodes within this NC belong to different ACs, and since the logical grouping of sensors in ACs implies a lower probability that a service be requested from a node of a different colour, ABSN has a node only cache the service information of same-AC nodes that lie within this node's NC. Furthermore, a node will proactively cache routing information for all the nodes within its NC.

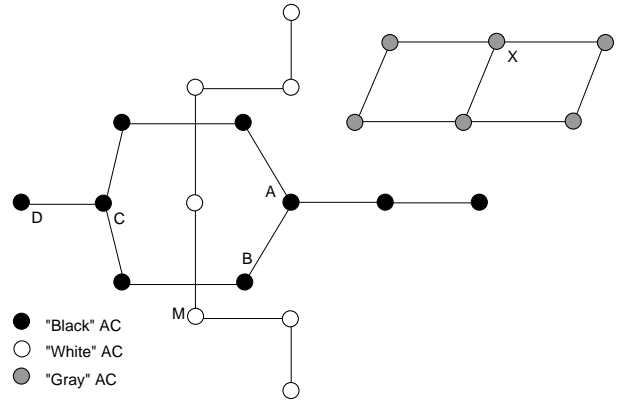


Fig. 2. The logical view over an activity-based network: each activity is colour-coded and the intra-AC connections are drawn; connectivity also exists among neighbouring nodes of the different, overlapping ACs (not drawn in the figure).

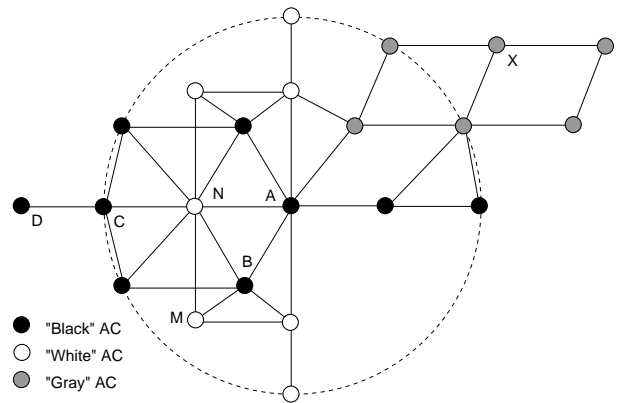


Fig. 3. A view over a sample ABSN network: in the EZRP fashion, every node, regardless of its AC, keeps a zone (NC) of a 2-hop radius (for this example) centered at itself. The NC for node A is drawn, and it gathers nodes from different ACs, including node A's own AC.

### C. Intra-NC Discovery

A lightweight link-state proactive protocol is employed intra-NC. This protocol sends periodic one-hop broadcast hello packets from each node, so that - at all times, and with a maximum latency equal to the hello period - every node knows the addresses of its neighbours and the link quality to each of them, used as metric. Also, a node that detects a drastic change in the metric to at least one of its direct neighbours (including a neighbour's arrival or disappearance) triggers a limited-range *link-state advertisement* (LSA), announcing the change within the radius of its own NC. LSAs are forwarded in a broadcast manner for only a number of hops equal to the radius of the NC.

This way, every node keeps a *routing map* of the NC as a weighted graph of nodes that can be reached in a number of hops equal to or less than the NC diameter.

In order to allow nodes to build a *service map* of the near AC, both hello and LSA packets carry, besides addresses, AC and metric information, the service IDs of the nodes these

packets advertise: they will both state the service ID of the source of the packet, and - in addition - a LSA will list the neighbours which changed state and, if available, their service information.

At a “black” node (see figure 3), only such incoming packets bearing service IDs for other “black” nodes have their service information cached. In figure 3, this leads to the “black” node A having a service table with the service information of all the “black” nodes in its NC: a hello packet originated at node B advertised the service ID of B, and a LSA triggered by node C (and forwarded over one “white” hop) advertised the service ID of C.

To save computation time and memory, no service information is cached about nodes of other colours.

#### D. Route and Service Query Solving

With a routing map and a service map in place at each node, the solving of a service query depends on the parameters of the query. A service of the same colour might be needed, as in the case of the aggregation of data from sensors belonging to the same patient, for automatic diagnosis of the patient. If the gateways to the wired network form an activity cluster with a well-known colour, then the downloading of sensor data from any patient sensor to the wired network will have to be preceded by the (different-AC) discovery of the gateway location.

Given these two cases, if a certain service of the same colour is needed, then

- the local service table is tried; if there is a match, then the service owner’s address is returned;
- if there is no match, the query is bordercast only to the border nodes (or, if none exists, the closest nodes to the border) of the same colour, exploiting the fact that ACs are connected, and thus greatly limiting the network overhead of the query.

If, on the other hand, a service of another colour is needed, then

- the local routing table is checked for the closest node of the searched colour, again relying on the fact that same-colour nodes are more likely to be closer to each other; if such a node (which we call a *gateway* for the entire searched AC) is found, the query is relayed to it, and it will proceed as in the same-colour case above; if more than one gateway is found, ABSN chooses the closest gateway to the query source node;
- if no such gateway exists, the query for the gateway is bordercast to all border nodes

A routing query will be solved exactly as in ZRP, regardless of activity IDs. Routes for all the nodes that can be reached within a number of hops equal to the NC diameter are read from the proactively built network map. Routes for any other nodes are discovered on-demand, through bordercasting.

This design makes discovery either proactive or reactive, depending on whether the unknown is a route or a service, and on the relation between the AC of the requestor and that of the destination:

- Route discovery is always proactive within the limits of a NC, and reactive outside, regardless of the relation between the colours of the nodes involved.
- Service discovery for same-colour nodes sharing a NC is fully proactive; in all other colour and distance conditions, service discovery is always a mix of proactiveness and reactivity: the query is initially forwarded for certain other nodes to solve, but will eventually reach a node that has discovered the required service information in advance.

#### IV. DISCOVERABILITY, OPTIMALITY AND OVERHEAD ANALYSIS

This section analyzes the performance of ABSN compared to the relevant related work protocols. It defines and uses parameters such as the *discoverability* of that service, the *optimality* and latency of the solving of a query and the network overhead added by a query to assess ABSN’s characteristics. Furthermore, this section analyzes the set of network topologies that ABSN serves better than other protocols.

The ability of ABSN to discover any service (including a route), if present in the network, is hence called the *discoverability* of that service. ABSN guarantees discoverability in all network settings, provided that any activity cluster is connected; this is like [21] and [3], and unlike [10].

We denote by the *optimality* of a query that query taking the shortest, lowest-latency route in the network. The optimality of ABSN depends on the NC radius and on the network topology of the searched AC. While having low discovery latency in a large number of network and activity topologies, ABSN only has suboptimal latency in few badly-formed activity cluster topologies.

As a routing protocol, ABSN is optimal and guarantees route discoverability, in the fashion of ZRP. The ABSN routing and discovery design following logical activity grouping greatly limits the network and computational overhead that a general-purpose EZRP would imply, if deployed in the pervasive environments ABSN is applicable to.

These statements will be substantiated by the discussion over local and global discovery in subsections IV-A and IV-B.

##### A. Neighbourhood Discovery

In regard to route discovery within a node’s NC, it is to note that the proactiveness of the lightweight link-state protocol that ABSN uses ensures that changes in the network or in the service provision are propagated in a timely fashion, with a maximum latency directly proportional to the NC radius. Since link-state protocols keep a dynamically updated view over the entire NC and are immune to routing loops, optimality and route discoverability at intra-NC route discovery are ensured.

On the other hand, link-state protocols broadcast any LSA throughout the NC. This implies that the radius of each node’s NC must be dynamically updated given the node density in the area, in order to keep network and computation overhead down.

In regard to service discovery, any node will announce throughout the NC any change in the states of its direct neighbours by only mentioning known service information: for example, in figure 3, when “black” node C enters the network, “white” node N will not cache C’s service information provided by C’s hellos. Thus, a LSA originated from N and announcing the new “black” node will not provide to A the service information that A could use (since A and C are of the same colour). We call the situation in which the forwarding of service information towards a node is stopped by interposing nodes of another colour by the term “colour shadowing”.

However, within a NC, ABSN is resilient to such apparent “colour shadowing”: even if the “black” AC weren’t connected, the new node C will also trigger a LSA, announcing its new link to node N. Since LSAs are forwarded regardless of the colour of their source and contain the complete service information of their source node, node A will receive a first-hand update about the services provided by C from C itself. In general, within an NC, every node will have a complete image of the same-colour services in the neighbourhood. This resilience is even independent of AC connectivity. Furthermore, still in the example in figure 3, if the “black” AC is connected, LSAs with the service information of the new node will redundantly reach node A on all fully “black” paths from C to A. This way, discoverability of same-colour services within an NC is guaranteed and is optimal - this adds to the optimality of the routing of packets intra-NC.

In the case of different-colour service discovery, on the other hand, while it is ensured that a service will be discovered if present (the service discoverability is also fully guaranteed), the optimality of the query depends on the choice of gateway for the searched AC. Yet, in the worst-case scenario in which a gateway is blindly chosen in the exactly opposite direction from the actual service searched, a maximum of 1 bordercast step will have the query solved (as shown in the example in figure 4).

### B. Global Discovery

At a global level, route discoverability and optimality are secured, by the design of ZRP. An average maximum number of  $\frac{D}{R}$  bordercast steps are employed for the route discovery towards a node which is  $D$  hops away, if the average NC radius of the nodes on the path is  $R$ .

ABSN greatly reduces network overhead by only forwarding service queries for a “black” service within the bounds of the “black” AC. In the case of symmetrical, radially deployed ACs (as in figure 5), the network overhead is reduced to a percentage of  $\frac{\text{black network area}}{\text{total network area}}$  of the network overhead in EZRP (here, we denote by the *area* of an AC or a network a qualitative measure that is proportional to the number of nodes in that AC, the nodes’ degree and to the number of hops this AC occupies).

Service discoverability in ABSN relies on the activity clusters being connected. If ACs are connected, ABSN guarantees the discoverability and optimality of a different-AC gateway node search, but only the discoverability of a service is

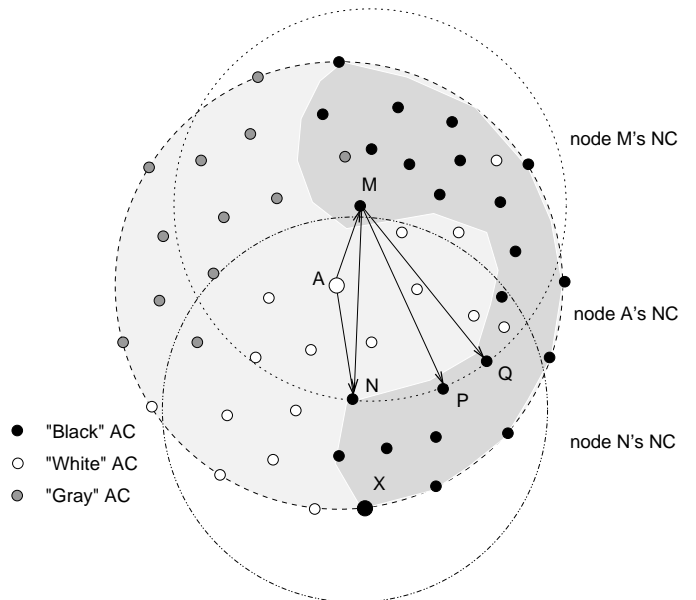


Fig. 4. For “white” node A to discover the services of “black” node X, A needs to choose a gateway for the “black” AC. The ABSN solution is to choose the best metric (closest) “black” node as a gateway (in this figure’s case, node M). Yet, for M to discover X it needs to do one bordercast step (forwarding the query to N, P and Q). An optimal choice for a “black” gateway would be node N, which can immediately answer it without latency, since X lies in N’s NC.

guaranteed: the optimality of the path the query takes towards the service depends on the topology of the searched AC: at a global level, “colour shadowing” is in effect, routing a query according to AC topology.

For example, in figure 6 a service query takes a sub-optimal path through the overall network. ABSN argues that this is a small price to pay for the reduction in network traffic. It is to note that this sub-optimality is only true for service requests: service replies are treated as regular data packets and, since routing is optimal, the replies take the optimal path back.

Furthermore, as shown in subsection IV-A, within the limits of a NC, “colour shadowing” does not limit discoverability. This indicates that global “shadows” or disconnections less than the NC radius in width will be overcome by the protocol. In order for the protocol to also cover the general case when ACs are widely disconnected, it only needs to be added a *join* mechanism so that disconnected subsets of one AC can be forwarded data over differently-coloured nodes. Given this, it follows that the preferred AC topologies only exclude disconnections wider than the NC radius.

## V. EVALUATION

This section gives an overview of the means of evaluating ABSN. Real-code simulated tests were performed to show the improvements in terms of network and computational overhead of ABSN over EZRP, as well as the scalability of ABSN.

ABSN was implemented on Moteiv’s IEEE 802.15.4-compatible *Tmote skies*, as a set of TinyOS [19] NesC [6] components. NesC and TinyOS (the de facto standard for

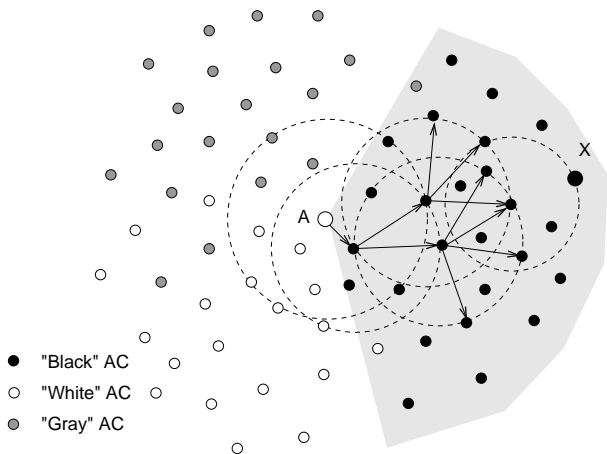


Fig. 5. For node A to discover the services of “black” X, a service query will be bordercast repeatedly. Unlike EZRP, which would bordercast the query symmetrically around A, ABSN limits the traffic to the bounds of the “black” AC.

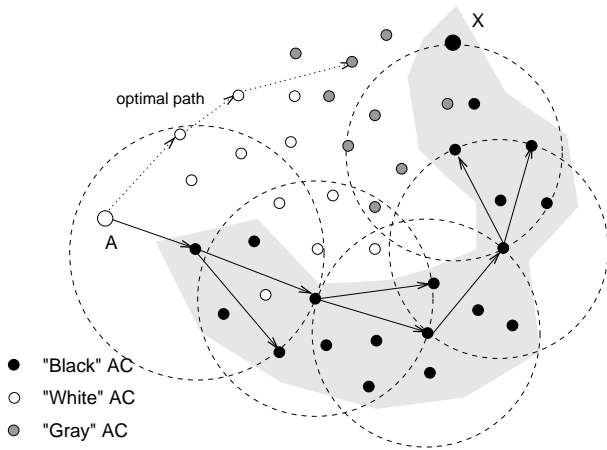


Fig. 6. The bordercasting of the “black” query will follow the deployed shape of the “black” AC, instead of taking an optimal path across other ACs.

programming low-resource sensor networks) were used in order to ground ABSN in practice.

For simulation results that would be a correct reflection of the running of the code on sensors, we chose a *real-code* simulator composed of two pieces of software: OMNeT++ [18], a general networking discrete-event simulation environment, and NesCT [12], a language translator from the embedded sensor language NesC into OMNeT++’s C++ classes. NesCT already comes with the translation for OMNeT++ of the basic TinyOS components. Thus, the simulations take into consideration all of TinyOS’s features and limitations.

Since ABSN extends EZRP by adapting its basic functionality for use in pervasive activity-based settings, a set of simulations show the decrease in the network overhead triggered by one service query, while being multiplied throughout the network in search of the destination node. Figures 7 and 8 show the network traffic - due to the multiplication of the single service query - as a comparison of two cases in

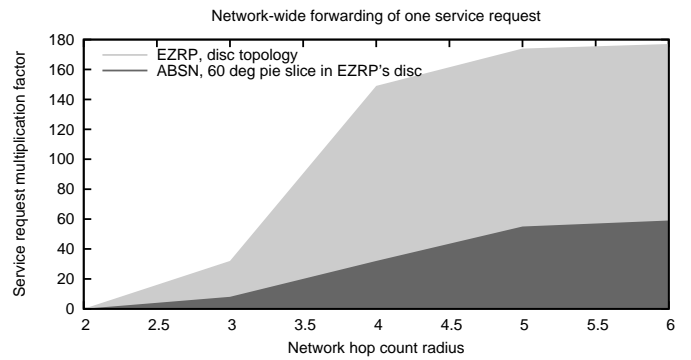


Fig. 7. Additional network traffic due to the multiplication of a single service query, in number of packets by the radius of the overall network. The network is a 3-degree extended star with varying radius; all nodes keep NCs of 2-hop radius; the query is intra-AC: it starts at the central node and is destined to a same-colour border node. The searched AC is a non-overlapping, 60-degrees pie slice in the network.

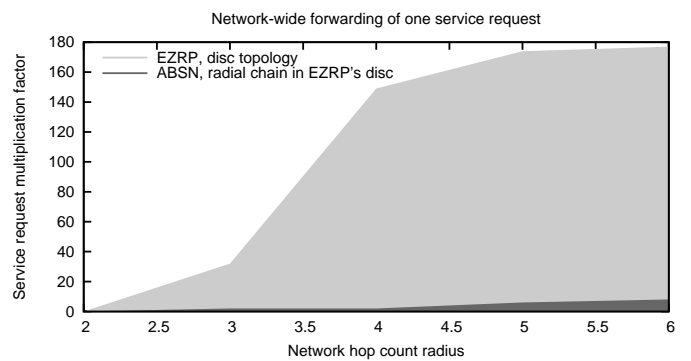


Fig. 8. Additional network traffic due to the multiplication of a single service query, in number of packets by the radius of the overall network. The network is made up by 3-degree nodes and is an extended star of varying radius; all nodes keep NCs of 2-hop radius; the query is intra-AC: it starts at the central node and is destined to a same-colour border node linked by a chain AC.

which the service discovery is being performed by EZRP and by ABSN, in the same network. The two sets of tests are performed over extended star networks of varying hopcount radius, composed of nodes with a constant degree equal to 3 and having NCs of constant radius 2. The source of the service query is the central node in the network, the searched service lies on the border of the network, and the two nodes have the same colour.

Two ACs occupy the network: in the case of figure 7, the AC including the source and the destination of the query occupies a non-overlapping 60-degrees pie slice in the overall network. The case in figure 8 is extreme, in that which that AC is a chain of nodes running from the central node towards the border destination node.

In figure 7, it can be noted that ABSN uses only a fraction (approximately equal to the pie slice proportion in the 2D network) of the traffic generated by EZRP in order to find the service. Figure 8 is a good example of extreme ABSN efficiency: only a number of packets that is linear to the distance from source to service is forwarded on the network, compared to EZRP’s storm of LSAs.

In the above cases, discovery latency is equal for both EZRP and ABSN, yet ABSN shows a great improvement in network bandwidth usage, by guiding the query along the path of an AC, instead of unconditionally bordercasting it through the network. While these tested topologies do not make a complete usage test, they do give a starting point for extrapolating activity-based network topologies in which ABSN is much more efficient than EZRP.

Furthermore, the tests above only evaluate same-AC queries, and show that intra-AC discovery is much more efficient in terms of use of bandwidth, while only yielding on speed in certain topologies. The more general case of queries originated in one AC and destined to another has a network usage efficiency that is a combination of the performance of EZRP and the efficiency of intra-AC discovery: the discovery of a gateway for the searched AC performs like EZRP, while the discovery of the service starting at the already found gateway performs like the tests above.

One other important test relates to the scalability of the protocol; as stated in subsection IV-A, since ABSN is a flat protocol, the main measure of scalability is the node size of the network clusters. While a large network cluster would improve response times by adding more proactiveness to the overall network, a limit over the NC size is imposed by resource bounds on the nodes and by limited network bandwidth.

Figure 9 shows the network traffic generated by the intra-NC protocol within the bounds of randomly generated, densely connected NCs in extended star topologies. The figure shows both the maintenance traffic (hello packets are sent by each node even in the absence of change) and network building traffic (LSA packets are sent in the process of building the NC, one incoming node at a time). The NC radius is kept constant at 2 hops, while the size of the NC varies to also model the node density. At the same time, the degree of all nodes (the number of links to neighbouring nodes) is kept proportional (by a constant factor) to the NC size. To model a possible real-world case, a number of 3 ACs overlap in this NC, and, at every network change, each node sensing the change triggers, on average,  $\frac{2}{3}$  of a LSA packet (a number given by the relation between the default TinyOS packet size and the size of node addresses and service identification fields).

The number of hello packets that are sent on the network only varies slightly below 10 times the size of NC, while the number of LSAs varies significantly with the NC size. This LSA overhead variation is recognizable as being proportional to the square of the NC size, which follows the theory: the forwarding of LSAs (a constant number of such LSAs originate from each node) would trigger their multiplication by a factor proportional to  $N * E$ , where  $N$  is the number of nodes and  $E$  is the number of edges in the network graph (equal to  $\frac{\rho * N}{2}$ , where  $\rho$  is the average degree of the nodes). This gives a multiplication factor proportional to  $N^2$ . Any variation in the test parameters (the number of overlapping ACs and the fraction of one LSA that any network change would trigger) would only change the multiplication factor of forwarded LSAs by a constant.

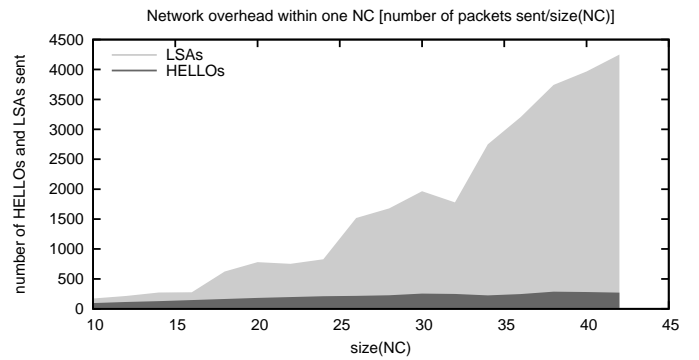


Fig. 9. Network traffic, in number of packets by number of nodes composing the NC. The hello packets have been sent out within the bounds of one NC by the link-state proactive intra-NC protocol during a period of time equal to 10 times the hello period. The LSA packets have been sent in the initial process of setting up the same network.

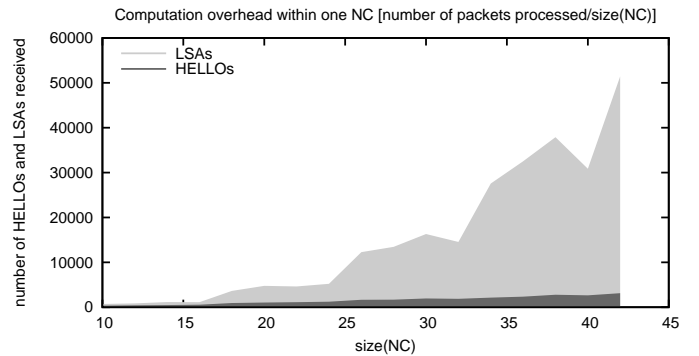


Fig. 10. Number of receive packet events, summed up for all the nodes in the NC. The hello packets have been sent out within the bounds of one NC by the link-state proactive intra-NC protocol during a period of time equal to 10 times the hello period. The LSA packets have been sent in the initial process of setting up the same network.

In the same set of experiments, figure 10 shows the total computational overhead added to the nodes in the network because of the network traffic in figure 9, summed up for all the nodes in the network. Only those received LSAs that announce a change to a node for the first time will actually be processed (a number equal to the number of unique LSAs triggered on the network, as in figure 9), the others being duplicates or old information.

Another measure of the scalability of the protocol is the size of the data structures needed for each node to act as a router, service provider and discoverer. An overview of the RAM size consumed by these data structures on a TinyOS sensor node is given in figure 11. It is to note in the figure that the adding of service discovery capabilities to nodes running a link-state routing protocol is done at an insignificant memory cost. Thus, such proactive service discovery schemes can be a natural, low-cost extension to link-state routing protocols in practice. A *Tmote sky* module with 10kB RAM and 48kB programming flash running ABSN will take 17.750 kB of ROM and will accommodate NCs and ACs of as many as 62 nodes, while the sizing down of a NC/AC to a sufficient 50 nodes will occupy 7.077 kB of RAM.

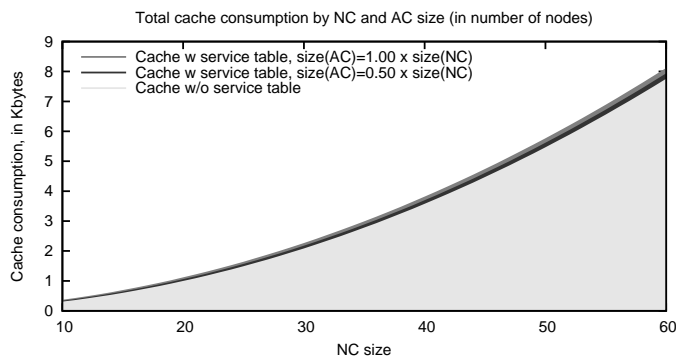


Fig. 11. An overview over RAM consumption on a sensor node running ABSN. Memory consumption is static in TinyOS (there is no dynamic allocation), so that total occupied size can be found at build time. The graphic leaves out the footprint of the application (which is constant with the NC size) and only considers the size of all data structures involved in routing and service discovery.

## VI. CONCLUSIONS

To summarize our contribution, ABSN is part of an effort to redesign classical network protocols for a better applicability in pervasive computing: it uses the high-level concept of *computational activities* (as logical bundles of data and resources) to give sensors knowledge about their usage even at the network layer and it makes use of this logical structuring of the network for a more effective service discovery scheme. Noting that in practical settings activity-based sensor patches are localized, ABSN designs a completely distributed, hybrid discovery protocol which is proactive in a neighbourhood zone and reactive outside, tailored so that any query among the sensors of one activity is routed through the network with minimum overhead, guided by the bounds of that activity. ABSN enhances the general-purpose *Extended Zone Routing Protocol* with logical sensor grouping and greatly lowers network overhead during the process of discovery, while keeping discovery latency close to optimal. Future work includes generalizing ABSN for use in networks with arbitrarily disconnected activity clusters.

Furthermore, sensor networks are characterized by a need to carefully integrate functionalities in order to achieve maximum efficiency (especially with respect to energy consumption). ABSN follows this idea and introduces a close interaction of research from ad hoc networking and human-centered pervasive computing.

Even though our design for ABSN data routing and service discovery emerges from the medical domain, we think that the approach is more generally applicable - there exists a range of other application domains where it is practical to model human activities and use this modeling for data routing and service discovery in a sensor network: fields of application like smart homes or office spaces and ad hoc peer-to-peer connectivity in public places could benefit by the idea of moving some application features into the lower routing layer.

## REFERENCES

- [1] O. Aziz, B. Lo, Guang-Zhong Yang, R. King, and A. Darzi. Pervasive Body Sensor Network: An Approach to Monitoring the Post-operative Surgical Patient. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks, 2006 (BSN 2006)*, pages 13–18. IEEE Press, 2006.
- [2] Jakob E. Bardram. Activity-Based Computing: Support for Mobility and Collaboration in Ubiquitous Computing. *Personal and Ubiquitous Computing*, 9(5):312–322, July 2005.
- [3] D. Chakraborty, A. Joshi, Y. Yesha, and T. Finin. GSD: A Novel Group-based Service Discovery Protocol for MANETS. *Mobile and Wireless Communication Networks*, 2002.
- [4] Liang Cheng. Service Advertisement and Discovery in Mobile Ad hoc Networks. *Computer Supported Cooperative Work workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments*, 2002.
- [5] Division of Engineering and Applied Sciences of Harvard University. CodeBlue: Wireless Sensor Networks for Medical Care. <http://www.eecs.harvard.edu/mdw/proj/codeblue/>.
- [6] David Gay, Philip Levis, and Robert von Behren. The nesC Language: A Holistic Approach to Networked Embedded Systems. In *Proceedings of the ACM SIGPLAN 2003 Conference on Programming Language Design and Implementation (PLDI)*, 2003.
- [7] L. Grajales and I. V. Nicolaescu. Wearable Multisensor Heart Rate Monitor. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks, 2006 (BSN 2006)*, pages 154–157. IEEE Press, 2006.
- [8] Zygmunt J. Haas, Marc R. Pearlman, and Prince Samar. The Zone Routing Protocol (ZRP) for Ad hoc Networks. *IETF MANET Internet Draft*, 2002.
- [9] S. Helal, N. Desai, V. Verma, and C. Lee. Konark - A Service Discovery and Delivery Protocol for Ad-Hoc Networks. *Wireless Communications and Networking Conference*, 2003.
- [10] A. Helmy, S. Garg, N. Nahata, and P. Pamu. CARD: A Contact-based Architecture for Resource Discovery in Wireless Ad Hoc Networks. *Springer Mobile Networks and Applications*, 10:99–113, 2005.
- [11] T. Hester, R. Hughes, D. Sherrill, B. Knorr, M. Akay, and P. Bonato. Using Wearable Sensors to Measure Motor Abilities Following Stroke. In *Proceedings of the International Workshop on Wearable and Implantable Body Sensor Networks, 2006 (BSN 2006)*, pages 5–8. IEEE Press, 2006.
- [12] Omer Sinan Kaya. NesCT: A language translator. <http://nesct.sourceforge.net/>.
- [13] R. Koodli and C. E. Perkins. Service Discovery in On-Demand Ad hoc Networks. *IETF MANET Internet Draft*, 2002.
- [14] Michael Nidd. Timeliness of Service Discovery in DEAPspace. In *ICPP Workshops*, page 73, 2000.
- [15] Olga Ratsimor, Dipanjan Chakraborty, Anupam Joshi, and Timothy Finin. Allia: Alliance-Based Service Discovery for Ad-hoc Environments. In *Proceedings of the Second ACM International Workshop on Mobile Commerce (WMC-02)*, pages 1–9, New York, September 28 2002. ACM Press.
- [16] Françoise SAILHAN and Valérie ISSARNY. Scalable Service Discovery for MANET. In *PerCom*, pages 235–244. IEEE Computer Society, 2005.
- [17] Gregor Schiele, Christian Becker, and Kurt Rothermel. Energy-Efficient Cluster-based Service Discovery. In *Proceedings of the 11th ACM SIGOPS European Workshop (SIGOPSEW04)*; Leuven, Belgium, September 20-22, 2004, Artikel in Tagungsband, pages 75–79. ACM SIGOPS, September 2004.
- [18] The Open TinyOS Community. OMNeT++: Discrete Event Simulation System. <http://www.omnetpp.org/>.
- [19] The Open TinyOS Community. TinyOS. <http://www.tinyos.net/>.
- [20] The PalCom Consortium. Palpable Computing. <http://www.ist-palcom.org/examplesOfWork/accidents.php>.
- [21] Christopher N. Ververidis and George C. Polyzos. Extended ZRP: a Routing Layer Based Service Discovery Protocol for Mobile Ad Hoc Networks. In *MobiQuitous*, pages 65–72. IEEE Computer Society, 2005.