DNS-based service discovery in ad hoc networks: evaluation and improvements*

Celeste Campo and Carlos García-Rubio

Dept. de Ingeniería Telemática - Universidad Carlos III de Madrid
Escuela Politécnica Superior - 28011 Leganés (Madrid)
{celeste, cgr}@it.uc3m.es

Abstract. In wireless networks, devices must be able to dynamically discover and share services in the environment. The problem of service discovery has attracted great research interest in the last years, particularly for ad hoc networks. Recently, the IETF has proposed the use of the DNS protocol for service discovery. For ad hoc networks, the IETF works in two proposals of distributed DNS, Multicast DNS and LLMNR, that can both be used for service discovery. In this paper we describe and compare through simulation the performance of service discovery based in these two proposals of distributed DNS. We also propose four simple improvements that reduce the traffic generated, and so the power consumption, especially of the most limited, battery powered, devices. We present simulation results that show the impact of our improvements in a typical scenario.

1 Introduction

The increment in the number of devices connected to networks has motivated the development of service discovery protocols, which help the user in the task of automatically discovering and using the wide range of services available in a network (e.g. printers, mail servers, etc.). Some service discovery protocols have been defined in the IETF for the Internet (SLP [1], SSDP [2]), and others have been defined by other standardization bodies, tied to a particular high-level technology (Jini [3], Salutation [4]). More recently, other service discovery protocols, specifically designed for ad hoc networks, have been defined, some tied to a wireless technology (SDP for Bluetooth [5], IAS for IrDA [6]), others that jointly deal with the problems of ad hoc routing and service discovery (GSD [7], HSID [8]), and others that work at the application layer of the protocol stack (DEAPspace [9], Konark [10], the post-query strategies [11], and PDP [12]). For a complete review of service discovery protocols, see [13].

In their answer messages, service discovery protocols return the name of the server or servers that offer the service, together with other relevant data (e.g. transport protocol, port, service attributes, etc.). Server names are preferred

* This work has been supported by the Everyware (MCyT TIC2003-08995-C02-01) and Easy Wireless (ITEA ip03008) projects.

to IP addresses because, when there are several responses, the user is usually prompted to select one among them. The device must send a DNS query to resolve the name of the selected server into an IP address, prior to accessing the service.

Recently, the Zeroconf IETF working group has proposed the use of DNS for service discovery, so devices don't need to implement two different protocols (the DNS protocol and a service discovery protocol) but just one for both functionalities. This proposal is known as DNS-Service Discovery (DNS-SD). For ad hoc networks, where service discovery is essential, but the infrastructure necessary to support it may not be always available, DNS-SD can work over any of the two current proposals of distributed DNS for infrastructureless networks: LLMNR and Multicast DNS. In this paper we compare both proposals, evaluating their performance and particularly the traffic they generate.

In wireless networks, one of the key issues is minimizing energy consumption, since most devices are battery powered and so their autonomy is increased. Several studies about power consumption in wireless devices show that wireless communications are responsible of a significant part of the energy consumption, and that the cost of transmitting a packet is almost independent of its size and of whether it is unicast or broadcast [14, 15]. These facts must be taken into account when designing protocols for these kind of environments. In this paper, we present some simple improvements that can reduce the traffic generated in DNS based service discovery, and so the power consumed.

The paper is organized as follows. First, section 2 describes the proposals for DNS-based service discovery in ad hoc networks, Multicast DNS and LLMNR. Then, section 3 compares the performance of both proposals through a simulation study, and section 4 proposes some improvements that reduce the number of transmissions and so the power consumption. Finally, section 5 discusses some implementation issues, and, section 6 the conclusions and future work.

2 DNS-based service discovery in ad hoc networks

DNS Service Discovery (DNS-SD) [16] provides support for service discovery over DNS, without making any change to the DNS protocol. With DNS-SD, devices can obtain a list of servers offering a given service type as a response to a DNS query. At the time of writing this paper, this proposal is in Internet Draft state.

DNS-SD works over DNS, so it may use the classical centralized architecture, based on a hierarchy of servers, or any of the DNS modifications for name resolution in infrastructureless networks, Multicast DNS or LLMNR, with a fully distributed architecture.

DNS-SD exploits the syntax and the semantic of the SRV resource records for service discovery, adding one level of indirection to allow the user obtaining instances of service types with different characteristics.

DNS is a protocol that requires network infrastructure and a heavy administrative management due to how domain names are assigned and delegated.

Regardless of whether DNS is being used for service discovery or not, a solution for name resolution in infraestructureless, ad hoc, networks is necessary. Recently, two proposals have been presented in the IETF for distributed DNS, and, as we previously mentioned, they may be also used for DNS-based service discovery: Multicast DNS and Linklocal Multicast Name Resolution.

Both proposals start from the DNS protocol but do out with the centralized architecture, replacing it by a fully distributed approach in which all the devices in the network have their own DNS server, and all DNS queries are multicast. In the following subsections we will describe in detail both proposals.

2.1 Multicast DNS

Multicast DNS [17], as DNS-SD, is fruit of a joint initiative of the Zeroconf and DNSEXT groups of the IETF, with Apple Computer as the prime mover. Multicast DNS defines a new top-level domain, `.local.`. All the names under this domain have meaning only in the local network in which they have been defined. There is no naming authority in charge of managing this domain, but any user or software may create their own names with the `.local.` suffix, provided that they don't clash with names chosen by other users in the same local network.

When the resolution of a name with `.local.` suffix is requested, the Multicast DNS protocol must be used. All devices in the network must have a "Multicast DNS client" that issues multicast resolution queries, and a "Multicast DNS server" that resolves these queries.

In Multicast DNS, applications that request a name resolution can have three modes of operation: "one-shot queries", the client waits for the first response and discards the others; "one query-multiple responses", the client waits for all the answer messages; and "continuous query", in which the client issues the same query periodically, and so it monitors the existence of some resource in the network.

In order to reduce network traffic in the last two modes, queries include all the records previously known by the client (stored in its cache), so a server will answer a query only if it knows of a resource record not included in the query. To include the known records in the query, the answer section of the DNS message is used (the use of the answer section of the message in a query is illegal in classical DNS). If all the known records do not fit in a query packet, this must be signalled setting the TrunCation (TC) bit in the header of the query message, and the rest of the known records must be sent in a new query with an empty query section.

In this protocol, servers send multicast answer messages, so all devices in the network receive all of them, and this way they keep their caches updated. Moreover, this allows fast detection of clashes between domain names used by different devices. To help reducing the number of collisions in the network, servers delay their answer messages to a query a random time uniformly generated between 20–120 ms. All replies must be authoritative answers, so a server never replies with information from its cache.

In Multicast DNS, the TTL (the Time-To-Live defined in DNS) of the resource records is chosen according to how mobile the device is, and how long it will remain in the same network. So, for static devices, large TTL values are configured, and for dynamic devices, small TTL values are used. The recommended default value for the TTL is 120 seconds, which means that other devices in the network may store outdated information about us for up to two minutes. Reducing the TTL reduces the time outdated data remains in the caches when someone leaves the network, but it increases the network traffic.

To reduce the number of stale entries in the cache, and so the number of false service discoveries, Multicast DNS introduces three mechanisms:

– The "goodbye" message: it is used when a server detects that it is about to leave the network or to shutdown. This message consists in a gratuitous answer message (i.e., an answer that do not correspond to any query) in which the device includes all its local resource records (services) with TTL value of zero sec. This way, all devices listening the goodbye message, will delete these records from their caches.
– Update entries: if there is a change in any resource record (e.g., a device changes the characteristics of a service it was offering), the server sends a gratuitous answer message with the updated resource records.
– Remove entries in the local cache: when a failure is detected using the information from a resource record in the cache (e.g., the service does not respond), or a change in the topology of the network is detected, the involved resource resords are removed from the cache.

2.2 Linklocal Multicast Name Resolution

Linklocal Multicast Name Resolution (LLMNR) [18] is an initiative from the DNSEXT group of the IETF, with Microsoft as the prime mover. Its way of approaching the problem of name resolution in ad hoc networks is much more conservative than Multicast DNS, with no modification in the use DNS message fields, and without defining any new domain name for the local scope.

In LLMNR, devices have a "LLMNR client", which sends name resolution queries, and a "LLMNR server", which answers the queries made by the clients.

LLMNR clients transmit their queries using multicast, and wait for answer messages to arrive. Servers which have one or more authoritative resource records that match the query, reply using unicast. Information from the caches cannot be included in the replies. LLMNR is more restrictive than DNS regarding the definition of authoritative zones. In DNS, the authoritative zone of a server comprises all the domain names in the sub-tree under its start of authority (SoA) resource record, except for those delegated to other DNS servers, while in LLMNR a server is authoritative just for the root of its zone and not for the sub-domains under it.

LLMNR uses the same TTL value for all the resource records in a server. This TTL value is chosen depending on how static or dynamic the network is. Larger TTL values reduce network traffic but generate stale cache entries in highly changing networks. For such networks, such as ad hoc networks, a TTL value of zero is recommended in the draft.

Regarding security aspects, both LLMR clients and LLMNR servers check the source addresses of the reply and query messages received, respectively, before accepting or discarding them. A client only accepts replies from servers with "on-link" IP addresses, i.e., with a source IP address that belongs to the same IP subnetwork as the client. Similarly, a server only answers unicast queries from "on-link" IP addresses, or from multicast queries that use local-scope multicast addresses. Moreover, servers must include in their answer messages just resource records that are reachable from the same subnetwork.

3 Comparative study of Multicast DNS and LLMNR

Both Multicast DNS and LLMNR keep the DNS message format, syntax and resource record format, although Multicast DNS introduces some changes in the way some of the fields of the DNS message are used (specifically, the use of the answer section in the queries). Regarding the use of these protocols together with DNS-SD to support service discovery, the main differences between both proposals are the following:

- Multicast answers: in LLMNR, servers answer using unicast, while in Multicast DNS they answer using multicast.
- Resource records caches: LLMNR recommends using TTL values of zero for ad hoc environments; therefore, no resource records are cached. Multicast DNS recommends using a TTL value of 120 seconds, and caches are used to improve the operation of the protocol.
- "Goodbye" message: The goodbye message is defined in Multicast DNS. LLMNR does not define an equivalent message. Since a TTL value of zero is recommended in LLMNR for ad hoc networks, no resource records will be stored in caches, and so no false or stale entries are possible.

In this section we will study through simulation the impact that these differences between Multicast DNS and LLMNR have when they are used for service discovery in ad hoc networks. We use OMNeT++[1].

We have simulated an area of 300×300 meters, with a number of devices (clients and servers offering services), all of them mobile, using a Random Way-Point model for the movements, with exponential "thinking times", and an IEEE 802.11 network interface in ad-hoc mode. We have used MAC broadcasts for multicast IP transmissions. Multicast multi-hop ad-hoc routing is not necessary, since both Multicast DNS and LLMNR are defined to be used just on the local link. The length of the simulation was elected to obtain results with a 90% confidence level and a 10% confidence interval.

The variables of our interest are: the number of messages transmitted (normalized per service search), the service discovery ratio (the ratio of services

[1] http://www.omnetpp.org/

discovered), and the service error ratio (ratio of stale or false services discovered).

An optimum service discovery solution for ad hoc networks should achieve as low number of messages transmitted as possible, so reducing power consumption, while keeping a high (close to 100%) service discovery ratio, and a low (close to 0%) service error ratio.

3.1 Multicast answer messages and the use of caches

In LLMNR, clients send queries using multicast, and servers send their answer messages using unicast; besides, clients are recommended not to make cache of the received answers. In service discovery terminology [12], this mode of operation is commonly known as "pull mode without cache". One of the main advantages of this mode is its reliability and its simplicity. In fact, its performance can be studied analytically. It can be shown that, since each time a service is needed, a query is sent, assuming no link failures, the service discovery ratio is 100%, and the service error ratio is 0%, since all available services respondi the query sent at the time when a service is needed. Given that there are n devices in the network, that each one offers a service, and that there are k different kind of services in the network, the number of messages transmitted per search follows Equation 1.

$$\text{NumberOfMessages} = \frac{k+n-1}{k} \tag{1}$$

Multicast DNS is more complex than LLMNR. Following again service discovery terminology, it behaves as a "pull mode with cache and with multicast responses", or what is equivalent, as a push mode with service announcement's rate controlled by the service request frequency in the network. Moreover, in Multicast DNS, service queries include previously known entries from the cache, what helps to reduce the number of replies necessary for that query.

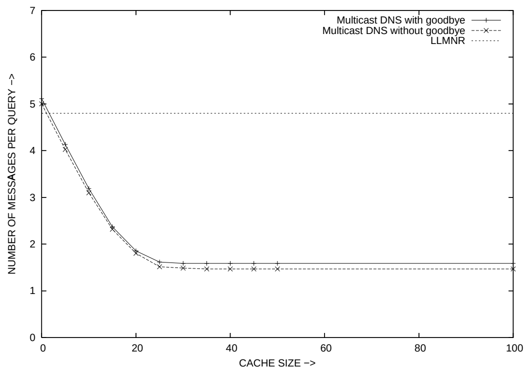
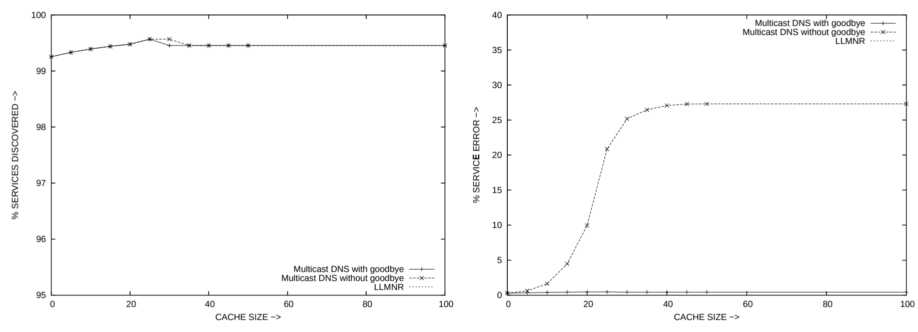


Fig. 1. Number of messages against cache size

Because of its complexity, we have carried out the performance evaluation of Multcast DNS through simulation. The results of these simulations are shown in Figures 1, and 2. The scenario simulated consists of 20 devices in average, each one with an average thinking time of 600 seconds, offering one service, with 5 different kinds of services in the network, and issuing a query (a service request) every 60 seconds in average. We simulate different cache sizes, from 0 to 100 entries (a cache with size n has space to store up to n resource records).



(a) (b)

Fig. 2. Service discovery and error percentages against cache size

As we can see in Figure 1, this mode of operation significantly reduces the number of messages per service search, compared with LLMNR. The price to pay is that this reduction comes with an increment in the uncertainty about the availability of the services discovered, since although the service discovery ratio reaches the 100%, Figure 2 (a), the service error ratio reaches a value close to 30% when moderate or large caches are used, Figure 2 (b) (plot labelled "Multicast DNS without goodbye")).

3.2 Goodbye messages

The above mentioned lost of reliability in the Multicast DNS protocol is alleviated through the use of the cache consistency mechanism defined in Multicast DNS. This mechanism allows deleting staled cache entries by using "goodbye" messages. We have repeated the simulations introducing now the use of the "goodbye" message, Figure 2 (plot labelled "Multicast DNS with goodbye"). We see that the service error ratio is reduced to 0% while the increase in the number of messages transmitted is not significant, and continues well under LLMNR, see Figure 1 (plot labelled "Multicast DNS with goodbye").

Considering the results we have obtained, we can conclude that Multicast DNS is more suitable than LLMNR to be used for service discovery in ad hoc networks, since it preserves protocol reliability while significantly reducing the number of transmissions for service discovery, and so the power consumed.

4 Proposed improvements to Multicast DNS

As we have seen above, Multicast DNS is more efficient for service discovery in ad hoc networks than LLMNR. However, the traffic efficiency of the protocol can be significantly improved with some simple modifications. In this section we propose four simple modifications, and evaluate how they improve the performance of Multicast DNS. They all try to reduce the number of network transmissions and receptions, particularly for the more limited devices, and so their power consumption.

4.1 Use services stored in the cache for the answers

In ad hoc networks, cooperation among devices is essential since the devices can carry out more complex tasks at a lower cost thanks to the cooperation. Our first proposal of modification for Multicast DNS is to allow all the devices that know about a service, not just those devices that offer themselves the service, to answer a service request query. In other words, we allow using the resource records in the cache (i.e., the non-authoritative resource records) for the replies.

Moreover, prior to answering, a device first listens for answer messages to the same query from other devices[2], it checks whether it knows about any other service that has not been announced yet, and if so, it sends its answer message, and if not it aborts its reply. This way, all devices cooperate to build the list of all available services of the requested type with the minimum number of messages transmitted, see Figure 3. In this figure, we can see that the reduction in the number of messages transmitted in this scenario is 12, 4% for big enough caches.

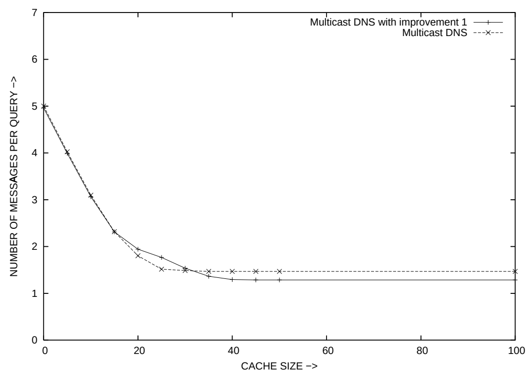


Fig. 3. Number of messages against cache size with improvement 1

[2] Remember that, to avoid collisions, in Multicast DNS all devices wait a random time before sending a reply to a query. We will come back on this later in this section.

4.2 Update the cache with the services included in the query

Continuing with the philosophy of exploiting the cooperation between devices, the second improvement we propose consists on updating the caches not just with the resource records obtained from answer messages, as Multicast DNS specifies, but also from the list of previously known services included in search queries. This way, as Figure 4 shows, the number of search messages is reduced in our scenario a $22,8\%$ with respect to Multicast DNS, for big enough caches.

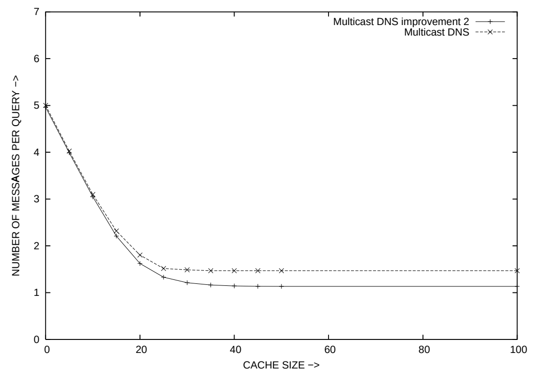


Fig. 4. Number of messages against cache size with improvement 2

4.3 Use a different distribution for the random waiting time

In Multicast DNS, to avoid collisions after a query, servers do not answer immediately, but wait a random time drawn from a uniform distribution between 20 and 120 msec. Our third proposal consists on generating this random time following a more intelligent distribution that statistically guarantees that devices with less energy constraints (e.g., with an AC adapter plugged in), and which know about more services, answer first, making most of the times unnecessary for the most limited devices to answer. To achieve this, we propose to generate the random time inversely proportional to the Time-To-Live (TTL) associated to the device, and to the number of services it knows. We assume that battery powered devices will have a low TTL configured (which is consistent with the fact that they are highly mobile).

Specifically, we propose the random time to be drawn from the expression in Equation 2, where $U(x, y)$ represents a uniform distribution between x and y, and the value 7200 sec. (120 minutes) is an heuristically chosen parameter that represents the time starting from which a device can be considered static.

$$\mathrm{U}(20, \quad 120 * \frac{7200}{7200 + \mathrm{TTL} * \#\mathrm{Cache\_Entries}}) \tag{2}$$

We have simulated an example scenario to measure what percentage of the answer messages are transmitted by different devices with different TTL values, in a heterogeneous scenario with 20 devices in average, with five different values of TTL: 500, 2500, 4500, 6500 and 9500 seconds, which are also their average thinking times. There are the same number of devices of each type (i.e., 20% with each TTL). The cache has a capacity for 10 entries, except for the more static devices (the ones with TTL = 9500), that are less limited and have a cache with capacity for 100 entries. The rest of the parameters of the simulation are the same than in previous ones.

Figure 5 shows the results of this simulation. We see that changing the strategy of generation of the random waiting time causes that 75% of the queries are answered by the devices with larger TTL, reducing the answer messages from the others to a forth of what they would have answered with an uniform distribution (as in Multicast DNS), and so reducing their power consumption. Moreover, the number of messages sent is reduced a 58, 86% because the devices with larger TTL and greater cache, which answer the 75% of the queries, are the ones that have a more complete and accurate view of the network, and most of the times the reply from any other device is not needed because there is no other new service to add.

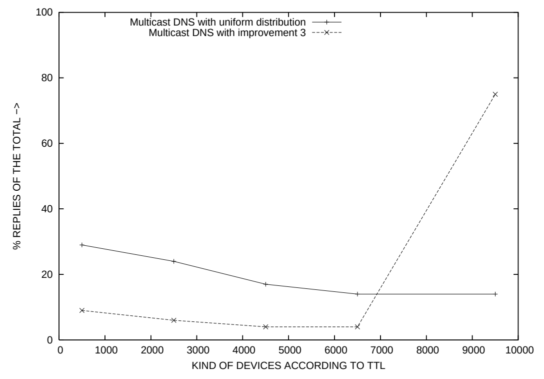


Fig. 5. Percentage of answer messages against TTL of the device with improvement 3

4.4 Optimize "one query-one response" queries

As we observed before, Multicast DNS distinguishes three modes of operation of the applications in search for a service: "one-shot queries" (also known as "one query-one response"), one-shot queries accumulating multiple responses ("one query-multiple responses"), and continuous querying. However, no field in the DNS message is used to distinguish one type of query from the other, and so the answers from the servers are the same in all cases; it is the client itself which, for example, in the case of one-shot queries, selects the first answer and discards the rest.

Our last proposal consists on defining a flag in the DNS header that could be used in Multicast DNS to indicate whether the query is of the kind “one query-one response” or “one query-multiple responses”. For this flag, any of the currently unused bits (9 to 11) of the parameters field of the DNS message header could be used.

This way, if a server receives a DNS query with the “one query-one response” flag set, before sending its reply, if it listens another reply from other device in the network, it will abort its reply, even though it would have something new to say. This way, the bandwidth consumed is greatly reduced.

5 Conclusions and future work

In ad hoc networks, devices must be able to discover and share services dynamically. Several protocols have been proposed for service discovery. Recently the IETF has proposed the use of the DNS protocol for service discovery. For ad hoc networks, the IETF works in two proposals of distributed DNS, that can be used for service discovery: Multicast DNS and LLMNR.

In this paper we have reviewed and analyzed both proposals from the point of view of their efficiency when used for service discovery in ad hoc networks. From our study, we conclude that the one that better fits the requirements of these kinds on environments is Multicast DNS. However, some very simple improvements can be introduced that help to improve their efficiency, especially regarding power consumption in limited devices. In this paper we have proposed and analyzed through simulation four improvements. The reduction in the number of messages transmitted is about 35%, depending on the scenario, for one query-multiple response requests, and may be much greater for one query-one response. Besides, this reduction is achieved in those devices where it is more necessary, in the more limited devices.

We are working on validating the viability of our proposals via real implementation. In this sense, starting from an implementation in J2SE of Bonjour [19], we are completing an implementation in J2SE of Multicast DNS and DNS-SD with and without the power-saving improvements we propose. There is also an implementation of Rendezvous for network cameras Axis 2100.

As a future work, besides finishing the implementation of our improvements to Multicast DNS in J2ME for PDAs, and in other devices usually found in pervasive computing environments, we are also interested in broaching the following problems. First, we want to test other distributions for the generation of the random time, and to study their effect and how to achieve a further reduction. Secondly, today the value of TTL is configured manually both in Multicast DNS and in LLMNR devices, but it would be very interesting that this value could be automatically learned from the mobility behaviour of the device, without any direct intervention from the user. Thirdly, we plan to do more simulations using different multicast ad-hoc routing protocols in larger areas, instead of IP broadcasts. Finally, we are aware of the security problems inherent with ad hoc networks, and we are working in a distributed trust model, so these networks can include automatic mechanisms to adapt the trust relation between the devices as they experiment positive and negative experiences [20].

References

1. RFC 2165: Service Location Protocol (1997)
2. Goland, Y.Y., Cai, T., Leach, P., Gu, Y.: Simple Service Discovery Protocol/1.0. Internet-Draft (work in progress) (1999) draft-cai-ssdp-v1-03.txt.
3. Jini: Architectural Overview. White Paper (1999)
4. Salutation Consortium: Online available at http://www.salutation.org (1998)
5. Bluetooth: (Specification v1.1, Part E: Service Discovery Protocol (SDP))
6. Association, I.D.: Infrared data association link management 1.1 (1996)
7. Chakraborty, D., Joshi, A., Yesha, Y., Fini, T.: GSD: A Novel Group-based Service Discovery Protocol for MANETS. In: 4th IEEE Conference on Mobile and Wireless Communications Networks (MWCN 2002), Stockholm. Sweden (2002) 140–144
8. Oh, C.S., Ko, Y.B., Kim, J.H.: A Hybrid Service Discovery for Improving Robustness in Mobile Ad Hoc Networks. In: The International Conference on Dependable Systems and Networks. DSN-2004, Florence, Italy (2004)
9. Nidd, M.: Service Discovery in DEAPspace. IEEE Personal Communications (2001)
10. Helal, S., Desai, N., Verma, V., Arslan, B.: Konark: A System and Protocols for Device Independent, Peer-to-Peer Discovery and Delivery of Mobile Services. IEEE Transactions on Systems, Man, and Cybernetics **33**(6) (2003) 682–696
11. Barbeau, M., Kranakis, E.: Modeling and Performance Analysis of Service Discovery Strategies in Ad Hoc Networks. In: International Conference on Wireless Networks. ICWN 2003, Nevada. Canada (2003) 44–50
12. Campo, C., Garcia-Rubio, C., Marin, A., Almenarez, F.: PDP: A lightweight discovery protocol for local-scope interactions in wireless ad hoc networks. Computer Networks (2006)
13. Zhu, F., Mutka, M., Ni, L.: Service discovery in pervasive computing environments. IEEE Pervasive Computing (2005)
14. Jones, C.E., Sivalingam, K.M., Agrawal, P., Chen, J.C.: A Survey of Energy Efficient Network Protocols for Wireless Networks. Wireless Networks **7**(4) (2001) 343–358
15. Feeney, L.M., Nilsson, M.: Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment. In: IEEE INFOCOM. (2001)
16. Cheshire, S., Krochmal, M.: DNS-Based Service Discovery. Internet-Draft (work in progress) (2005)
17. Cheshire, S., Krochmal, M.: Performing DNS queries via IP Multicast. Internet-Draft (work in progress) (2005)
18. Esibov, L., Adoba, B., Thaler, D.: Linklocal Multicast Name Resolution (LLMNR). Internet-Draft (work in progress) (2005)
19. http://jmdns.sourceforge.net/.
20. Díaz, D., Marín, A., Almenárez, F.: A smartcard solution for access control and trust management for nomadic users. In: Seventh Smart Card Research and Advanced Application IFIP Conference (CARDIS 2006), Tarragona (Spain) (2006)