Service Discovery in Ad Hoc Networks

Paula Kuosmanen

Abstract— **Service advertisement and discovery is an important component for ad hoc communications and collaboration in ubiquitous computing environments. However, the nature of ad hoc networks pose additional problems – caused by highly dynamic network topology and nodes' limited power sources - compared to wireless networks with infrastructure, even if the infrastructure is just services that stay continuously available most of the time, like a network printer. Service discovery protocols have been designed specifically for ad hoc networks, e.g. JESA, DEAPspace, and Konark, but they also fail to solve all the problems.**

Index Terms—**ad hoc networks, service discovery, ubiquitous computing**

I. INTRODUCTION

Service is an essential concept to ubiquitous computing - computing that integrates computation into the environment. In a truly ubiquitous computing environment devices and services are required to properly discover, configure and communicate with each other without human interaction. Services are self-describing, location and platform independent components that are meaningful to a human user, like printing or file storage. They can be dynamically discovered and configured. Service-oriented computing is the only way with our current knowledge to implement a ubiquitous computing environment.

The concepts of service and service discovery have been around for a while, and several protocols do exist. One of the most common examples cited in literature is that of a visiting person (usually a reporter) in a foreign office. She writes something on her laptop and wants to use the office printer to print it, preferably without having to ask someone first for the printer drivers and the name of the printer. As long as her laptop and the printer both support service discovery, they can dynamically connect to each other and the printing job gets done. This is the simple scenario. Since it happens in a "smart office", there surely is a service directory to ask connection details from, or even if there isn't, the printer stays still, powered up and definitely reachable.

However, truly ubiquitous computing also requires support for ability for mobile nodes to communicate together whenever there are any other nodes present without any fixed infrastructure (like the printer in the previous example). These so-called ad hoc networks have no central administration and due to their dynamic nature the user can never know where there might be services present, how long they will stay present or how they can be located. For example a mobile phone could connect to another mobile phone that had an access to the Internet and use Internet through it. In this scenario the connecting phone might suddenly run out of power or move out of range. Therefore for a ubiquitous computing environment there is a need for a service discovery protocol that can also function satisfactorily in ad hoc networks.

In addition to all the challenges inherit in wireless communication, ad hoc networks pose additional challenges to service discovery. Mostly these challenges are technology-oriented, although some do originate directly from ways of thinking and trying to fit solutions for fixed networks into ad hoc networks. This far no perfect service discovery protocol for ad hoc networks has been created. All have their shortcomings, whether it be about wasted bandwidth or assuming the presence of a certain node.

In this paper ad hoc networks are described and the challenges they pose to service discovery analyzed. These challenges include the rapidly changing network topology and limited power sources. Also larger, multi-hop ad hoc networks cause additional problems most service discovery methods fail to address. Additionally several service discovery methods are analyzed for their suitability to ad hoc networks.

The rest of this paper is structured as follows: section II presents an overview of ad hoc networks. Section III introduces the challenges ad hoc networks pose to service discovery, and in section IV existing solutions are analyzed. Section V contains additional problems even the protocols designed for ad hoc networks fail to solve and solutions for them. Conclusions are drawn in section VI and directions for further work are also discussed.

II. OVERVIEW ON AD HOC NETWORKS

An ad hoc network is a collection of potentially mobile nodes - i.e. communication devices - that have a desire to communicate but no fixed infrastructure available. Each individual node is responsible for dynamically finding all other nodes it can directly communicate with. However, usually not all nodes can communicate directly with all other nodes, so nodes are also required to relay packets on behalf of other nodes in order to deliver data across the ad hoc network [1].

Ad hoc networks can be built around any wireless technology, e.g. IEEE 802.11 or Bluetooth. They have several characteristics that clearly differentiate them from fixed and cellular networks. First of all, because the nodes can move independently, the network topology can change rapidly and in ways impossible to define in advance. With a node acting only as an end-host this situation doesn't really differ from moving in a cellular network but each node may also act as a router for other nodes, therefore disrupting also their communications. Also transmission links can change rapidly and randomly as also the router nodes relying on limited batteries adjust their transmission and receiving parameters to save energy [2]. In addition to changing topology and link conditions, ad hoc networks also face the problem of trust and security. In addition to not being able to trust the wireless transmission medium, the nodes can't trust each other either, not even the ones acting as routers. This topic, however, is beyond the scope of this paper.

Ad hoc networks are usually used in situations where infrastructure isn't available, can't be trusted or shouldn't be relied on in emergency situations. Possible applications include military settings, networks of sensors scattered over a certain area, space or undersea exploration, and such everyday thing as temporary communication outdoors or in buildings with no fixed networks [1].

III. CHALLENGES FOR SERVICE DISCOVERY

A. Rapidly changing network topology

All traditional service discovery approaches more or less assume fairly stable network. Even in mobile computing at least the base stations are expected to be fixed components. However, as an ad hoc network is characterized by the lack of fixed infrastructure and rapidly changing network topology, there can be no assumptions of any node being available at certain point in time. If a centralized service directory would be used, its data would soon be inconsistent due to frequent disconnections of nodes that provide services. And if the directory node were to be disconnected, the whole system would be paralyzed [3]. One solution is not to use directories at all.

B. Limited power sources

The majority of ad hoc networks consist of wireless devices that usually rely on batteries or other limited power sources. This places two important restrictions on the behavior required from them. First, they should not be required to spend unreasonable amounts of energy in their communications, and second, they may be available only periodically to conserve their batteries. The challenges posed by the latter were already discussed in the previous section in the case of a node suddenly being disconnected from the network.

If a node can not rely on the existence of a service directory, it must use some method to locate the services directly. In wireless networks this method is usually broadcasting or multicasting. This approach is not without problems either. To keep a high level of consistency, frequent messages are required. This in turn consumes huge amounts of both bandwidth and energy [3]. In ad hoc networks it is extremely important to keep the number of transmission to the minimum to conserve batteries, so another approach is required. Some attempts are introduced in section IV.

C. Suitability of traditional protocols

The most commonly known and used service discovery protocols Jini, Universal Plug and Play (UPnP), Salutation and Service Location Protocol (SLP) [4] are all mainly designed for networks with fixed infrastructure. The client is expected to be the only node that moves. This is clearly not the case in ad hoc networks.

Jini can be ruled unsuitable simply because it uses a centralized discovery approach. Even though a look-up directory could be in theory set up on demand and found via multicasting, there are no guarantees in an ad hoc network that the directory node won't leave the network between the service registration by the service provider and the service look-up by the client. Also Universal Description, Discovery and Integration (UDDI) commonly used in distributed web-based services shares this weakness. The other three aforementioned protocols do not require the use of centralized directory [5] and therefore could be suitable for ad hoc communication in this aspect – except that their constant multicasting for services or of service offers would cause all searching nodes and/or provider nodes to run out of energy far too soon. And in multi-hop networks also nodes not involved in the offer/request transmission would have to use their limited resources to forward the messages. Caching of service offers isn't a working alternative either as the service provider may leave the network at any time.

The biggest obstacle, however, is that in ad hoc environment all these protocols are designed for small networks and focus on single-hop communication – if multicast is used in bigger networks, a flood of messages is generated; if broadcast is used, a client node may remain oblivious of services its neighbor knows about as other clients aren't required to store and forward service information. Therefore none of the above-mentioned protocols are completely suitable for ad hoc service discovery, and new protocols have been invented specifically for ad hoc networks.

IV. SERVICE DISCOVERY PROTOCOLS FOR AD HOC NETWORKS

A. Basics

Many attempts have been made to either adapt existing service discovery protocols into wireless ad hoc environments or to create completely new service discovery protocols. In the commercial field the developments include Salutation Lite and JiniME, both of which are "lightweight" versions of their respective protocols especially designed for mobile devices with limited power supplies. However, neither of these really addresses the other challenges of ad hoc networking.

Service discovery protocols can be directory-based or directory-less, depending on whether a directory that stores the service advertisements exists or not. Directory-less protocols, e.g. Konark, always depend on broadcasting or multicasting of either service offers or requests, or in some cases, both. The directory-based architectures are further divided into centralized (e.g. Jini) and distributed directory architectures, former of which can not be used successfully in ad hoc networks. Distributed directory architectures consist of infrastructure-less distributed directory architectures (e.g. DSDP) and infrastructure-based distributed directory architectures (e.g. VIA), depending on whether directories reside on mobile ad hoc networks or on infrastructure-based networks [6]. In addition to these clear-cut definitions, also so-called hybrid architectures exist. Usually they can use both a directory-based and directory-less approaches.

In the following subsections three service discovery methods designed more or less for ad hoc networks are presented, namely JESA, DEAPspace, and Konark. Each of these presents a slightly different approach. Centaurus [7], although often mentioned in the context of ubiquitous computing, isn't included, since it's designed mainly for mobile, heterogeneous clients, not ad hoc networking – even though it is sometimes mentioned in ad hoc context.

B. *JESA*

Java Enhanced Service Architecture, or JESA for short, is a lightweight, Java-based middleware for spontaneous service discovery and usage. JESA's core component is JESA Service Discovery Protocol (JSDP) that can work transparently with or without a service broker, thus allowing both small peer-to-peer communities and large scalable networks. In peer-to-peer mode services are advertised or searched for using broadcast messages and replies are sent in unicast. In larger networks service providers register with a service broker and must after that stop multicasting advertisements and replying to requests. Brokers are discoverable as services that answer requests for registered services [8].

Even though officially JESA has been developed for ad hoc networks, it doesn't solve all the problems inherent in ad hoc networking. It is scalable but it doesn't address network topology or the requirements for minimum number of transmissions. If the broker a service provider has registered with is disconnected, the service is no longer available. And searching for a broker instead of a service by using multicast doesn't save the client (or any other nodes) any transmissions.

C. *DEAPspace*

DEAPspace comes a bit closer to a truly distributed system for ad hoc networking. It isn't a middleware, but an algorithm. Unlike in JESA, in DEAPspace each node stores device and service information of all the devices it is aware of. This "world view" is then periodically broadcast to all neighboring nodes. Each listening device will incorporate this information into its own world view. Failures to receive broadcast – e.g. because the connectivity was poor – doesn't result in any significant problems either, since any node that did not miss the transmission will repeat the information in its own world view [9]. However, this periodic broadcast could result in a multicast storm [6], effectively flooding the communication channel and draining the batteries of the nodes.

D. *Konark*

Konark is also a middleware designed specifically for peer-to-peer ad hoc networks. Like DEAPspace, it's completely distributed system. In Konark each participating device has the capabilities to host its local services, deliver its own services using a resident micro-HTTP server, query the network for available services offered by others, and use the services it discovers in the network. What separates Konark from DEAPspace is its service discovery protocol. Instead of broadcasting, Konark nodes use multicasting. Additionally, when a node sees a service message on Konark multicast address, it multicasts the difference of its own view of relevant services and services in the received message, instead of all of its knowledge. The multicast is scheduled to wait a random time, so that the multicast storm of immediate, simultaneous multicasts can be avoided and there is time for the network to learn of available services. Nodes that have no new information compared to the multicasts by others up to their scheduled point will then be able to cancel their transmissions. This way unnecessary network bandwidth usage can be reduced, unlike in plain periodic advertisement schemes or DEAPspace [10].

E. *Conclusion*

Even though it is clear that a centralized directory won't work, and neither will simple advertisement/request multicasting, judging from the previous analysis a good solution may not exist.

But all hope is not lost. A good solution may not exist yet, but research is on-going. One alternative approach to decentralized service discovery that could solve the message flood problem, a service ring, is presented in [11]. A service ring groups together devices that are both physically close to each other and offer similar services. Within each ring is a designated service access point (SAP) that knows about all the services offered within its ring. Search messages enter the ring only if the SAP knows that the service might be offered within the ring, and thus large parts of the network do not need to be visited at all. As long as the nodes within the ring have a way of noticing the SAP's disappearance and also a way to periodically affirm their existence to it, this approach also works rather well in ad hoc networks. With time, also other approaches will most likely be found.

V. ADDITIONAL PROBLEMS AND SOLUTIONS

This far in this paper different approaches have been considered only from service discovery perspective and by analyzing one approach at a time. This section looks briefly again at the vision of truly ubiquitous computing and considers other ways to reduce the required message transmissions and power usage than reducing the number of sent messages.

Traditionally service discovery is implemented at the application layer. However, in [12] it is suggested that service discovery could also be implemented at the network layer. Coupling service discovery with routing would be more convenient, as in reality routing and service discovery are not independent activities in wireless networks. Once a service request is received by a provider, it is necessary to establish a route to the client, or once a service offer is received by a client, it needs a route to the service provider being advertised.

Although many protocols for service discovery have been invented and many problems solved to at least some degree, one of the biggest obstacles for truly ubiquitous computing, computing that integrates seamlessly with the environment and allows any device to use any service in the vicinity, still remains: not all clients can use all services. Jini clients can call only Jini services, UPnP only UPnP, and so forth – the rift between protocols is wide and no single protocol is strong enough to become a standard. As long as wireless devices can not communicate with each other regardless of their types, there can be no smooth ad hoc service discovery nor true ubiquitous computing.

Mappings have been made, achieving some level of interoperability in e.g. from Salutation to Bluetooth SDP [13]. Also middleware solutions have been invented that discover the services available at a certain location, independent of the advertising discovery protocol and then dynamically adapt themselves to allow interoperation [14]. However, both approaches fail to achieve their goal. Any single mappings made do not provide a universal service discovery, and with either making all the mappings or the middleware approach the mobile nodes really do not have the resources to support the kind of complex code that would be needed.

VI. CONCLUSIONS

Ad hoc networks are difficult environments for service discovery. With ad hoc networks I mean truly ad hoc networks where all nodes are potentially mobile, and connect and disconnect spontaneously. Most visions of ubiquitous computing environment nowadays consist of a fixed architecture and a visiting client node, and therefore are not as difficult to deal with as dynamic ad hoc networks where also the service providers or discovery directories can vanish without warning. Unfortunately scientific literature mixes these two concepts far too often and misses the fact that in ad hoc networks one can not presume that a fixed directory node is always present.

Even protocols supposedly designed for ad hoc networks don't quite make it. The essential challenge practically boils down to one question: how to tell the clients where the service providers are? It is certain that one centralized directory can not be used, as its disappearance would bring the whole system to a halt. But a distributed directory isn't enough either. Whenever a directory node goes down, the services registered with it become unavailable. On top of all, they don't know it and patiently keep waiting for service requests that never arrive. Having to renew the registration periodically solves this problem somewhat, as well as the problem of finding out which service providers are no longer in the network. But then again, this "lease renewal" wastes the provider's – that may also be a mobile node with limited battery – energy with constant transmission even though it has enough work servicing its clients.

The other alternative is not to use a directory at all, but in this approach we also run into the limitations of energy needed for constant transmissions. Whenever a service is advertised or searched for, the message is multicast and all nodes must waste their energy forwarding it, even though they may not be interested in at all. On the other hand, if each client caches service advertisements, it must constantly check if the provider is still there. Again, energy is wasted.

Both of these approaches are used in the current protocols, and neither of them quite works as hoped. Also hybrid approaches are used, but mixing two inadequate approaches doesn't create a good one. Unfortunately there may not be a good solution, as it's either-or – either the directory or multicasting. Some improvements that reduce the amount of multicast traffic have fortunately been made, like in Konark or service rings, but as ad hoc networks grow larger and larger, even that may not be enough. What is needed is an approach that doesn't depend on a directory but also minimizes all extra overhead. Even though it has not been implemented in commercial products yet, the proposed piggybacking of service discovery messages with routing messages may actually be a working solution. After all, routing messages have to be sent in any case in multi-hop wireless networks.

REFERENCES

[1] Ram Ramanathan and Jason Redi. (2002, May). A Brief Overview of Ad Hoc Networks: Challenges and Directions. *IEEE Communications* Magazin*e* [Online]. 40(5). pp. 20-22.

[2] M. Scott Orson, Joseph P. Macker and Gregory H. Cirincione. (1999, July-August). Internet-based mobile ad hoc networking. *IEEE Internet Computing* [Online]. 3(4). pp. 63-70.

[3] Radu Handorean and Gruia-Catalin Roman (2002). Service Provision in Ad Hoc Networks. *Proceedings of the 5th International Conference on Coordination Models and Languages* [Online]. 2315. pp. 207 - 219.

[4] Sumi Helal (2002), Standards for Service Discovery and Delivery. *IEEE Pervasive Computing* [Online]. 1(3). pp. 95-100.

[5] Andry Rakotonirainy and Greg Groves (2002). Resource Discovery for Pervasive Environments. *Proceedings Book On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE: Confederated International Conferences CoopIS, DOA, and ODBASE 2002* [Online]. 2519. pp. 866-883.

[6] Chunglae Cho1 and Duckki Lee. Survey of Service Discovery Architectures for Mobile Ad hoc Networks. [Online]. Available: http://www.cise.ufl.edu/class/cen5531fa05/files/ccho-dlee.pdf

[7] Lalana Kagal, *et al*. A Highly Adaptable Infrastructure for Service Discovery and Management in Ubiquitous Computing. [Online]. Available: http://www.cs.umbc.edu/courses/graduate/CMSC691T/spring2001/rlist/centuarus.pdf

[8] Stephan Preuß (2002, May). JESA Service Discovery Protocol: Efficient Service Discovery in Ad-Hoc Networks. *International IFIP-TC6 networking conference No2* [Online], Pisa. (2345). pp. 1196-1201.

[9] Moreno Marzolla (2001, July 17). Computing Everywhere: DEAPspace and Web Presence. [Online]. Available: http://www.dsi.unive.it/~marzolla/dottorato/panzieri.pdf

[10] Sumi Helal, Nitin Desai, Varun Verma and Choonhwa Lee (2003, Marc). Konark – A Service Discovery and Delivery Protocol for Ad-Hoc Networks. *IEEE Wireless Communications and Networking, 2003. WCNC 2003* [Online]. 3. pp. 2107- 2113.

[11] Michael Klein, Birgitta König-Ries, and Philipp Obreiter (2003, Sep). Service Rings – A Semantic Overlay for Service Discovery in Ad hoc Networks. *14th International Workshop on Database and Expert Systems Applications, 2003. Proceedings* [Online]. pp. 180- 185.

[12] J. Antonio Garcia-Macias and Dante Arias Torres (2005, June). Service Discovery in Mobile Ad-Hoc Networks: Better at the Network Layer? *Parallel Processing, 2005. ICPP 2005 Workshops. International Conference Workshops on* [Online]. pp. 452 – 457.

[13] F. Zhu, M.W. Mutka and L. M.Ni (2005, Oct/Dec). Service discovery in pervasive computing environments. *IEEE Pervasive Computing*. 4(4). pp. 81-90.

[14] Paul Grace, Gordon S. Blair, and Sam Samuel (2003). Middleware Awareness in Mobile Computing. *Proceedings of the 23rd International Conference on Distributed Computing Systems* [Online]. pp. 382.