SERVICE DISCOVERY IN MOBILE PEER-TO-PEER ENVIRONMENT

Arto Hämäläinen
Lappeenranta University of Technology
P.O. Box 20, 53851 Lappeenranta, Finland
arto.hamalainen@lut.fi

Jari Porras
Lappeenranta University of Technology
P.O. Box 20, 53851 Lappeenranta, Finland
jari.porras@lut.fi

Pekka Jäppinen
Lappeenranta University of Technology
P.O. Box 20, 53851 Lappeenranta, Finland
pekka.jappinen@lut.fi

ABSTRACT

Future pervasive computing service platforms are widely used with wireless mobile devices. In such wireless ad hoc and peer-to-peer environments, service discovery plays important role. In addition to functional service discovery protocol, efficient utilization of available network technologies improves the usability of an ad hoc communication system. PeerHood is our peer-to-peer environment, which tries to address these issues. This paper presents the service discovery process in PeerHood.

KEYWORDS

Service discovery, peer-to-peer, ad hoc networking

1. INTRODUCTION

Mobile networks are in a significant role in future service networks. Mobile service networks are not created by duplicating all Internet services, but by selecting suitable ones and by creating new ones to support them. A service platform is required to control the service creation, deployment, control and discovery to enable efficient service operation in mobile networks. The I-Centric Communications platform defined inside the Wireless World Research Forum [1] is such a platform. Tafazolli [2] named peer-to-peer and ad hoc communication as the first requirements for I-centric service platform. This paper will describe PeerHood, our peer-to-peer environment, and especially its service discovery procedures.

Service performs some task needed by the user. Mobile Services are services which are used by wireless mobile devices. Mobile communication can be based on infrastructure or it can be ad hoc communication in peer-to-peer manner. Particularly in ad hoc

communication, Mobile Services set requirements for service discovery process, placing service discovery protocols in essential position.

We start this paper by summarizing related research in service discovery area. General components of a service discovery protocol will be explained and characteristics of existing service discovery protocols compared. Subsequently, we present the general architecture of PeerHood, our Peer-to-Peer environment. The main topic of this paper is to describe the service discovery procedure in PeerHood. This discussion is presented after general description of PeerHood environment. Finally, we give our conclusive remarks.

2. RELATED WORK

Considerable amount of research and development has been done in the context of service discovery protocols, mainly for use in wired and infrastructure-based wireless networks. The best known examples of service discovery protocols developed both in the industry and academia are Lookup service of Jini [3], UPnP [4], Bluetooth Service Discovery Protocol [5], Service Location Protocol [6] and UDDI [7].

Zhu et al. [8] have created a common taxonomy for service discovery, and compared service discovery protocols according to components based on this taxonomy. In this taxonomy, 10 different components have been discovered. *Naming of services and attributes* can be either template-based or template-based and predefined. Division can also be done according to *initial communication method* - whether it is unicast, multicast or broadcast. Services can be *discovered and registered* either by querying or based on announcements. In the *service discovery infrastructure* point of view, certain protocols require service directory, whereas others can operate without a directory. *State of service information* can be either hard-coded or soft state. The *scope of service discovery* can be based either on network topologies, user roles or contextual information. Usability of services is often improved or weakened by the chosen *service selection* method. Automatic service selection based on application-specific service weight simplifies the selection process, but may cause selection of unsuitable services. On the other hand, manual service selection causes the user more inconvenience. *Service invocation* is done simply by providing service location, defining communication mechanism or even application operations. *Service usage* is explicitly released or based on leases. Client can keep up with the s*tatus of service* by receiving notifications from the service, or by explicitly polling it periodically.

In [9], authors have discovered three approaches to service interaction. First way of interacting with a discovered service is to *download the service component* to the client device and execute it on the device. There's no need to maintain connectivity after the service component has been downloaded. Alternative approach for *interacting with the discovered service* is through a defined interface. The interface component can be downloaded to device like in Jini, or the interface can be defined in the control messages of the service discovery process. Yet another interaction method exists. *Emulation* can be

provided to translate control messages and service responses between the client and the service provider.

Bluetooth wireless technology has been developed as an ad hoc wireless technology, and therefore service discovery has been a requirement from the start of the development of the technology. Bluetooth Service Discovery Protocol (SDP) is a part of the main Bluetooth specification [5]. Bluetooth SDP operates on Bluetooth Logical Link Control and Adaptation Protocol layer (L2CAP). SDP can be used to search for specific Bluetooth service or browse services on a specific device or on all remote devices. SDP retrieves service records for each service, including service name, service class, protocol descriptors and profile descriptors. Bluetooth SDP uses unicast as an initial communication method, although usually SDP is run immediately after Bluetooth inquiry process with discovered devices as target addresses, reaching all discoverable and connectable devices in the neighborhood.

3. PEERHOOD ARCHITECTURE

The main goal of PeerHood is to provide a communication environment, in which mobile devices act in a peer-to-peer manner [10]. PeerHood helps applications to find remote devices and services and connect to them. Device and service discovery is an essential part of such environment. In order to quickly establish ad hoc connections, PeerHood is constantly monitoring the wireless neighborhood for other PeerHood-enabled devices. Services on each PeerHood device are stored locally on device and service database. Stored information is provided on request for PeerHood applications.

In addition to core immediate personal area network (PAN), consisting devices found nearby, PeerHood is designed to enable connections to selected distant remote devices. Using infrastructure networks, e.g. Internet, home, corporate and vehicular area networks, a user can extend his network to cover all his personal services regardless of his location. This approach called Personal Network (PN) has been introduced in [11]. In PeerHood, this approach is supported with the GPRS plugin, which uses GPRS or similar infrastructure network to expand the PeerHood neighborhood.

PeerHood architecture can be divided in three separate entities. The PeerHood daemon is the component, which ensures the continuous knowledge about other wireless devices. PeerHood library is the component, which provides applications interface for operations provided by PeerHood. PeerHood library is used both in local communication between application and daemon and between PeerHood applications in different devices. PeerHood plug-ins provide functions specific to the wireless network technologies. Plug-ins are used by the PeerHood library and the PeerHood daemon. Relationship between different components can be seen in Figure 1, which pictures the PeerHood software architecture.

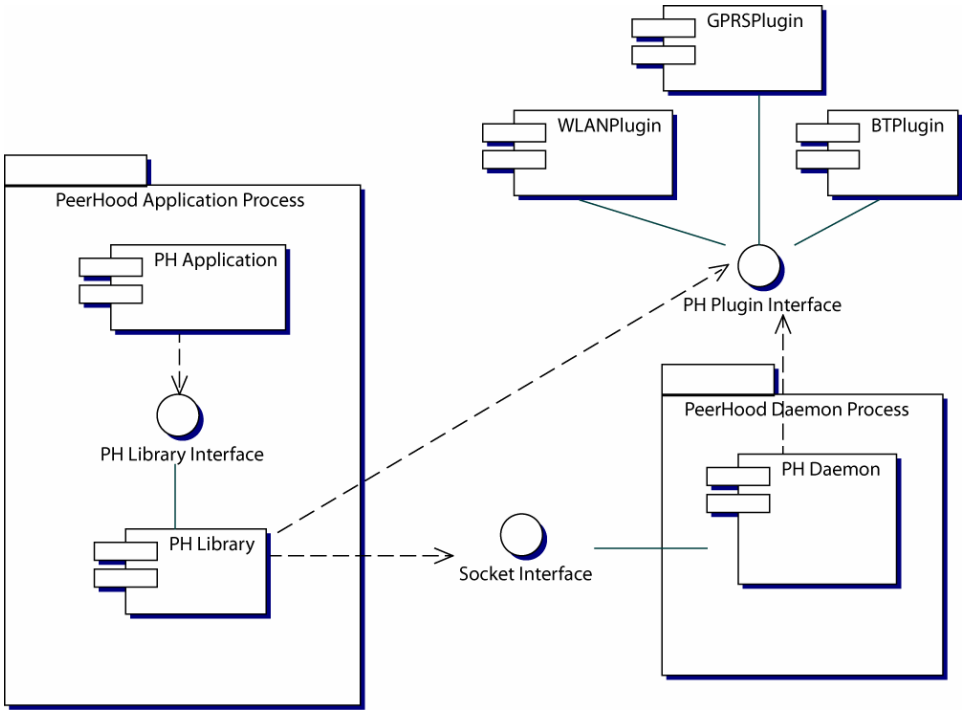

Figure 1 - PeerHood software architecture

3.1 PeerHood daemon

Main operation of PeerHood is performed by the PeerHood daemon. PeerHood daemon is a background application, which takes care of the device and service discovery. Its responsibilities include maintaining a list of both remote and local services.

Communication between PeerHood-enabled application and PeerHood daemon is handled using local socket interface provided by the PeerHood library. Applications can use this interface to search for remote devices and services and to register services to local PeerHood daemon. Daemon takes care of answering to service requests.

3.2 PeerHood plug-ins

Different network technologies have been implemented as plug-ins, which can be loaded dynamically by other PeerHood components. Currently, network plug-ins have been implemented for Bluetooth, Wireless LAN and GPRS.

Bluetooth offers several choices for data communications. First of all, data communications can be established directly on Logical Link Control and Adaptation (L2CAP) layer. L2CAP is the layer, which all data transmission rely on when using Bluetooth wireless technology. On top of L2CAP, RFCOMM protocol provides virtual serial port support, on top of which either OBEX or PPP connections can be established. OBEX is designed for exchange of small data objects, for example vCards. PPP provides

point to point IP connection between devices. However, another choice for IP connections over Bluetooth exists, namely Bluetooth Network Encapsulation protocol (BNEP), which offers IP connectivity directly on top of L2CAP without the need of RFCOMM layer. This provides smaller overhead and better network formation as RFCOMM/PPP combination. We have chosen direct L2CAP operation for Bluetooth connectivity in PeerHood. We can avoid additional overhead caused by either RFCOMM and PPP or BNEP, and still offer ordered and reliable data delivery.

Wireless LAN (WLAN) and GPRS plug-ins for PeerHood both operate over IP connections. The difference between WLAN and GPRS plug-ins comes from the service discovery and connection establishment process. WLAN plug-in uses broadcast-based service discovery and offers direct connections between communicating devices without the need for additional devices. GPRS plug-in uses proxy device as an intermediary. Reason for requiring a proxy with GPRS usage is the usual practice from to provide only private non-reachable address for GPRS clients. The plug-ins have been named after the most appropriate network technology for the plug-in, although they are not strictly limited to those technologies. For example, WLAN plug-in can also be used on IP networks created by Bluetooth BNEP protocol, just by choosing the BNEP network interface.

3.3 PeerHood library

Applications use the PeerHood library both to request information from the daemon and to connect to remote services. The PeerHood library is a software component, which can be dynamically loaded into an application. It provides an interface, which has all the required functions for PeerHood operation. Applications can register their own services to a local PeerHood service database, which is advertised to remote devices on request. Also, service discovery requests to local PeerHood daemon are sent using the PeerHood library.

The library is also used to establish connections to remote services, and to transmit data between the devices. From the application point-of-view, connection establishment and data transmission process is transparent from the chosen network technology. To provide a common interface for several wireless technologies is a secondary objective of PeerHood, primary objective being the ability to monitor wireless neighborhood and to provide information about nearby devices and services. Furthermore, initial support for roaming between different wireless connections has been developed for PeerHood.

4. SERVICE DISCOVERY IN PEERHOOD

Service discovery in PeerHood is *query-based*, and it's done by the PeerHood daemon. During initialization, the daemon loads configured network plug-ins, which start monitoring the wireless neighborhood for other PeerHood devices. When a new device is found, plug-in updates the device entry in the device database managed by the daemon. After that, device and service info can be requested from the device. Each network plug-in has slightly different ways to discover new PeerHood devices. Characteristics and

constraints of each wireless system have been taken into account, and the most suitable method has been selected.

Bluetooth inquiry process and Service Discovery Protocol specified in Bluetooth wireless technology provides us means for discovering Bluetooth devices running PeerHood. We can use a specific Universally Unique Identifier (UUID) for PeerHood. When a device providing this Bluetooth service is found, it's known to be a PeerHood device. We can continue with PeerHood communication tasks. However, Since the Bluetooth service discovery is strongly connected to Bluetooth specification and technology itself, it's not feasible to use it when using other wireless technologies.

Wireless LAN plug-in of the PeerHood uses single hop broadcast messages to find out other PeerHood devices. After starting to listen for replies, PeerHood daemon sends a broadcast message to a predefined port number. PeerHood devices in the same subnet receive the broadcast message and respond to it by sending unicast reply back to the source address. The devices, which responded to request, can be added to the local device database.

The GPRS plug-in is configured to use a proxy, which forwards PeerHood device discovery requests to all connected devices. Likewise, replies to discovery requests are forwarded back to the requesting device. After receiving these replies, requesting device can add new remote devices to common device database. The GPRS plug-in is mainly intended for expanding the PeerHood neighborhood by creating Personal network [11] with pre-defined connections to favorite services. Other use is to provide supporting infrastructure-based technology for roaming between network technologies.

4.1 Service Registration

In order to be found by other PeerHood devices, PeerHood application needs to register its service to the local PeerHood daemon. PeerHood daemon maintains a database of available devices and services. As service location information, a service entry includes the port to connect. Other information registered includes name of the service and the attributes. Currently, little research has been done on PeerHood Service descriptions, but service description templates and definitions are one of the future research areas of PeerHood. Initial survey of related research suggests, that in addition to common service description templates, a system should offer predefined set of typical services, to remove ambiguity regarding the service descriptions. Services should have also user-friendly, human-readable forms, in case of manual service selection by the user. Service location protocol specification [6] defines service location information as universal resource locators (URL), which are of the form *"service:"<srvtype>"://"<addrspec>*. URL's are a well known form and therefore could be a good way to provide human-readable information about the service type and location.

4.2 Service Discovery Process

After daemon has found neighboring PeerHood devices with the technology specific device discovery methods, common sequence of PeerHood service discovery can be carried out. The process of PeerHood service discovery and interaction is shown in Figure 2. To clarify the representation in the figure, the required step of device discovery is omitted. Operation of PeerHood daemon includes constant search for other PeerHood daemons in the neighborhood and sending of information requests to discovered devices. Requested information may include *device info, service info, available network technologies and list of neighboring PeerHood devices already found by the target device*.

When starting up, the PeerHood application (client or server) initializes the PeerHood instance by creating a local socket connection to local PeerHood daemon. This connection is later used to register services to the daemon and to request the list of found devices and services from the daemon. As soon as PeerHood server application has registered its service to the daemon in its own device, that daemon responds to the arriving service discovery requests with the updated service list responses. This is depicted in the Figure 2 with two different service discovery responses sent by the daemon of Device 2. At first, before the registration of the PrintServer application, an empty service discovery response is sent, while after the registration the response includes updated service information with Print server service.

Daemon constantly sends information requests to other PeerHood daemons found in the neighborhood. The contents and the time interval of the requests can be configured by the administrator of the device. For example, we might not need to request list of services every time we discover a certain device. Or, we might exclude the request for the list of neighbors from the information request, if we are interested only for devices and services found nearby.

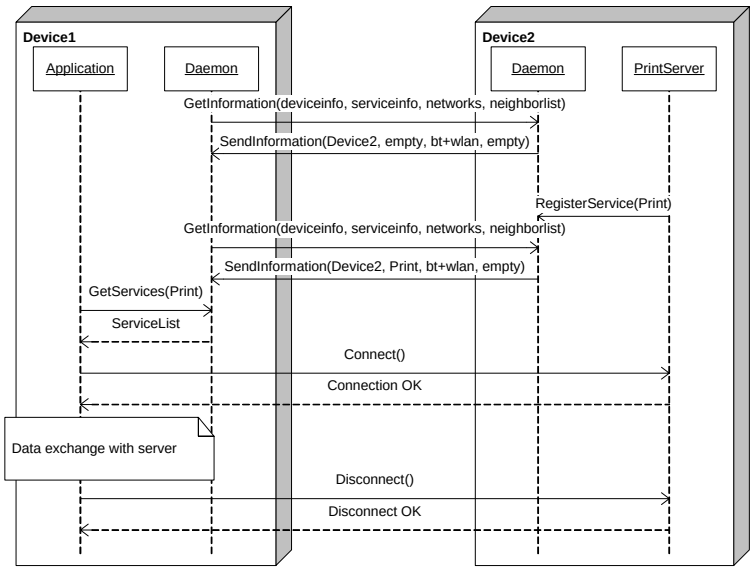

Figure 2 - Service Discovery and Interaction in PeerHood

After the daemon has received information from remote PeerHood devices, it updates device and service information in the local device and service database, managed by the daemon itself. The information is ready to be supplied to the local PeerHood applications. Applications can query a list of all discovered devices and services, or devices providing certain service. Daemon returns the list of locations of requested services. These requests are done using the local socket connection, established during initialization.

All the communication between PeerHood application and Daemon is done by using functions provided by the PeerHood library. Also connection establishment and data exchange between PeerHood applications is done using the PeerHood library. Daemon and library uses network plug-ins as an actual component communicating with other devices.

PeerHood has been designed to utilize extended neighborhood information, which includes the exchange of stored device and service information between neighboring PeerHood devices [12]. Neighborhood information can be used to synchronize the device databases between different devices. It diminishes the influence of failed device discovery attempts, and it can be used also to devote specific device (e.g. personal computer) to perform device and service discoveries on behalf of mobile device. This is carried out in the regular device discovery process, with requesting the list of neighbors in the information request sent between daemons.

4.4 Service Interaction

Of the service selection approaches presented by Tyan [9], we have chosen to define an *interface* as the method for service interaction in PeerHood. For each PeerHood service, the service discovery process results to a location of the service. Location information includes reference to PeerHood plug-in which can be used to connect to device, the technology specific addresses of the remote device, and the port for the target service. Connection establishment based on retrieved location information is done automatically by the PeerHood library. Upon a successful connection, the library returns a connected interface reference to the application. As for the service selection mechanism, PeerHood library provides both automatic and manual ones. Application developers may choose whichever is more suitable. Automatic service selection is done by ordering the PeerHood interface to connect to a desired service type. Manual selection can be done by requesting a list of service entries of certain type from the PeerHood daemon, and selecting the most suitable one from the list. By providing two different methods for service selection, we can overcome the inconvenience caused by the manual service selection, but offer a manual selection when necessary.

5. CONCLUSION

Service discovery is a fundamental part of future network environment. Wireless connectivity places service discovery in an important role. This is even more emphasized in ad hoc connections, with no possibility for permanent centralized directory for services.

We have developed PeerHood, a system for ad hoc networking, providing connections using different wireless network technologies: Bluetooth, Wireless LAN and GPRS. To address the important issue of service discovery in wireless networks, a device and service discovery process has been implemented as one of the central activities of PeerHood. This process considers different requirements and possibilities of wireless network technologies, and provides the most suitable ways of discovering other devices and services. PeerHood also includes a common interface, which can be used to interact with found services, regardless of the underlying network technology.

We have chosen to offer both automatic and manual service selection. This enables both convenient automatic service selection of common services, and on the other hand, more exact manual service selection, if service type requires accurate consideration of service attributes, or if the service attributes include ambiguous information.

REFERENCES

[1] Arbanowski, S. et al., I-centric Communications: personalization, ambient awareness, and adaptability for future mobile services. *IEEE Communications Magazine*, volume 42, Issue 9, pp. 63-69, 2004.

[2] Tafazolli, R. (ed.), *Technologies for the Wireless Future: Wireless World Research Forum.* John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, England, 2005.

[3] Sun Microsystems, Jini Technology Core Platform Specification, 2003. from http://www.sun.com/software/jini/specs/core2_0.pdf

[4] UPnP Forum. UPnP Device Architecture. Document version 1.0.1, July 2006, from http://www.upnp.org/specs/arch/UPnP-DeviceArchitecture-v1.0-20060720.pdf

[5] Bluetooth SIG. Specification for Bluetooth technology. version 1.1, 2001.

[6] Guttman, E. et al., Service Location Protocol, Version 2. RFC 2608, 1999. from http://www.ietf.org/rfc/rfc2608.txt

[7] Clement, UDDI Version 3.0.2, 2004. from http://uddi.org/pubs/uddi_v3.htm

[8] Zhu, F. et al., Service Discovery in Pervasive Computing Environments. *IEEE Pervasive Computing*, Volume 4, Issue 4, pp. 81-90, 2005.

[9] Tyan, J. and Mahmoud, Q. H., A comprehensive service discovery solution for mobile ad hoc networks. *Mobile Networks and Applications*, Volume 10, Issue 4, pp. 423-434, 2005.

[10] Porras, J. et al., Peer-to-peer Communication Approach for a Mobile Environment. *37th Annual Hawaii International Conference on System Sciences*, 2004.

[11] Niemegeers, I. G. and Heemstra de Groot, S. M., From Personal Area Networks to Personal Networks: A User Oriented Approach. Wireless Personal Communications: An International Journal, Volume 22, Issue 2, pp. 175-186, 2002.

[12] Hämäläinen, A. and Porras, J., Enhancing Mobile Peer-to-Peer Environment with Neighborhood Information. *Proceedings of the 3rd Workshop on Applications of Wireless Communications*. Lappeenranta, Finland, pp. 39-48, 2005.