Service Discovery in TinyObj: Strategies and Approaches

Pavel Poupyrev, Takahiro Sasao, Shunsuke Saruwatari, Hiroyuki Morikawa, Tomonori Aoyama
The Graduate School of Frontier Science, Tokyo University
7-3-1 Hongo, Bunkyo-ku,
Tokyo, 113-8656 Japan
{pavel, sasao, saru, mori, aoyama}@mlab.t.u-tokyo.ac.jp

Peter Davis
Adaptive Communications
Research Laboratories, ATR
2-2-2 Hikaridai, "Keihanna Science City"
Kyoto, 619-0288 Japan
davis@atr.jp

ABSTRACT

In this paper, we describe TinyObj service discovery model, which uses short wireless packet broadcasts for service discovery within a user's vicinity. We present a prototype implementation for the discovery model including the hardware and software development of a wireless discovery device. The developed wireless device, called Buoy, can be used as an independent service discovery device or as an attachment for cellular phones. The aim of the design is to provide the basic functionality for service discovery in ubiquitous environment. The developed software includes a uniform user interface that makes the system expandable and customizable. Indeed, a user can easily add, advertise, discover and remove any services in the system.

1. INTRODUCTION

Discovery of information is an essential problem in our daily lives. One of the easiest ways of finding information is using an Internet search engine that replies to user's queries with global search results. However, this way of information discovery provides little connectivity with the user's surrounding world since the search is conducted over a worldwide information database.

A new discipline, called ubiquitous computing, has been proposed [1-2]. Ubiquitous computing provides to a user a better connection with its surrounding world. In fact, this is achieved by embedding small computing devices in every object of the physical world. These devices can be equipped with short range wireless modules making the discovery of other devices and their services accessible to a user. We define this discovery as a real-world service discovery in proximity.

The real-world service discovery in proximity is different from the conventional ways of discovering services. Currently, a user discovers services upon request on the Internet or using other media such as newspapers. However, discovery in ubiquitous environment is performed in the background. Indeed, a user discovers services while moving from one place to another. This allows a user to find services in new locations or even newly advertised services in common visited places. The real-world discovery very well fits with the concept of ubiquitous computing where a user interacts with the real-world using background wireless computing devices.

A good example of a real-world service discovery in ubiquitous environment is the discovery of an ATM service. In this scenario, a user wants to find an ATM that belongs to a particular bank or provides VISA credit card operations. In order to do that a user has to define the ATM service discovery preferences specifying discovery details such as the bank name. When a user comes in proximity of an ATM, the user's device discovers the presence of the ATM. If a discovered ATM matches the specified preferences then the device alerts the user about its discovery.

The main contribution of this paper is the proposal of the *TinyObj model* for real-world service discovery in a ubiquitous environment and its *prototype implementation*. Since wireless ubiquitous devices have scarce resources, such as battery and processing power, we use a broadcast protocol eliminating the necessity for a complex routing protocol.

The prototype implementation has been developed including both software and hardware modules. We developed a wireless device for service discovery that can be used as an attachment with cellular phones or as an independent wireless discovery device. The developed discovery device is based on an original CSMA/CA-based MAC protocol.. The designed software provides to a user the ability of adding, advertising, discovering and removing new services using a uniform graphical user interface, unlike other discovery systems, which require knowledge of a programming language.

We will describe our prototype implementation covering three elements of the system:

Service matchmaking: The number of provided services usually greatly exceeds the number of services a user needs. The system should provide a mechanism for receiving focused search results. Also, service matchmaking should have a general way to represent any service. A user should be able to easily add and remove services from the system.

User Interaction: Our service discovery is meant to support two types of mobile users: a user with a cellular phone and a user with a wireless discovery device. One of the requirements is to provide easy user interaction with the device for both types of users.

Broadcast Protocol: Broadcasting is an important component of the system and should provide efficient data dissemination and consume as less power as possible. In this paper, we only discuss the requirements for the broadcast protocol.

The rest of the paper is organized as follows: In the next section we will present related works followed by an explanation about the TinyObj discovery model. In section 4 we will explain our implementation prototype, covering three major service discovery elements: service matchmaking, user interaction and network protocol approaches. In the last section we will conclude our work and describe directions for future work.

2. RELATED WORKS

A few systems, which support service discovery, based on short wireless communication, have been proposed such as nTAG [3], SpotMe [4], IntelliBadge [5], Conference Assistant [6], and Proxy Lady [7]. The goal of these systems is to empower users with additional features at conferences and meetings such as the discovery of friends, events or announcements, etc. Unlike these systems, which comprise pre-defined set of services, we target the development of extendable service discovery system where a new service can be easily added, advertised, discovered and removed.

Other research works, SLP [8] and JINI [9], provide a platform with an API for service discovery. However, implementation of service discovery requires knowledge of the programming language, such as Java or C/C++. In comparison, we focus on the development of a service discovery system that will provide a uniform graphical user interface as well as an API. Moreover, SLP and JINI systems are based on TCP/IP protocol, which requires computing devices with powerful processor, large memory and sufficient battery. Ubiquitous devices have strict limitations on available resources. Thus, it makes it more complicated to use these solutions.

Other research, such as ECA [10] and Smart-its [11], focuses on building hardware and software for ubiquitous computing. These projects provide a solution for quick implementation of new applications in ubiquitous environment where wireless devices are equipped with sensors and actuators to perceive context information. However, these projects do not particularly focus on service discovery issues in the ubiquitous computing.

3. DISCOVERY IN TINYOBJ

In this section we describe the TinyObj discovery model. We will use the same ATM service scenario to present TinyObj discovery. The TinyObj discovery model consists of four elements: 1) data initialization; 2) wireless data exchange 3) service matchmaking and 4) discovery alerts.

First, a user performs *data initialization*. A user can initialize two types of data: advertisements and preferences. For example, when a user wants to find an ATM service, a user specifies preferences (a filter) for service discovery as follows: only ATMs which support VISA credit card with a withdrawal fee of less than $1.5. In the meanwhile, a service provider, who advertises a service, may define a service advertisement like: Citibank ATM service, credit cards supported are VISA, MasterCard, and the withdrawal fee is $1.05.

Secondly, after the data has been initialized the *wireless data exchange* processes may begin. The wireless data exchange is based on sending periodic data broadcasts. It is not necessary for all participants to broadcast data.

Participants can be either active or passive. Active participants periodically broadcast data in proximity and passive ones save data on the wireless device without broadcasting. Normally, a service advertiser and a user can select whether they want to use active or passive discovery strategy.

In the ATM scenario, the bank uses an active strategy because it is interested in service delivery to the users. The users may instead use a passive strategy and store a filter on the device. The device tracks all services which match the stored filter.

However, it is possible to broadcast not only advertisements but also preferences. In this case, an advertiser receives preferences, which are compared with all advertisements stored in the device's database. Therefore, both a service advertiser and a discoverer can accomplish the discovery.

TinyObj concept is based only on broadcast transmissions. Thus, when a wireless device discovers a service it does not provide a method to reply to the sender using the same media. A user can access the sender via other media such as e-mail, phone, or web, using the contact information included in the packet.

Thirdly, after participants decide whether to be an active or passive discoverer, the *service matchmaking* process matches all received broadcast packets with the locally stored data. When a device receives an advertisement it is compared with all stored filters. Similarly, if a device receives preferences it is applied to advertisements stored on the device. The service matchmaking engine provides a function that takes an advertisement and preferences as parameters, and returns a comparison result in the form of a true or false value. If the returned result is true then a service matchmaking engine has succeeded in discovering.

Forth, after service matchmaking detects a match a device notifies a user with a *discovery alert* such as sound, vibration, etc. After viewing the content of the discovered service a user can access a sender using the contact information included in the data packet.

4. PROTOTYPE

Currently, the TinyObj system is implemented as a working prototype. We have built a wireless discovery device, called Buoy (Fig. 1). Also we have developed the device interaction software for cellular phones and personal computers.

4.1 Buoy

The Buoy device provides minimum functionality for service discovery. This includes storing/removing advertisements and preferences to/from the Buoy local database, performing service-matchmaking, communicating with a cellular phone or a PC, and broadcasting data.

Fig 1: Three variations of the wireless discovery device: the wireless device with an add-on component, a cellular phone and cradle

The discovery wireless device consists of two components namely a wireless module and an add-on component (Fig. 2). The wireless module includes the following components: a battery, an antenna, an Atmel ATmega128L processor, a Chipcon CC1000 wireless module, and two serial interfaces where one can be used by a cradle and another can be used by a cellular phone. The add-on component connects to the wireless module to provide a primitive input/output user interface equipped with a buzzer, two LEDs, a button and an Infrared port. The dimension of the device is 62x40x15mm.

4.2 Service Matchmaking

The service matchmaking is a vital component for successful discovery. The service matchmaking defines a format for service data and provides an algorithm for comparing advertisements and preferences. The format includes a list of name-value attributes to describe the service. This format enables the creation of services. However, our proposed system does not provide any access methods to the service but provides only service availability information. For example, if a user discovers a printer service, it may contain an IP address that can be used to reach the printer. The service matchmaking algorithm should provide narrowed search results because it is important to minimize the number of false alerts. For example, in an ATM service scenario, a user wants to discover ATM services satisfying some search preferences.

The system uses a service descriptor (Fig.3) that serves as a template to define advertisements and preferences. Also, the service descriptor includes information so that the TinyObj software knows how to represent data in a common way.

Fig. 3 shows an example of a service descriptor for an ATM service. The general parameters for a service descriptor are a service name, a description and a category. Attributes are optional parameters, which allow a user to provide more detailed information about a service. The attributes are useful for narrowing search results since they can specify search preferences more precisely. An attribute includes a name, a type, default values, a display and required options.

The type defines a data format used by the software to verify data input correctness and to represent the data. The type can be selected from the available types. The supported types in the current implementation are: Integer, Currency, Boolean, Select,

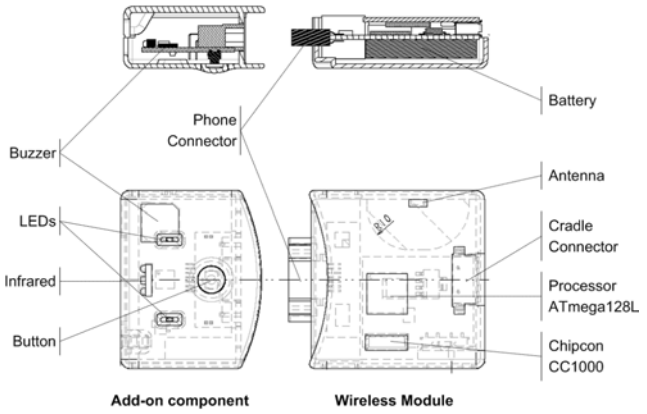


Fig 2: Buoy device schematics

Service name: ATM **Category:** Finance→Bank
Description: ATM service discovery

Attribute Name	Type	Default Value	Display	Required
Bank name	String	VISA, Master, Plus	☑	☑
Credit cards	Select		☐	☑
Availability	Select	Open, Closed	☑	☐
Charge fee	Currency		☑	☐
Location	String		☐	☐

Fig 3: A service descriptor example for an ATM service

and String types. The default value provides initial values when a new advertisement or filter is created.

A display option is used by the software for partial representation of advertisements. For example, Fig. 4a represents a list of discovered advertisements. Two ATM advertisements, Citibank and Mizuho, have partial representation depicting only three attributes: Bank name, Availability and Charge fee. In this case, TinyObj software represents only attributes, which have the display option enabled in the service descriptor.

Thus, each service has a service descriptor that serves as a template for the creation of advertisements or preferences. TinyObj software allows a user to create a new service descriptor using a graphical user interface specifying a service name, a category, a description and a list of attributes.

4.3 User Interactions

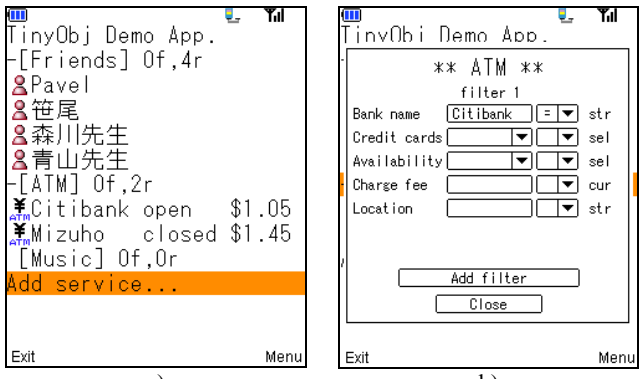
The TinyObj system is designed mainly for two types of users: a cellular phone user and a user that uses only the Buoy device without attaching it to a cellular phone. In this section we will describe interaction for both types of users.

4.3.1 Cellular phone user interaction

In Fig. 1, the Buoy wireless module is attached to a cellular phone. In order to advertise or discover a service, a user first has to store a service descriptor on the Buoy device so that the Buoy software knows how to handle a service.

In the current version of the TinyObj prototype, all created service descriptors are stored in the web-enabled database. A user can search for a necessary service descriptor using keyword searches or category searches.

After discovery of the service descriptor a user can create



a) b)

Fig 4: A cellular phone screen shots of TinyObj application a) a list of services registered with cellular phone b) an ATM service filter creation

advertisements or preferences. When using service descriptors, the software can represent a form to assist the user in the creation of an advertisement or preferences. For example, Fig. 4b represents the case when a user creates a filter for discovering any ATM services that belongs to Citibank. After inputting the data, the user then specifies whether to choose an active or passive discovery mode. Finally, a user uploads and stores the input data and the service descriptor. Uploading a service descriptor is necessary if a user wants to create or edit the data later. Fig. 4a represents an example of stored service descriptors on the Buoy device including Friends, ATM and Music.

After all these steps, a user is able to start discovering. When a new service is discovered, the phone alerts the user through sound or vibration depending on the phone's settings.

4.3.2 Buoy device user interaction

A user who does not have a cellular phone is still able to discover services in proximity using only Buoy. However, the interaction requires an additional device, such as a PC or a cellular phone, only when initializing data or viewing discovered services. The actual discovery does not require the use of any extra device.

Buoy consists of two modules (Fig. 2): a wireless module and an add-on component. The add-on component has a primitive input/output interface with two LEDs, a button and a buzzer. Since the interface is very limited the use of an extra device becomes necessary in order to initialize the device as well as to view the discovered data. A user is provided with web-enabled PC software that can be accessed from the Internet. To initialize and view data with a computer, the Buoy uses the Infrared port..

Thus, the user interaction with Buoy works as follows. First, a user, using the web-enabled PC software, searches for a service descriptor. After finding the necessary service descriptor, a user creates an advertisement or a filter based on its content. Then, a user specifies whether to use active or passive discovery. Unlike cellular phone user interaction, a user selects an alert, such as flashing LEDs or a buzzer sound, for discovery notifications. Finally, a user uploads and stores these settings on Buoy using the Infrared port interface.

When Buoy discovers an advertisement, it notifies a user with the predefined alert associated with the service so that a user knows which type of service has been discovered. In order to view the discovered data a user has to use a desktop PC or a cellular phone.

4.4 Broadcast protocol

The TinyObj model is based on a broadcast packet exchange. Broadcast packets can be received by all neighboring nodes, which are in range of the transmitter. In our system the broadcast protocol does not provide a two-way communication scheme like usual routing protocols based on the query/reply model. This model has been selected to overcome the complexity and the limited scalability of typical packet routing protocol implementation especially in the case of highly dynamic mobile networks.

Broadcasting is a simple protocol that can be easily tuned to the application requirements. We define two requirements for our broadcast protocol, which are: 1) efficient packet dissemination and 2) minimization of power consumption. Efficient packet broadcasting is necessary because with the increase in the number of broadcasts the collision rate increases that leads to a poor packet delivery rate. The current prototype is based on CSMA/CA protocol that consumes large amount of power while listening to a channel. If we assume a constant current drainage of 110 mAh, the battery would last for approximately 6 hours.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we described TinyObj service discovery model, which uses short wireless packet broadcasts for service discovery within a user's vicinity. The broadcasting provides a simple way for data exchanges. We have presented a prototype implementation of the TinyObj model including both the software and the hardware. The hardware comprises the Buoy device that provides minimum functionality for service discovery. We also described user interactions with the Buoy device. The TinyObj software has been designed in order to provide a uniform graphical interface so that a user can easily add, remove, advertise and discover new services.

Future work would include studies on the implementation of power-efficient broadcast protocol and power-saving mechanisms. Moreover, the user interactions with the service discovery software should be improved to provide a generic platform. In order to do so, we are planning to conduct scenario field studies, adding new features to the system required by the different scenarios.

6. REFERENCES

[1] Mark Weiser, "Hot Topics: Ubiquitous Computing" *IEEE Computer*, October 1993.

[2] Mark Weiser, "Some Computer Science Problems in Ubiquitous Computing," *Communications of the ACM,* July 1993.

[3] http://www.ntag.com/, 2005

[4] http://www.spotme.ch/, 2005

[5] Donna Cox, Volodymyr Kindratenko, and David Pointer, "IntelliBadge™: Towards Providing Location-Aware Value-Added Services at Academic Conferences," in *UbiComp 2003,* Seattle, WA, USA.

[6] Anind K. Dey, Daniel Salber, Gregory D. Abowd and Masayasu Futakawa, "The Conference Assistant: Combining Context-Awareness with Wearable Computing," In *3rd International Symposium on Wearable Computers*, 1999.

[7] Per Dahlberg, Fredrik Ljungberg, and Johan Sanneblad, "Supporting Opportunistic Communication in Mobile Settings," in *CHI 2000 Extended Abstracts on Human Factors in Computing Systems*, 2000.

[8] Guttman E., Perkins C. and Kaplan S., "Service Location Protocol," RFC 2608

[9] Edwards W. K., "Core Jini," *Prentice-Hall*, 2001.

[10] Tsutomu Terada, Masahiko Tsukamoto, Keisuke Hayakawa, Tomoki Yoshihisa, Yasue Kishino, Atsushi Kashitani, and Shojiro Nishio, "Ubiquitous Chip: A Rule-Based I/O Control Device for Ubiquitous Computing," *Pervasive 2004,* pp. 238-253, 2004

[11] M. Beigl, and H. Gellersen, "Smart-its: An Embedded Platform for Smart Objects," *Smart Objects Conference (sOc),* 2003.