Wireless service developing for ubiquitous computing environments using J2ME technologies

José Miguel Rubio
Escuela de Ingeniería Informática
Facultad de Ingeniería, PUCV
Valparaíso, Chile
jose.rubio.l@ucv.cl

Claudio Cubillos
Escuela de Ingeniería Informática
Facultad de Ingeniería, PUCV
Valparaíso, Chile
claudio.cubillos@ucv.cl

***Abstract*—From some years ago, technologies are ubiquitous, omnipresent and integrated seamlessly in our common activities. There is a need for developing services and applications that use all the capabilities of mobile devices available, for connectivity and data processing. Java 2 Micro Edition (J2ME) technology develops these services for use in a wide range of devices, with portable code, and compatible with several classes of hardware and software. Wireless communications technologies (WiFi, Bluetooth) allow devices to communicate with their environment. Bluetooth technologies are becoming massive, and are present in the majority of these devices. This kind of connectivity is possible using the J2ME Application Programming Interfaces (APIs), but it requires some expertise and a wide knowledge in order to implement these components. This document proposes a framework focused towards the service development using bluetooth technologies with more simplicity. Also, it presents the design and implementation of a mobile information service that uses this framework in a ubiquitous environment.**

***Keywords*-ubiquitous computing; wireless services; service discovering; bluetooth; mobility.**

I. INTRODUCTION

Using technology in daily tasks today is a very common aspect of our lives, leaving behind the fact that we interact with several classes of devices in every activity we do. This was the vision of Mark Weiser [1], 18 years ago. He wrote about environments that integrate computing with common tasks, allowing the user to interact in natural ways making computers nearly imperceptible. He designated this integration as Ubiquitous Computing, talking about the ease of access and availability of these technologies. This vision is materialized in the proliferation of a wide range of devices (mobile phones, Personal Digital Assistants: PDAs) and different connectivity options, (Short Message Service: SMS, Bluetooth, GPRS: General Packet Radio Service, EDGE: Enhanced Data rates for GSM of Evolution), giving multiple interaction methods. In order to integrate these devices with our common tasks in a more profitable way, there are several technologies centralized in building applications and services for this class of devices. One of them is the J2ME technology.

The next sections describe a simplified approach for using J2ME technologies available for mobile device development. A framework is proposed with components and development directions for this class of devices. Different practical applications are proposed. Finally, a development of a mobile information service is exposed, based on this framework, mixing wireless and web technologies to access information in a localized context.

II. THE J2ME PLATFORM

J2ME is a technology developed by Sun Microsystems to build applications for mobile devices. It is different from other major editions of Java (Java 2 Standard Edition (J2SE), Java 2 Enterprise Edition (J2EE)); basically in a smaller group of functionality that fits the computing, memory and data storing capabilities of limited devices.

A. Configurations and Profiles

To make a difference between different levels of capacities, there are some configurations (minimal technological requirements) and profiles (basic functionality and services) to differentiate between device classes. A common configuration is CLDC (Connected Limited Device Configuration), with minimal requirements of memory, power consumption, and some kind of wireless connectivity, generally HTTP (Hypertext Transfer Protocol) and SMS [2]. The common profile adopted by mobile devices is MIDP (Mobile Information Device Profile), that defines some required functionality about user interface, data persistency, application lifecycle and multimedia control [3].

B. User Interface

The user interface in J2ME offers a limited group of components, based on limitations of screen size and reduced keyboard functions of mobile devices. The user interface is limited to selection lists, basic forms, text input dialogs, and low-level screen painting (Canvas). The user inputs are command buttons and an alphanumerical keyboard.

 ISBN: 978-1-61208-100-7

C. Connectivity

Different connectivity options have a centralized access through Generic Connection Framework (GCF) APIs. GCF allows connecting with any connection available (HTTP, SMS, Bluetooth, local file system) using a unified API. Each connection implemented is based on a generic connection class, and extended to support specific functions related to each particular connection implementation.

D. Optional Packages

To provide additional functions over the basic configurations and profiles, there are some optional packages that access extended functionality, standardized in different Java Specification Request (JSR). Table I lists typical optional packages for mobile devices.

Table I
SOME OPTIONAL PACKAGES FOR J2ME

Specification	Description
JSR-75	File system access and PIM
JSR-82	Bluetooth
JSR-120	Wireless Messaging (SMS)
JSR-135	Multimedia API
JSR-172	Web Services

E. J2ME-Enabled Devices

J2ME is the most available platform in the market. There are several mobile devices today that allow executing J2ME applications, basically mobile phones, smartphones, and PDAs. In Windows Mobile (Pocket PC) systems a commercial J2ME suite must be installed.

F. Application and Service Development in J2ME

There is a variety of tools and programming environments that support the complete mobile development cycle: design, coding, compilation, preverification, packaging, tests, and deployment. The main suite for J2ME programming is the Wireless Toolkit from Sun Microsystems. Programming environments like Netbeans and Eclipse, integrate this toolkit to allow developing J2ME applications. Emulation utilities allow verifying applications for different device models. Deploying applications to mobile devices are performed through direct transfer (infrared, bluetooth) or a web download. Usually, a JAR file (Java ARchive) is provided with all the application components and resources. In some cases, a JAD file (Java Application Descriptor) is required to simplify the install process.

III. J2ME CONNECTIVITY

Mobile devices have a lot of connectivity options, even more than some major device classes. This is one of the main reasons for developing applications for J2ME-enabled devices. A typical mobile phone can be converted into a client's email, web browser, game console, multimedia player and even run business applications and perform online commercial transactions.

A. HTTP/HTTPS Connectivity

In a typical J2ME device, at least HTTP connectivity is required. The majority of online services are web-based. The current increase of web technologies allows access to a variety of contents (information, images, audio/video). Some mobile devices cannot support all of them due to it's limitations in processing, memory or screen size. Many online services (email, social networks) have a "mobile" version or supply a J2ME client application.

B. Bluetooth Connectivity

Bluetooth is a wireless communication technology oriented to connect different classes of device in a reduced area (10-100mts), using low power consumption and a minor complexity compared with other wireless technologies like WiFi. Specified initially for the Bluetooth Special Interest Group (SIG), composed by several industry leaders. The current specification adopted its version 2.1 [4]. From their first release, it has become a standard industry for wireless transfer and personal area networks (PAN). The majority of new mobile devices have included this technology, allowing data transfers, network connections, use of headsets and other wireless peripherals [6].

The bluetooth specification allows inter-operating between independent systems, defining messages for each layer of the protocol stack (high and low level), and between internal subsystems (controller and host). To ensure interoperability between applications some bluetooth profiles are defined to standardize mechanisms and protocols for different applications like file transfer, networking, printing, or synchronization. Each profile defines messages, procedures and rules for the most common services. The Generic Access Profile (GAP) is the basis for all other profiles. Figure 1 shows the hierarchy of basic bluetooth profiles [5].

Several mobile phones and PDAs offer standard bluetooth services and integrate these services with personal information management (PIM) utilities like contact management, tasks and calendar events. There are many potential uses for taking advantage of those functions, developing custom services to allow users to interact with their environment using these devices.

C. JSR-82: Java APIs for Bluetooth Wireless Technology (JAWBT)

J2ME platform offers an additional API for accessing the bluetooth technology: the JSR-82 optional package. Motorola defined the original specification in 2002. Based on the core bluetooth specification, their objective is *to define a standard set of APIs that will enable an open, third-party*

 ISBN: 978-1-61208-100-7

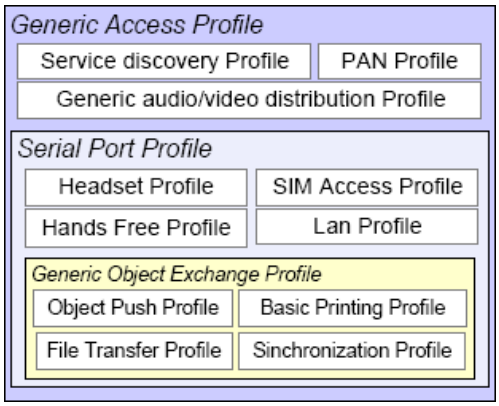

Figure 1. Main Hierarchy of Bluetooth Profiles

application development environment for Bluetooth wireless technology [8].

JAWBT defines a set of components to represent two aspects of the bluetooth specification: core elements and the Generic Object Exchange Profile (GOEP, also known as OBEX). The javax.bluetooth package contains interfaces and classes that represent the core specification: devices, data structures, service discovering, including the L2CAP and SPP protocol. The javax.obex package is another independent set of elements that meets the specification for OBEX connections: header sets, authentification, sessions and operations for client/server connections [5].

IV. Framework for Bluetooth-Based Service Development

The main motivation behind this framework is the high potential of bluetooth technology as a mean for offering extensive functionality in ubiquitous environments, using common devices, standard protocols and data formats, ensuring high interoperability in this class of environments. This section shows the proposed framework for development of bluetooth-based services. The main goal is to provide ready-to-use components in applications that need bluetooth connectivity.

A. Bluetooth Service Development

The JSR-82 API offer the core components for accessing bluetooth technology, nevertheless, this does not provide any direct implementation of common bluetooth profiles. Each developer must implement standard or custom services based on these components. To develop bluetooth based services it is required some knowledge of the technology, and API specifications. There are a few public projects that offer these kind of components, but examples for coding bluetooth services are basic and limited in functionality, and do not provide full implementation of some basic and useful services, leaving the implementation of these class of services out of scope for some projects that would need it.

The goal is to define a common framework for direct access of common bluetooth services, to quickly develop any kind of bluetooth applications. The initial approach is to get a set of components that implement basic bluetooth profiles (Serial Port Profile, Object Push Profile), giving the chance to extend and create new custom services based on the core services. Also, it is defined for some basic directions using patterns and rules to make stable and optimal services. The final goal is to centralize the development efforts in the main application, abstracting bluetooth detailed elements. All components are based in the J2ME APIs: Generic Connection Framework and Bluetooth.

B. Framework Components

The main areas of functionality that include this framework are the following:

1) Bluetooth constants and data: Direct access to main constants, attributes and UUIDs (Unique identifiers for known services and protocols) defined by the bluetooth specification.

2) Service publishing: Defining and publishing standard and custom services.

3) Service discovery: To enable discovering and selection of remote services, access to attributes and properties and to generate the service connection.

4) Bluetooth service connection: To provide the basic implementation of service connection and for init custom data transfers.

5) Logging and debugging: To help developers keep track of the testing executions. Logging capacities for debugging are implemented.

All components are based in J2ME APIs: javax.microedition.io, javax.bluetooth and javax.obex. The main package (btlibrary.core) includes the only core components. Additional packages add specific implementations.

C. Implementation

The core library components implement all basic functionality areas, providing the basis for implementing or extending additional elements. Figure 2 shows the core classes and interfaces. To show some examples of the main components and their logic, the following sections describe the main relation and dependencies of these components, in typical tasks like service publishing, service discovery and client connections.

1) Service publishing: To publish a bluetooth service it is required the following tasks:

- Setup the device as visible (discoverable).
- Identify the service (name, UUID, attributes).
- Wait for connections.
- Process each incoming connection.
- End the service.

 ISBN: 978-1-61208-100-7

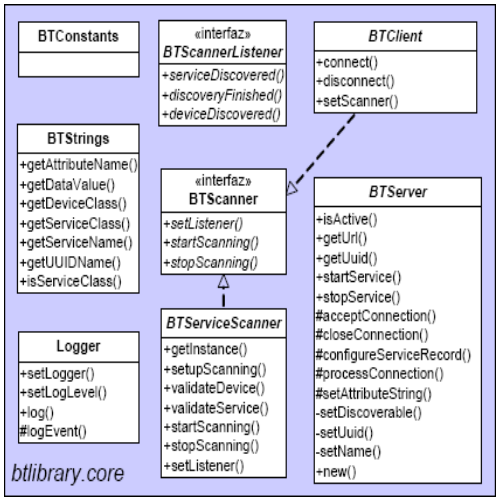

Figure 2. Classes and components of btlibrary.core

The class that implements this functionality is BTServer. It provides methods for creating, identifying, starting or stopping, and processes incoming connections, abstracting the internal logic for these tasks. Figure 2 shows its main methods.

2) Service discovery: The main tasks to discover remote services are:

- Setup the discovery search pattern (the service UUID and attributes required).
- Start the device discovery (or get a cached list of previously known devices).
- Start a service discovery for selected devices.
- Select a service discovered and get the service address.

The class that implements this process is BTServiceScanner. This class notifies applications while new devices and services are found, through the BTScannerListener interface. Figure 3 shows some relations and messages to accomplish this task, and the JSR-82 API calls involved.

3) Connecting bluetooth services: As a result of discovery service tasks available service addresses are obtained. To connect some some of these services we need the service address. BTClient class implements this functionality, including the discovery functions, allowing connection and use of the service operations available.

As an example, a bluetooth file transfer profile service (FTP OBEX) offers functions for browsing remote folders, sending, downloading or deleting files. The ObexFtpClient class, based on BTClient, implements these functions. This bluetooth client offers access to file transfer functions using simple calls, abstracting the protocol and specification logic. Figure 4 shows client operations (connection, file listing and transfer) and API calls involved in relation to bluetooth APIs.

4) User interface definitions: An additional topic is a definition of a generic user interface for use at interactive

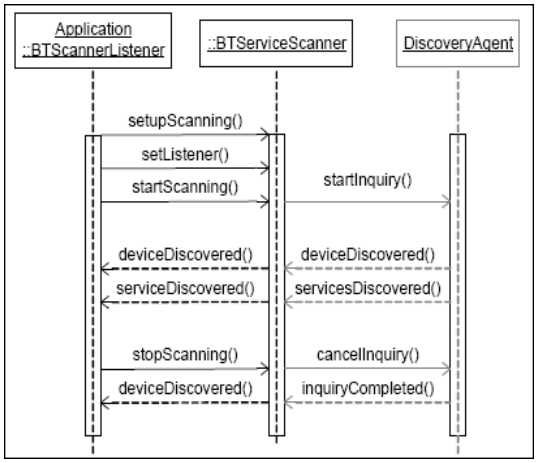

Figure 3. API calls involved in service discovery.

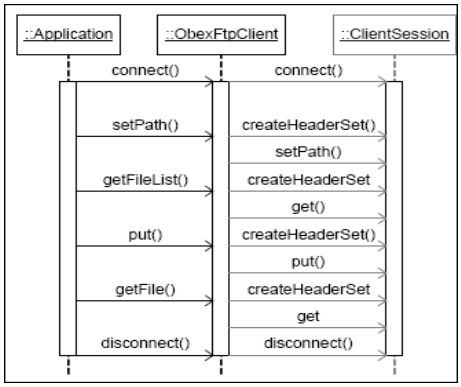

Figure 4. Message sequences in a OBEX FTP service.

service discovery applications, in order to provide a common GUI to control this process, to show discovery progress and status. A graphical service selection interface must provide the following functions:

- To show/hide the selection dialog.
- Show available devices/services while they are discovered (or previously discovered).
- To control the service discovery process (canceling, restarting).
- To select a device from the dialog list.
- To close the dialog and cancel the selection.

The formal definition is implemented in the ConnectionDialog interface. Additionally, there has been implemented a specific dialog for J2ME user interface (ConnectionDialogME), that looks like the model shown in Figure 5.

5) Design Patterns: In order to make an easier service design and application integration, there should be patterns to develop bluetooth services and clients to help the process to be simpler and faster. In general, there are simple steps to accomplish in order to get a successful development, adding basic and advanced examples in order to guide developers.

 ISBN: 978-1-61208-100-7

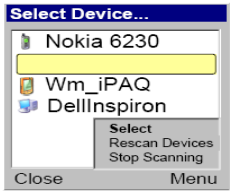

Figure 5. Service discovery connection dialog for J2ME.

As an example, the service creation and publishing process needs to consider some basic patterns and directions:

- To define the base/reference bluetooth profile in order to build the service: generally, a serial port profile (SPP) or an object exchange (OBEX) profile.
- To define the service identifier (UUID). If it is planned to implement some standard bluetooth service, it must assign the specific UUID designated.
- For standard bluetooth profiles or services, it must follow all the official requirements defined to ensure interoperability.
- For custom services, the primary task is to define the public interface, constants and data structures used to exchange messages.
- To optimize connection resources and bandwidth, data transfers must try to use the maximum packet length available, avoiding small data transmissions.
- Due to limitations in the number of concurrent connections, timeout mechanisms are required to avoid blocking new connections when some previous connection is lost or is still awaiting data.

V. Application Fields

Bluetooth services development can have multiple applications. There are several projects related to bluetooth technologies and ubiquitous environments. All of them could be simplified using the components of this framework. Some typical application domains are presented in next sections.

A. Localized Information Services

An information service can provide or process data using the user location as a context. For example a project that allows a museum visitant to receive instant information (text, images, audio/video) about paintings or other things located near the current location or room using his mobile phone with a J2ME application via bluetooth. The key aspect for success is to identify the users and manage profiles, to customize messages and the users experience.

B. Local Environment Interaction

Users can interact with the local environment when some bluetooth-enabled devices are located near the user. For example: a teacher enters the classroom and uses his mobile phone to transfer the class slides and then starts and controls the presentation using a bluetooth service running in a local computer connected to a media projector.

C. Machine to Machine Communication (M2M)

Due to high interoperability and compatibility of bluetooth protocols and profiles, it is possible to implement several classes of device-to-device communication: synchronization, data exchange, collaborative work and distributed applications.

D. Communications and Networks

In this area we may find extended applications to add new communication possibilities in a localized environment. Chat or instant messaging applications, networked games, Internet access for email, news, web browsers, database access, remote access, audio/video streaming. Many internet-based services are possible using a bluetooth service as a gateway.

VI. Practical Application in a Mobile Information System

As a demonstration of use of this framework in a real application for an ubiquitous environment, there has been implemented a mobile information system for academicals and communicational purposes. This system provides the following functions to teachers and students:

- To publish general or group announcements (for teachers, students, class groups, student groups, workgroups)
- To publish calendar events to achieve best attendance to key activities and to complete important tasks.
- To send and receive private messages.
- To store a global directory to access contact information from users (phone, email).
- To associate users with their mobile devices, in order to receive notifications directly.
- To register user profiles and groups, ensuring security and customized information access.

As the user interface for the service, there are some alternatives:

- A web interface accessible via common web browsers. With XHTML compliance allow mobile web browsers to access the portal content in a simple format [7].
- A J2ME client application installed on mobile devices to access portal content with a simplified browser with http connection using a local bluetooth service acting as gateway, giving some extra bluetooth-based features.
- A message interface using a local bluetooth service that receives commands to process and trigger some type of action, like changing user configurations or requesting contacts.

A. Service Architecture

There are three main areas of implementation for this service. Figure 6 shows the schematic composition of the complete service.

 ISBN: 978-1-61208-100-7

1) Data Server: A typical HTTP server is the web content provider and the data interface for some service components. A database server is the global data container. Only one computer acts as a data server.

2) Bluetooth Service: The bluetooth service is composed by different minor bluetooth services and processes for specific tasks: a HTTP gateway service for processing web requests and responses; a notification agent that discovers nearby devices, retrieves messages, news or events associated to the owner's device and sends only new notifications to each discovered device; a command reception service reads messages sent by mobile devices to request information, update profile properties and obtains objects like contacts, applications or events. There can be multiple instances of this bluetooth service installed in any machine connected to the Internet and with a bluetooth installed device.

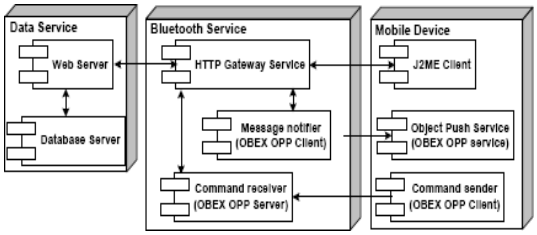

Figure 6. Mobile information service architecture.

3) Mobile Device: It is the main user interface for this service. It can get notifications automatically when it enters in a bluetooth service area, using the internal object push service activated in each device. Generally, this service integrates the reception of special objects with some "organizer" functions. When it receives some contact, event or task they are automatically saved or scheduled.

Also it allows sending commands to any bluetooth service to request or set information, seeking online contacts, messages and resources and receives via a bluetooth the transfer of the required content to get the J2ME client application and update the contact list.

Finally, the J2ME client application installed in this device can access the web site using the http gateway provided by any bluetooth service in range, allowing direct access to the web content in a simplified interface and providing additional mobile and bluetooth related functions directly like trigger phone calls or download content.

VII. Conclusions

J2ME technology allows developing services for a wide range of devices. Bluetooth connectivity is highly supported in newer devices and can interoperate with a variety of device classes due to industry-adopted standard protocols and formats. JSR-82 API is the access point to bluetooth technology in J2ME. Bluetooth service development through this API is a complex task.

This framework can simplify the service development providing direct access to common application requirements in bluetooth-enabled applications: service discovery, publishing and connecting to remote bluetooth services. There are many possible applications of this technology for ubiquitous computing environments: localized information services, environment interaction, machine-to-machine communication and online and network applications.

Through developing a mobile information service based on this framework shows one of many applications of bluetooth technologies to common tasks. In this service we have applied all aspects covered by the framework: creating, discovering and connecting to standard and custom bluetooth services. While in the design and implementation phases some elements of this framework were improved, completed and fixed. The final set of functionality provided by the framework has been enough to start a proper bluetooth service development. The future tasks are: optimizations for several processes like service discovery, bandwidth usage, multiple service synchronization, in order to simplify the code to reduce library size and timeout strategies.

VIII. Acknowledgement

This work has been partially funded by CONICYT through Fondecyt Project No. 11080284 and the Pontificia Universidad Católica de Valparaíso (www.pucv.cl) through Nucleus Project No. 037.215/2008 "Collaborative Systems".

References

[1] M. Weiser, "The Computer of the 21th Century", Scientific American, sept. 1991, pp. 94-100.

[2] Sun Microsystems, "JSR 139: Connected Limited Device Configuration Specification", Version 1.1, 2003, from http://jcp.org/en/jsr/detail?id=139 [accessed, May 3, 2010].

[3] Sun Microsystems, "JSR 271: Mobile Information Device Profile (MIDP)", 2005, from http://jcp.org/en/jsr/detail?id=271 [accessed, May 3, 2010].

[4] Bluetooth SIG, "Bluetooth Specification", Version 2.1 +EDR, The Bluetooth Special Interest Group, 2007, from http://www.bluetooth.com/SiteCollectionDocuments/Core_V21__EDR.zip [accessed, May 3, 2010]

[5] Bluetooth SIG, "Bluetooth Specification 1.1: Part K:10 Generic Object Exchange Profile (GOEP)", 2001, pp. 310-338 from http://www.bluetooth.com/SiteCollectionDocuments/GOEP_SPEC_V12.pdf [accessed, May 3, 2010]

[6] SO Sullivan, "JSR82: Past, Present and Future for Java/Bluetooth APIs", Rococo Software, 2007, from http://www.rococosoft.com/weblog/archives/java_bluetooth/jsr82_tips_techniques/ [accessed, May 3, 2010]

[7] W3C, "XHTML Basic 1.1", W3C Recommendation, 2008, from http://www.w3.org/TR/xhtml-basic/ [accessed, May 3, 2010].

[8] C. Bala Kumar, PJ Kline and TJ Thompson, "Bluetooth Application Programming with the Java APIs", Elsevier, 2004.

 ISBN: 978-1-61208-100-7