Distributed Java Mobile Information System

Ahmad Shukri Mohd Noor
Faculty of Science and Technology
University Malaysia Terengganu,
21030 Kuala Terengganu, Malaysia.
yazid@umt.edu.my
ashukri@umt.edu.my

Abstract

Technology is now everywhere; integrated into our everyday objects and activities. Mobile computers and devices are now widely affordable and powerful enough. Because of that, developments of mobile applications become a phenomenon today. Mobile applications could be used both to connect people with each other, and to link them with crucial data from almost anywhere in mobile manner. Since mobile devices come in a variety of purposes and properties and in different communication mechanisms, a portable software platform and a flexible and extensible networking supports need to be considered. Java, specifically Java Micro Edition (JME), is particularly an attractive development environment that suits the purpose. This paper discusses a research project of a Java-based smart mobile application. The application, a client/server based application, is designed using JME. It allows the built application to be deployed on any client devices and access to a large variety of network services on the server side. The open source database.

1. Introduction

Mobile and wireless computing have changed and benefited the way people live and work. They connect them with each other and they can access data from almost anywhere in pervasive manner. Mobile devices like mobile phones are capable of much more than simple voice communication. They can act as digital data transmission devices such as modem (interface between computers and data networks) and mobile Internet browsers. They have become an object of extensive interdisciplinary research into possible new applications.

Mobile applications can be subdivided into two main types [1]: voice and messaging services, and general applications. The services dominate on all mobile phones today. There are a number of mobile telephony systems in used nowadays like GSM (Global System for Mobile Communications), WAP (Wireless Access Protocol), GPRS (General Packet Radio Service), EDGE (Enhanced Data GSM Environment), and now 3G broadband technologies such as HSDPA (High-Speed Downlink Packet Access). The most widely adopted communication is Short Messaging Service (SMS), which is not only for one-to-one messaging purpose but also prominence in entertainment industry like voting systems and contests –based reality shows. The applications seem to be more to personal information management software, mobile front ends to enterprise applications, stimulating games, and other mobile software than the former type. These applications either built on browser based which typically accessed by using the WAP, or downloaded into phones which previously created either by Binary Runtime Environment for Wireless (BREW), Java Platform (Java Micro Edition), or .NET Compact Framework (CF) developers.

Browser-based applications, native applications, and hybrid applications are the mobile application technologies that have been developed to support effective services and applications that work within the resource constraints mobile devices and wireless networks. Browser-based applications are developed using a markup language interpreted by a browser residing in the device. Native applications are executed in a device's own runtime environment mainly by using portable-standardize platform like BREW and JME, and highly support interactive wireless applications. Hybrid applications target to integrate the best parts of the rest two models. The user enters URLs in the browser to download native applications from remote servers, and the runtime environment executes them on the device.

One attractive development environment for smart mobile applications is Java. It allows a considerably

richer higher level set of development libraries and components that it would be possible using traditional embedded system languages. Additionally, its code not only can run on dozens of small platforms, but also has important safety features for downloadable code [2].

Following the success of the core Java technology – Java Standard Edition (JSE) and Java Enterprise Edition (JEE), Sun Microsystems recognized the need of expanding it into new areas – programming enterprise servers on one side, and small, resource-constrained devices on the other. It was designed for consumer devices (range in size from pagers, mobile phones, PDA and things like set-top boxes) and has been named Java Micro Edition (JME) [3].

2. Java Micro Edition (JME)

In June 1999, Sun has released this third standardization with a set of configurations and profiles at the JavaOne Conference. This version aimed at embedded and resource constrained devices especially wireless devices. JME (formerly known as Java 2 Micro Edition) is actually based on the previous versions of Java, JSE (formerly known as Java 2 Standard Edition) and JEE (formerly known as Java 2 Enterprise Edition) with some additional classes to support its purposes. This relationship is illustrated in Figure 1. Since Java offers a lot of benefits ranging from security, cross-platform compatibility, object oriented programming language and large supports from the developer community, it is ideally suited become the standard application development language for wireless devices. To guarantee the interoperability between the mobile devices needs and capabilities, standardization is necessary. As illustrated in Figure 2, JME is divided into configurations, profiles and optional APIs to customize the Java runtime environment.

Configurations are device oriented which provide a minimal set of features that all devices in the configurations must support. They consist of Java virtual machine, core libraries, standard classes and APIs. Two configurations are currently defined: 1. Connected Limited Device Configuration (CLDC), 2. Connected Device Configuration (CDC), both are particularly designed for a specific kind of device based on memory constraints and processing power. CLDC sits on top of the Kilobyte Virtual Machine (KVM), a small version of classic Java virtual machine to specifically support consumer and embedded devices. CLDC aims at low-end consumer devices which are smaller devices than CDC like pagers, smart phones and PDAs. These devices have constraints memory of less than 512KB, small screen size, low bandwidth and intermittent network connection (typically wireless).

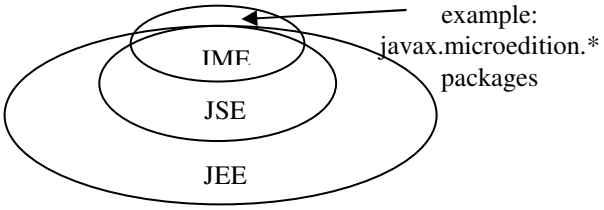

Figure 1. Java platform architecture

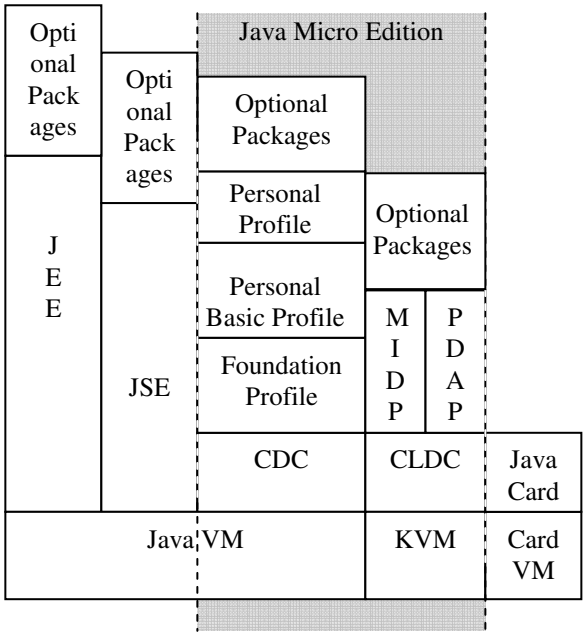

Figure 2. Java Micro Edition platform

The current version is CLDC 1.1 (JSR 139) [4] with some added features compared to previous version like floating point numbers, additional error handling capabilities, and a minimal security manager. CDC on the other hand, targeted devices with 2MB or more of total memory. These devices have minimum memory of 512KB, high bandwidth and persistent network connectivity. Examples are set-top boxes, high-end communicator, navigation systems and Internet TV. As illustrated in Figure 3, CLDC is actually a subset of CDC with its own added features.

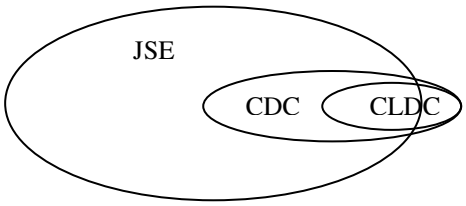

Figure 3. Relationships between JSE, CDC and CLDC

The profile layers are on top of configurations. They are more on application oriented and provide APIs or libraries to develop applications for a particular family of devices which are specific libraries than those in configuration. The goal is to guarantee interoperability within a particular device family by defining a standard Java platform for those devices. As such, application is portable to any devices that support specific profile. Two main profiles built for CLDC are Mobile Information Device Profile (MIDP) and Personal Digital Assistant Profile (PDAP) that is built for palmtop devices, and for CDC there are Foundation Profile (the base profile which does not support user interface but more to networking purposes), and four other profiles extend from it are Personal Basis Profile, Personal Profile (with full graphical user interface support to provide a platform for Web applets), Remote Method Invocation (RMI) Profile, and Game Profile. The only profile exist at the current time is MIDP and its current version is MIDP 2.0 (JSR 118) [5].

MIDP applications are called MIDlets. It is a Java application that uses the MIDP profile and the CLDC configuration. All MIDP applications run under the control of KVM. KVM is controlled by device's Application Management Software (AMS), usually provided by the device manufacturer. AMS controls the entire application lifecycle from installation, upgrade and version management, to removal of an application. A MIDlet is installed by deploying its class files to a device. The class files will be packaged in a Java Archive (JAR), either as a single application or as a suite of MIDlets. An optional descriptor file named Java Application Descriptor (JAD) describes the contents of the JAR. A MIDlet can be deployed to a device either by Over The Air (OTA) downloading, USB, Infra Red or Bluetooth, Serial cable, Email and MMS, or MIDlet pre-deployment (placing the MIDlet suite installation package to a predefined folder in local media (user area) or removable media (memory card)).

The last part of JME is optional APIs as in Figure 2, layered on top of profiles. The goal is to allow flexibility of definition of APIs on top of different profiles. They are general purpose libraries, independent and not bound to a particular device family. Some examples are Location API and Wireless Messaging API. Figure 4 pictures a handful of configurations, profiles and a range of devices supported by JME.

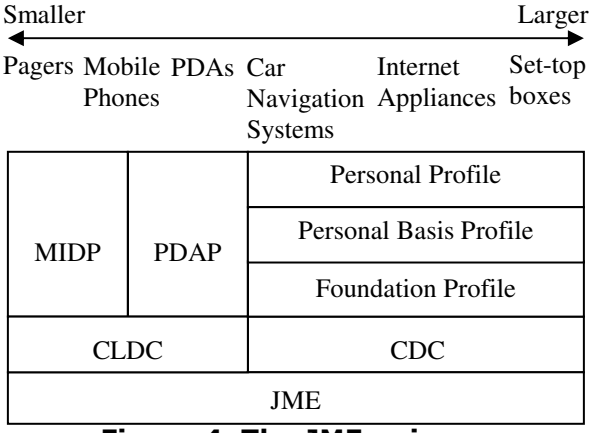

Figure 4. The JME universe

Variation of mobile devices with different communication mechanisms requires a flexible and extensible JME networking supports. An abstraction of JSE network and I/O classes is designed for JME to make connection between mobile devices and web servers possible. It is called Generic Connection Framework (GCF) and used at the programming level. Instead of using different abstractions for different protocols, only one static method of the Connector class is used. The method is Connector.open(String connect). The connection parameter passed to the static method could be file I/O, serial port communication, datagram connection, or an http connection. If successful, this method returns an object that implements one of the generic connection interfaces.

One of the most prominent communication mechanisms is HTTP. It is built around requests and responses. Client sends a request for a server to view a webpage, the server sends back a response containing the data of the requested webpage. The HTTP operations are POST and GET. The significant difference is that instead of being pasted on the end of the URL using GET, the parameters are passed as the body of the request using POST. Since MIDP specification requires that implementations at least support the HTTP 1.1 protocol, there is no portability issue and it is guaranteed available on all MIDP devices. By using HTTP, it gives mobile devices the ability to access a large variety of network services reside on the server, such as Java Servlets, Java Server Pages, Perl Scripts and CGI. But, HTTP is not a secure

protocol and vulnerable to hackers. A more secure protocol, Secure HTTP (HTTPS), was then created. It runs over an encrypted Secure Sockets Layer (SSL) or Transport Layer Security (TLS) connection. It is syntactically identical to the HTTP protocol but it provides a layer of authentication and encryption between HTTP and sockets. This connection mechanism is typically illustrated in Figure 5.

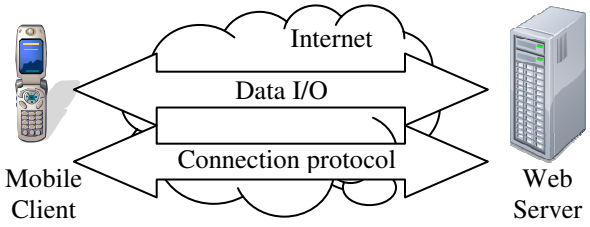

Figure 5. Connection mechanism between mobile device and web server

3. System Design

Many mobile applications particularly for complaining purposes are using SMS based technology. Basically, users will be given with a specific format comprises of a number to send followed with one or a number of different data. Commonly, users will be asked to follow a specific format before they can send a complaint. Usually, the format is a common-three-coded character representing a particular purpose such as REF for refusal, followed with other data such as location, date and time. Most cases are failed to be reported since the users could not recall the format and even the SMS number itself. A balanced tradeoff between overall performance of the system and user friendliness could lead to a successful system. Data should not be limited to only text data, other type of data like images could be used in order to provide complaints with clearer picture of the actual problem.

New system architecture and protocols must be designed to confront with the above scenarios. Some functions need to improved at both client and server sides to achieve the desired system performance.

This system design is based on three-tier client server architecture. The user interface runs on the client and the database is stored on the server. Both client and server are Java based implementations. There are two types of user. The first user, called public, is anyone who sends the complaint, and the other, called staff, is someone who takes responsibility of the complaint. The public initiates the complaint and sends (query) it to the database using his or her handset. Upon receiving the query, the server stores the data according to a staff's responsible service area. The staffs can search (query) and retrieve any relevant data using the handset and take an immediate relevant action. A complaint can easily be translated into an action since it comes together with an image. It pictures a clearer situation of what has been complained. The staffs can access the information via HTTP connection from the handset once they select the complaint identification number (unique number assigned to each complaint upon submitted) which appears on the device.

3.1 System Architecture

This architecture will describe in details on the system design strategies, including the data model. The system architecture under consideration is illustrated in Figure 6. The server comprises of server's implementations that are responsible for two main tasks. The first task is to accept information from the public and store it into the database residing in the server. The second task is to search any relevant information from the database and sends them back to the staff.

On the server side, Java servlets which are written in Java programming language are used as the network service to support the communication between client mobile requests and servers responses. This choice is also due to the portability feature offered by Java. Apache Tomcat is used as a development web server to serve the servlets. The connection between the server's implementations and the MySQL database is accomplished via its native Java driver that supports heterogeneous environment. Since MySQL supports BLOBs, all image data from mobile client is stored in their native binary format, and all other data is stored as their original type.

The following are utilized for the development of the server application:

- MySQL Server 5.0.
- Connector/J MySQL JDBC driver as the connector between Java servlet and MySQL database.
- Apache Tomcat 4.1.34 as a development web server to serve the servlets.
- AduanServlet.java class for submitting information from mobile client to database.
- TugasanServlet.java class for searching and sending information from database to mobile client.

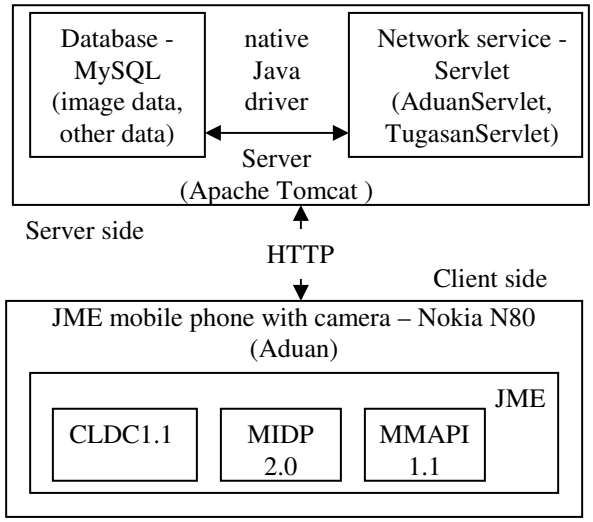

Figure 6. The system architecture

The following features are provided by the mobile client's implementation via a graphical user interface:

- Connecting to the server's implementations.
- Allowing public users to snap picture as one of the data.
- Invoking the server's implementations with the appropriate commands.
- Displaying and presenting the query results to the user.

The client's implementations comprised of JME applications called MIDlets. All the user interfaces were created by using MIDP 2.0 classes while Mobile Media API (MMAPI) was used to provide capability to snap picture using camera of the device.

4. Application Model

A mobile complaint application (m-Aduan) to demonstrate the practicability of the system architecture is developed. It allows users to lodge a report or complaint of a breakdown facility of a building, whenever and wherever they bumped into it. This is done wirelessly using their handset. Additionally, they can take a picture of the broken down facility. Then, any staffs who is responsible for service and maintenance of that building can view a list of relevant complaints. After a necessary action has been taken, they can update the data in the database residing in the server using their handset.

Figure 7 shows some typical screenshots of the client interface of m-Aduan application. It starts with Login Interface, to Complaint Interface to complaint something, or Task Interface to view a list of new task(s) to be taken care of. A snapshot of breakdown facility can be attached together with the complaint so that the staff can get a better picture of the problem.

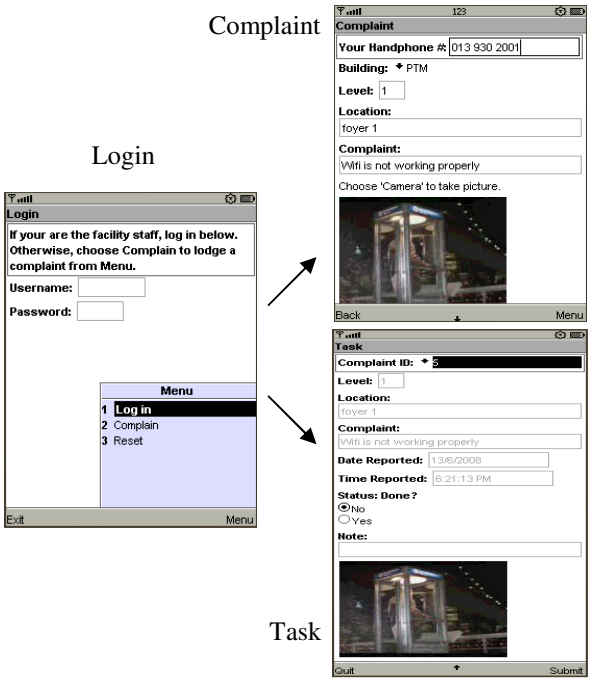

Figure 7. Screenshots of client interfaces of m-Aduan application

5. Related Works

Connectivity and portability are important aspects of mobile application. There are many applications use SMS based connectivity such as Delhi Traffic Police [15] and Malaysian Commercial Vehicle Licensing Board [16], and a little uses other based technology, particularly for complaint purposes. But, with the advancements of mobile telephony systems like 3G, many browser based system and other native and hybrid applications built using JME, BREW and .NET CF become more dominant.

The Delhi Traffic Police (DTP) uses SMS based technology in order to enlarge the reach of the commuters to lodge complaints against auto rickshaw drivers. All such complaints received via that SMS will be downloaded through Internet and action will be taken against such defaulting auto-rickshaw drivers.

Malaysian government is moving ahead in providing SMS services as an added measure of convenience for public people to interact with government agencies. Some SMS-based applications are developed for complaint purposes such as Commercial Vehicle Licensing Board complaint system for those relying on

buses, feeder buses, taxis or the LRT, but unhappy with the public transportation system. The SMS number and data format for submission vary from one with another.

Unfortunately, by the time of this writing, there is no JME, BREW or even .NET CF based complaint application found but SMS based technology.

6. Conclusion

JME mobile application platform is ideal for mobile facility complaint system as it supports application connectivity, interoperability and portability. With its rich features and libraries as well as the development tools, it is an excellent environment to develop and test new mobile application with some communication types. The success of the application architecture lies both within the portability and interoperability aspects, and how fast the available connection to support the processing needs, and how well it can adapt to meet the changing requirements of new mobile device technologies and facility complaint data such as video recording.

7. Acknowledgements

Supports from Mr. Jonathan Knudsen from Sun Microsystems and the management of UiTM Kuala Terengganu are acknowledged.

8. References

[1] Sun Microsystems, Mobile Data Services and Platforms. August 2006. http://developers.sun.com/mobility/midp/articles/midpwap2/.

[2] J. Knudsen, *Wireless Java: Developing with J2ME*, Apress, USA, 2003.

[3] J. W. Muchow, *Core J2ME Technology and MIDP*, Prentice Hall, USA, 2002.

[4] Sun Microsystems, JSR-139 Connected Limited Device Configuration 1.1. http://jcp.org/jsr/detail/139.jsp.

[5] Sun Microsystems, JSR 118 Mobile Information Device Profile 2.0. http://jcp.org/jsr/detail/118.jsp.

[6] R. Riggs, et. al, *Programming Wireless Devices with the Java™ 2 Platform, Micro Edition, Second Edition*, Addison Wesley, USA, 2003.

[7] V. Piroumian, *Wireless J2ME™ Platform Programming*, Prentice Hall PTR, USA, 2002.

[8] M. J. Yuan, *Enterprise J2ME: Developing Mobile Java Applications*, Prentice Hall PTR, USA, 2004.

[9] Sun Microsystems, J2ME Low Level Network Programming with MIDP 2.0. http://developers.sun.com/mobility/midp/articles/midp2network/index.html.

[10] Network Programming with J2ME Wireless Devices. http://www.wirelessdevnet.com/channels/java/features/j2me_http.phtml.

[11] M. Mallick, *Mobile and Wireless Design Essentials*, John Wiley & Sons, USA, 2003.

[12] Wireless technology. http://searchmobilecomputing.techtarget.com/sDefinition/0,,sid40_gci213380,00.html.

[13] IBM, Networking with J2ME. http://www.ibm.com/developerworks/java/library/wi-jio/.

[14] Technology Inside, WAP, GPRS, HSDPA on the move!. 4.9.2007. http://technologyinside.com/2007/09/04/wap-gprs-hsdpa-on-the-move/.

[15] Delhi Traffic Police, SMS based complaint system. 18.3.2005. http://www.delhitrafficpolice.nic.in/press-releases/sms-based-complaint-system.htm.

[16] TheStar Online, Your right to expect good service. 13.3.2006. http://www.thestar.com.my/news/story.asp?file=/2006/3/13/nation/13647669&sec=nation.