XML and J2ME Based Mobile Commerce

Stephen J.H. Yang, Irene Chen*, and Rick Chen
National Kaohsiung First University of Science and Technology
Department of Computer and Communication Engineering
*Institute of Management
jhyang@ccms.nkfust.edu.tw

Abstract

In this paper, we will introduce our work on how we develop a mobile environment under Java technology. In recent years, the Internet is becoming wireless, and there are more and more intelligent devices connecting to Internet through wireless technology. Unlike PC industry, these devices have its exclusive operating systems and hardware. It is hard for application developers to develop programs on these devices and integrate the existence services for these devices to use. Solution for this situation is Java technology. J2ME is the new generation Java technology that especially focuses on the mobile devices. It provides a similar environment as standard Java environment for developer to develop applications on these devices.

The most common mobile devices are cellular phone and PDA. In this paper we will focus on how to run Java programs on these two devices. We will introduce what is J2ME technology and compare J2ME with other wireless technology such as WAP technology. Then, we will present how to program Java applications on mobile devices. We use Spotlet for PDA while MIDlet for Java phone. We will demonstrate a web site with mobile devices support and an XMIDlet system for PDA and cellular phone, which support dynamic download and execute XML-based applications.

Keywords：J2ME, MIDlet, XML, Java Phone, PDA

1. Introduction

Recent years, computer chips have become smaller and smaller. People have put these chips into devices around our life. These devices include mobile devices, such as cellular phones, PDA, two-way pagers, and consumer electronics, such as set top box and Web phone. The main concept is to extend the computing environment to our business and personal life. Our life will become more digitalized and mobilized as shown in Figure 1. In addition to adding chips in these devices, there is an important issue that is how to connect all these devices through network connectivity. Various resources can be shared among these devices. That is providing one kind of service and serves all kinds of devices. This is the only way to achieve real time information and data synchronization.

Not like PC industry, mobile devices usually have their private operating systems and hardware design, which results in difficulties when we want to integrate all these devices. If there is no unified solution for developing applications on these devices, application developers will need to use different programming language for different devices, respectively. From economic viewpoint, if one application can only execute on one cellular phone or one PDA, it cannot get best economic benefits. How to make applications execute all over these platforms will be a critical issue. Cross platforms is an important feature that can attract developers.

Figure 1 Relationship between human and intelligent devices

In recently years, development environment for small handheld devices are mainly designed for cellular phones and PDA. In PDA industry, the present two main operating systems are WinCE promoted by Microsoft and PalmOS promoted by Palm Computing. But in the cellular phone industry there are a lot of manufactories contend with each other, the most popular are Motorola, Nokia, and Phone.com. In virtually the small device fittings market has not figure out a strong guiding standard that consequence the difficulties of concerning software developments. Based on our past experiences of developing WAP applications [1], we recognized that utilize WAP SDK by Nokia in the concerning software developments had hardly not modified when navigating in others WAP navigator. As a result, Sun Microsystems developed a small Java to apply to the small devices in the second edition, which is Java 2 Micro Edition (J2ME). The J2ME can apply to the development of mobile devices and information appliances application. WAP was not anticipated when it carries out in the market has brought about many discussions. We concerned that the crucial problems is WAP user can only navigate the simplified information in a monochrome picture with a high cost but only a lame navigator. Besides the high cost of Internet connection and unfamiliar operate surface the WAP user still have to tolerate the slow Internet speed. Another point of view is the market eliminate new products very fast that no mobile phone has time to upgrade, therefore if no adequate values there is no way to attract the consumers.

From business opinion, if we just use WAP technology, we cannot solve all the problems in the mobile commerce. First, WAP only provides browsing capability and all information is provides from server. It also needs continuous network connection to fetch the data. It is a serious drawback because the infrastructure of the mobile environment is not good enough. The connection cost will be a serious problem. This problem still exists even from GSM network to GPRS. J2ME seems provides a solution, it supports development for cellular phones and PDA. It allows application developers to create many different kinds of application on the devices. In the specification of J2ME technology, Java phone can dynamic download application in the near feature. It means that the phone is expandable and not just consumable. Enterprise can develop their own business application on Java Phone for their need. In this paper, we have utilized XML and J2ME technology to develop a wireless mobile environment and through develop application on cellular phones and PDA to build a complete mobile commerce platform.

The rest of this paper is organized as follows. Section 2 will address what is J2ME technology. Section 3 will show how to program on those devices that support J2ME technology and their program life cycle. Section 4 addresses the designs and implementation of a virtual mobile environment, which allows different users with different devices to access the computer servers. Each role can get their service through login identification. We also designed a XML based system, which support dynamic downloading XML based application

to J2ME devices. Finally, we conclude this paper with our future research in Section 5.

2. J2ME Programming

Sun Microsystems has recognized that one size of program does not fit all, it has grouped its Java technology into three editions. Each aimed at a specific area of today's computing industry. The Java 2 Enterprise Edition is especially designed for server side development; Java 2 Standard Edition is designed for PC-based computer development; and Java 2 Micro Edition is designed for consumer electronic devices and embedded systems. All editions have their own Java virtual machine and optimized class libraries according to their target industry. In the following, we will brief introduce all these editions.

Java 2 Standard Edition (J2SE): J2EE and J2ME are based on this edition. This edition implements all features that mentioned in the Java 2 specification. J2SE is for the familiar and well-established desktop computer markets. It supports all kernel class libraries and type libraries. It can develop client side Java applications and Java Applets.

Java 2 Enterprise Edition (J2EE): J2EE is a superset of J2SE that is target to enterprise programming. J2EE addresses the needs of corporations that need to write server-based programs that support thousands of clients. J2EE focuses on three elements: Robust application deployment, Interoperability with legacy systems, and enhanced component-based development. In our implementation, we also choose J2EE technology for our server-side solution.

Java 2 Micro Edition: It focuses on the consumer and embedded devices. This edition does not support all features of Java language. J2ME is a layered structure platform [2,3]. While connected consumer devices such as cellular phones, pagers, PDA, and set-top boxes have many things in common. They are also diverse in forms, functions and features. To address this diversity, an essential requirement for J2ME is not only small size but also modularity and customizability.

2.1 Configurations and Profiles in J2ME

The J2ME architecture is modular and scalable so that it can support the kinds of consumer and embedded devices on demand. To enable this, J2ME provides a range of virtual machine technologies, each optimized for the different processor types and memory size. For low-end, resource-limited products, J2ME technology supports minimal configurations of the JVM and Java APIs that just fit essential capabilities of each kind of device. When device manufacturers develop new features in their devices, or service providers develop new applications, the minimal configurations can be expanded with additional APIs or with a richer complement of JVM features. To achieve this goal, two essential concepts are defined by J2ME. One is configuration and the other is profile.

Configuration defines a minimum platform for a "horizontal" category or grouping of devices, each with similar requirements on total memory budget and processing power. A configuration defines the Java language and virtual machine features and minimum class libraries that a device manufacturer or a content provider can expect to be available on all devices of the same category. In this paper, we focus on CLDC, which stands for Connected Limited Device Configuration. CLDC is for devices with limited connection capability, such as PDA, mobile phone, two-way pagers.

A profile addresses the specific demands of a certain "vertical" market segment or device family. The main goal of a profile is to guarantee interoperability within a certain vertical device family or domain by defining a standard Java platform for that market. Profiles typically include class libraries that are far more domain-specific than the class

libraries provided in a configuration. In this paper, we focus on MIDP, which stands for Mobile Information Device Profile [4,5].

In J2ME, an application is written for a particular profile, and a profile is based upon or extends a particular configuration. Thus, all of the features of a configuration are automatically included in the profile and may be used by applications written for that profile. In its simplest terms, a configuration is a contract between a profile implementer and a device's Java virtual machine. The virtual machines of all the devices in the market segment agree to implement all the features defined in the configuration. And the profile implementers agree to use only those features defined in the configuration. Thus, portability is achieved between the profile and the devices served by that configuration. New devices can take advantage of existing profiles. And new profiles can be installed on existing devices. At the implementation level, a profile is defined simply as a collection of Java APIs and class libraries that reside on top of a specified configuration and that provide the additional domain-specific capabilities for devices in a specific market segment.

Although J2ME defined a lot of profiles, until now MIDP is the only profile, which has reference implement. And there are a lot of Third Party Company develop the MIDP SDK. For PDA, the PDA profile is still request comments. If we want to develop the Java program on PDA, the only solution is to use the CLDC reference implements.

J2ME's Virtual Machine (KVM) technology is a compact, portable Java virtual machine specifically designed from the ground up for small, resource-constrained devices [6]. The high-level design goal for the KVM technology was to create the smallest possible, complete Java virtual machine that would maintain all the central aspects of the Java programming language, but would run in a resource-constrained device with only a few hundred kilobytes total memory.

The "K" in KVM stands for "kilo." It was so named because its memory is measured in kilobytes (whereas desktop systems are measured in megabytes). KVM is suitable for 16/32-bit RISC/CISC microprocessors with a total memory of no more than a few hundred kilobytes (potentially less than 128 kilobytes). This typically applies to digital cellular phones, pagers, personal organizers, and small retail payment terminals.

The minimum total memory required by a KVM implementation is about 128 KB, including the virtual machine, the minimum Java class libraries specified by the configuration, and some heap space for running Java applications. A more typical implementation requires a total memory of 256 KB, of which half is used as heap space for applications, 40 to 80 KB is needed for the virtual machine itself, and the rest is reserved for configuration and profile class libraries. The ratio between volatile memory (e.g., DRAM) and non-volatile memory (e.g., ROM or Flash) in the total memory varies considerably depending on the implementation, the device, the configuration, and the profile. A simple KVM implementation without system class prelinking support needs more volatile memory than a KVM implementation with system classes (or even applications) preloaded into the device.

The actual role of a KVM technology in target devices can vary significantly. In some implementations, the KVM technology is used on top of an existing native software stack to give the device the ability to download and run dynamic, interactive, secure Java content on the device. In other implementations, the KVM technology is used at a lower level to also implement the lower-level system software and applications of the device in the Java programming language. Several alternative usage models are possible.

At the present time, the KVM and CLDC technologies are closely related. CLDC technology runs only on top of KVM technology, and CLDC technology is the only configuration supported by KVM technology. However, over time it is expected that CLDC technology will run on other J2ME virtual machine implementations and that the KVM

technology may perhaps support other configurations as they are defined.

Besides, KVM does not contain user interface related class libraries. This is another design issue. Because difference devices may have different input/output, this is not necessary to put user interface related class library into KVM package. So KVM can run at different with minor changed. If there is a need for user interface, it can separate create a suitable component for that platform.

2.1 How to program J2ME

Here we will introduce how to develop application on the cellular phones and PDA. J2ME has different develop cycle with other standard Java programs, so as it's application life cycle. Because of device memory restrict and virtual machine difference, J2ME applications are different from general Java application. The Java Compiler does not check the kernel class libraries and Java types during compiling period. In many editions, there is a check or preverifier. The tools can helps to check and preverifier applications before put to the target platform.

In J2ME, there is another function for preverifier. It helps to lower the virtual machine's workload. Traditional Java program need byte code verifier check before it can execute in the virtual machine, no matter it comes from local machine or through network transfer. It is a security issue, to prevent from the bad intention to destroy the applications. But J2ME devices do not have computer power as PC computers. It must do this job in other ways. Therefore the job divides into two passes. For example, through two-pass class verifier, MyApp.java first compiled into MyApp.class, and use preverifier to transform into another MyApp.class (it will transform into a .prc file if it uses CLDC to develop Java applications on PDA). In the second pass, transfer the MyApp.class to the target platform, then byte code verifier on the platform will check the application if qualify when it execute. It is the same procedure no matter we develop when kinds of J2ME applications.

2.2 CLDC programming: Spotlet

Using CLDC reference implement, we can develop Java applications on it. The CLDC application is called Spotlet. Just like all Applets extend java.applet.Applet, all Servlet extends javax.Servlet.http.HttpServlet. Spotlet is designed to extend com.sun.kjava.Spotlet.

2.3 MIDP programming – MIDlet

MIDP is a second reference implement in the J2ME technology. This profile is target on the cellular phones and two-way pagers. Besides a virtual machine, it also contains a good develop environment. It contains a RAD tool called J2ME Wireless Toolkit. The MIDP application is call MIDlet application. MIDlet is very similar with Applet. They have similar application life cycle, which divides into three stages: Start: to acquire resources and start. Pause: to release resources and become quiescent (wait). Because the feature of the cellular phone, user may holds the application program when they get a phone call. When the program is at pause stage, it will contain the necessary resources. Destroy: to release all resources, destroy threads and end all activity.

There is another concept of the MIDlet program; many MIDlet applications can build a MIDlet suite. The MIDlet suite is in a compressed format (jar format). In this file contains all necessary class libraries and related resources, such as icons, and images (png format). The jar file also contains a description file called Manifest.MF. The Manifest contains the following information: Name, Version, Vender, jar-URL, jar-Size

In addition to develop the jar file, the system automatically generate a description file in jad format with the application. The description file contains the information of file size, application support environment and application location URL. It can help the client devices to distinguish whether the device support the application. The bandwidth of the wireless network is not fast enough, it may be the best way to prevent the device to download unwell application into the device.

A MIDlet download flow contains four main steps: Client asked server to provide the MIDlet application. Client downloads the JAD description to it, and check if there has enough execute environment for this MIDlet. If pass the verifier, client downloads the application file into the device. Activate the program and execute it. For security reason, there is a special component in the MIDP specification. It is a Java application manager; it controls all the MIDlet action. It is designed and implement for native language. Because it needs to maintain the life cycle of all the MIDlet applications, and MIDlet suites cannot communicate with each other and cannot share resources. The trend of today is to download application Over-The-Air (OTA). But the OTA specification is still under construction, for now it still cannot achieve this goal. The manner it uses is to use a RS232 cable connects to desktop PC. Download to the cellular phone.

2.4 XML Parser for Small Devices

Because of the J2ME specification defines that all MIDP devices need to support the HTTP protocol. It is well known that HTTP is a request-response protocol. The data transfer between client and server is in Web pages format. It is difficult for MIDP devices to retrieve information in a mixed page. The best way is to define a clear and simple data presentation method. The XML is especially useful in this situation. But there is another problem that the traditional XML parser cannot fit the use of these small devices. Because these small devices usually contain only hundreds of kilobytes of memory, it is not possible to put a megabytes level XML parser on it [7,8]. The solution is to rewrite a XML parser especially for small devices to use. Because of the size constraint, the new parser may drop some features that XML specification defined, such like DTD support [9]. The size of the XML parser is not the only issue that needs to be considered. Because the parser will dynamic allocate extra memory when it load a XML document, it is very important that if the extra memory will exceed the device original has.

2.5 Export enterprise service to mobile devices

Because J2ME devices only support HTTP protocol at this time, there is no other way to export an enterprise service to those devices except using Servlet or JSP as a service proxy. J2ME devices may not direct to access enterprise such as back-end database or business logic components such as Enterprise Java Bean (EJB), but it can access these services through service proxy written by Servlet and JSP. There is another approach to export service to J2ME devices. It is called Jini Surrogate architecture [10,11,12]. The Jini Surrogate architecture is designed for those devices, which cannot execute a complete Jini environment and lets them connect to a Jini network. But the current stage of Jini Surrogate architecture is for J2ME devices with CDC configuration, such as set-top box, and screen phones, which has persistence network connection. The technology is developed under Java RMI technology. The mobile devices such like mobile phones and PDA does not fit the requirement, because they only support HTTP protocol. In the paper we used Servlet or JSP as our service proxy and through HTTP protocol to export the service to the mobile devices.

2.6 Use persistence refresh as push mechanism

Push mechanism is very important in mobile environment. But there is no other push mechanism except the short messages. But short messages still has many limits. It can only transfer text to client, and the length of the short message cannot be longer than 153 characters. Both WAP and Java environment does not provide the push mechanism. But J2ME provides thread mechanism; we can use a low priority thread as a background message receiver to get messages from server. Take a advertisement rotator for example, we can use a advertisement rotator thread especially responsible for advertisement fetch and display. Although this may resolve some problems, it is not the final solution. It is very important to have a real push mechanism in mobile environment. The current process of the mobile environment is tending to use a short message to activate a Java Phone application and fetch the message from server.

2.7 Data synchronization on mobile devices

The current open standard of the data synchronization is SyncML, which defined by world lead telecom company and software vender such as Nokia, IBM, and Microsoft. It is a important issue to make all devices have the same data at one time. Because the SyncML is too complexity to fit our use, we defined our own data synchronization mechanism and XML document tags. Our system uses this mechanism not only to synchronize data but also to communicate to the server. When J2ME applications need to synchronize date to back-end server, it will first activate a sync agent to collect the data in XML format and send to the server. The sync agent on the server will receive the data from HTTP request body and parse the data and update the back-end database and collect the data need to synchronize to the client device. The sync agent receives data from response body and parses the data and updates the RMS on the device and completes the synchronization process.

2.8 J2ME Design Guideline

Because of the environment constraint, developing J2ME applications is not as same as the traditional Java applications. First, we should know that the resource on the small devices is not enough if we program in wrong method. Because of the garbage collection mechanism does not as same as J2SE does, it is encourage that you should create class instance only when you need that instance. After you used the class, you should release the resource immediately. The release method is to assign declare variable to null. It is a good manner to first declare the variables you need in your application, but create it only when you need that object. Another feature of the J2ME programming is user interface design. Because the screen size of these mobile devices are quit small, it is encourage that you should put important information or related information in a single screen as possible as you can. And present application in as few screens as possible. You should work hard to discover the optimum user interface for you applications.

It is a better way to fetch system environment first before you write your application. Such like you should know that the device screen width and height in order to arrange the best view of the user interface component. Because each device may has different size and different presentation screen. You should pay more effort to deal with this situation or you application will looks great on one device and looks in complete mess on the other device.

Figure 2 show that different devices may have different screen size.

Figure 2 Devices with different screen size

3. Mobile Web Environments

The WWW changed the whole business environment around the world. The buyers can get the lowest price around the world in few seconds and the sellers can do their business in whole world. Today's business environment is integrated into a global market. People today can trade with each other around the world without geography limits. Figure 3 shows the basic trading environment over WWW. Customers can browse and purchase products through Web browsers at home. When enterprise receive the purchase order from customers. They will confirm the order information and pass it to warehouse for post processing. In warehouse support system, the online packager packages the products and use the proper delivery method sends to the customer's address.

Recent years, mobile Internet and mobile devices grow up very fast. One of the most successful devices is mobile phone. The first generation of smart phone has WAP technology in it. Just as mention before, WAP gives mobile phone Internet capability. Customers have another way to purchase product through WAP phone. So the business environment has changed to Figure 4. Shopping center provides WML content format to support WAP devices. Because of the specification of the WAP technology, there is a WAP gateway between the client and Web server for content transformations. Customers browse the WML version shopping center on their WAP phone.

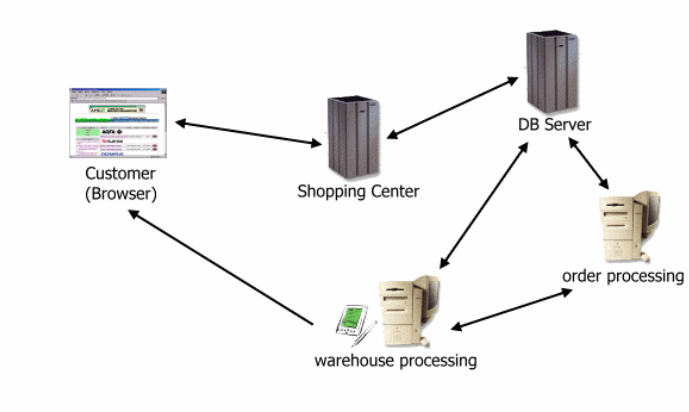

Figure 3 Trading environment over WWW

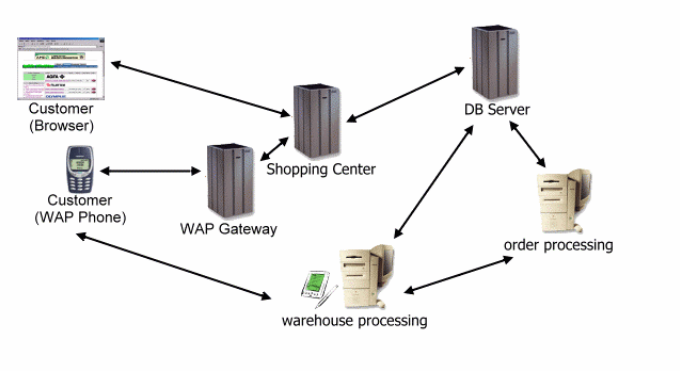

Figure 4 Trading environment with WAP

3.1. Mobile Commerce

Because of the first generation smart phone only has WAP technology on it, it cannot provide many features but only browse. We can see that the only difference between Figure 3 and Figure 4 is that Figure 4 adds WAP phone support. It is not enough for a enterprise that the only benefit from the mobile Internet is to provide mobile shopping mechanism for customers. In addition to customers, there are many kinds of people involved in a enterprise just like managers, sales, officers, and warehouse workers, etc. Because of the J2ME technology, the new generation Java Phone has local computing, persistence storage, and other features [13,14]. It means that programmers can development MIS applications on it. It can execute Java applications to access enterprise services. Today, the mobile commerce environment has change to Figure 5.

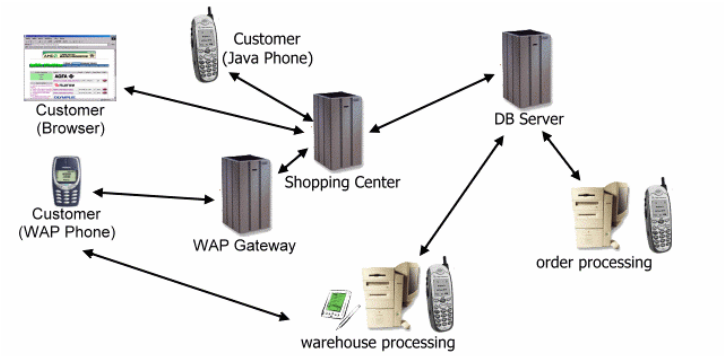

Figure 5 Trading environment with J2ME

The Java phone is not only a smart phone but also a computing environment and can be upgraded. Because of the features it has, we can develop many kinds of application for different people who involved in the enterprise. Customers can still use Java Phone to browse and purchase products and they can save prefer products' information in their phones. Each kinds of employee in enterprise can also use proper J2ME applications to replace the Desktop PC programs. For example, it is not convenient for warehouse workers to record stock quantity and package order products information through Desktop PC. Because warehouse factory is usually very big and Desktop PC is not mobile enough for those people to use. Another approach is to use Web Pad or PDA for stock checking or other jobs, which

need high mobility. But today's Java phone can replaced the PDA' job and basically Java phone is cheaper than Web Pad and PDA.

3.2. Wireless Virtual Store

Today, mobile phone is not only a communication tool but also a personal computing device that can complement PC and PDA's functionality. In this system, we would like to demonstrate the functionality of the Java Phone. We build a virtual enterprise environment and shows which kinds of people can use Java Phone to reduce their jobs and get information from it.

3.3 System Architecture

The system has three parts to serve different kinds of user. As shown in
Figure 6, our target users include customers, employee and system administrators.

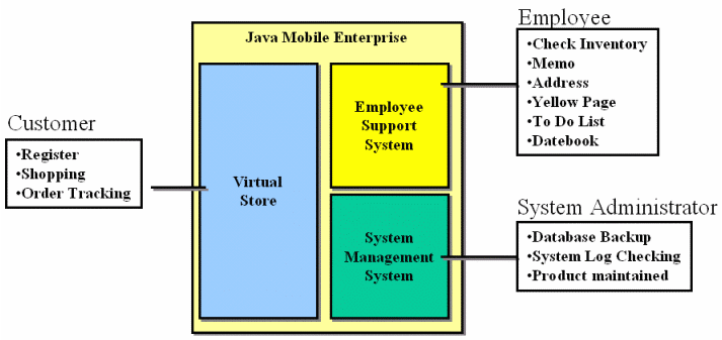

Figure 6. Mobile commerce overview

Customers can purchase or browse products on virtual store through Web browsers and Java phone. We also simulate an enterprise to provide employees several services in employee support system, such as check inventory for warehouse and sales. System administrator can use system management to maintain product information. The system developed under J2EE and J2ME technology is a 3-Tier architecture [17]. Figure 6 shows the system architecture.

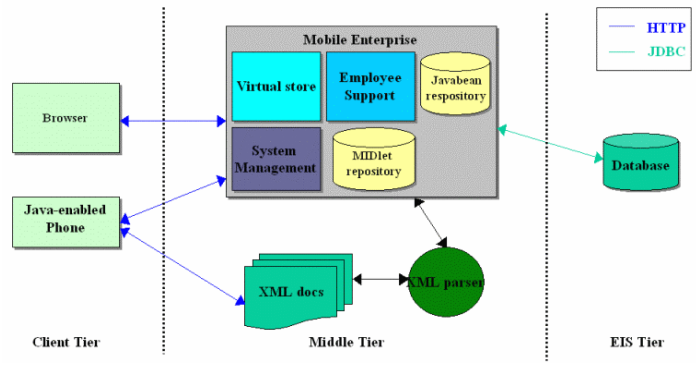

Figure 7. Mobile commerce system architecture

3.3.1 MIDlet Based Client Tier

In the client tier, we need to develop the MIDlet programs. There are three kinds of MIDlet programs in our systems: virtual store, employee support system, and system management program. The connection between client and server is HTTP protocol and the data exchange format is based on XML technology. All data need to be exchanged between client and server is saved to XML format than transfer to each other. Because that all mobile phone, which support MIDP specification, need support HTTP protocol. It means that if you use other protocol in your MIDlet applications, it will lose portability to other platform because other platform may not support the specific communication protocol. HTTP is a request-response communication protocol; it means that we cannot transfer our data to server like by socket or UDP protocol. The only way to exchange data between client and server is to request a page and post data method. All we get from server is like a Web page; it is hard to retrieve information from it. Using XML document saved this problem and gives a great flexibility to define our data exchange format. In the system, we defined our data format for synchronization.

3.3.2 XML and J2EE Based Middle Tier

In the middle tier, we choose J2EE architecture as our base technology and use JRun Application Server Enterprise Edition as our application server [15,16,17,18]. All mobile enterprise Web applications are stored on the application server. We used JSP and Servlet technology to develop Web site and used XML parser to generate XML documents and parse XML documents. We also put the MIDlet applications on the server repository for download. Until now there is no dynamic download mechanism for MIDlet installation. If we want to demonstrate the applications, we need to download it from server manually and activate it by using Java Phone emulator or saved the application to a real Java Phone through RS232 cable.

3.3.3 SQL Based EIS Tier

EIS tier is our database location. In this implement, we choose Microsoft SQL Server 7.0 as our database. The JDBC driver we used is version 2.0 and can setup from JRun Management Console. Web server application can retrieve data from database through JDBC driver. The client can get data information through request to a server data generator. The generator can retrieve data information and package it in a XML document and response to the client. Client parse the XML document and saved it to the RMS and for later use. It is a better approach than WAP does. Because it is still very expense for mobile phone communication. If we can download the data at one time and can saved it to persistence storage. It can save a lot of cost. In our system, we choose this approach many times. Next section, we will detail discuss each sub systems in the mobile enterprise.

4. Designs and Implementation

The virtual store system is an online commerce system, customers can use browser and Java Phone to shopping and browsing products. Figure 8 is the Virtual Store Use Case diagram. As we can see the virtual store provides several functionalities. User can register as a member of the virtual store. User can login into the system and browse and shopping in the system.

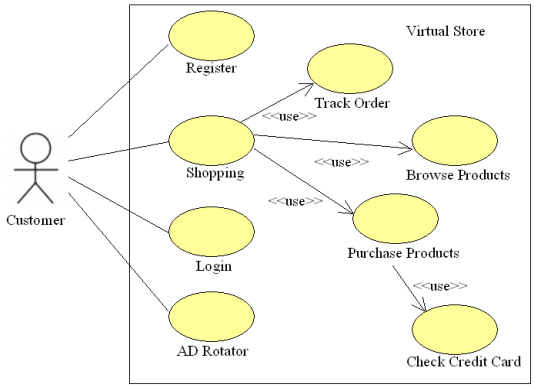

Figure 8. Virtual Store Use Case Diagram

The system also provides the user to track his order. When user purchase products, the system will check his credit card number if it is a valid credit card number. Here we simulate the credit card checking as an empty class. If the credit card number is 16 numeric characters, we assume the credit card number is valid. When accept the order from customers. The virtual store pass the order to the employee support system and there is another employee will verify the order and pass it to the warehouse. Figure 9 shows the Virtual Store system architecture. Basically, the virtual store provides user to purchase online through Web browsers. Customers should first register as a member of the system, and login into the system. The system also provides an advertisement rotator, when customers purchase the products. Customers can also get the information of the top 10 sellers and top 10 viewers of the products.

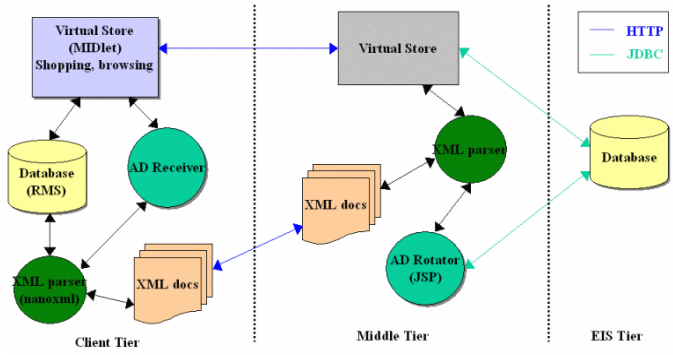

Figure 9. Virtual Store System Architecture

The Web version of the virtual store is just like the normal online store. The most important part of the system is the MIDlet version application. As we can see in Figure 9, MIDlet version virtual store has its own product database stored in the RMS system and a thread responsible for advertisement retrieve. When customers use his Java Phone to execute the virtual store application. The system will first check that if the user agent is connect to the Internet. If it is not connected to the Internet at that time, the system will execute as a standalone application on the Java Phone. If it is connected to the Internet, it will first check the Virtual Store product database. If the product database is newer than the product database on the Java Phone, it will activate the synchronization process and synchronized the product database on the Java Phone.

4.1 Client Side

From now on, customers can browse product database information and purchase product offline. If the connection is alive when customers browse product, the system will create a thread especially response for retrieve the advertisements. Because of the screen size of the Java Phone, we decide only show the advertisement text on top of the Java Phone screen. A Web based advertisement can show the whole information of the advertisement, include advertisement URL (AD-URL), advertisement image (AD.GIF), and advertisement text (AD-Text) as shown in
Figure 10. But Java Phone can only show the advertisement text on the top of the phone screen as shown in
Figure 11. The Left side of the
Figure 11 is a Nokia emulator and the right side of the
Figure 11 is a emulator of the Sun J2ME toolkits.

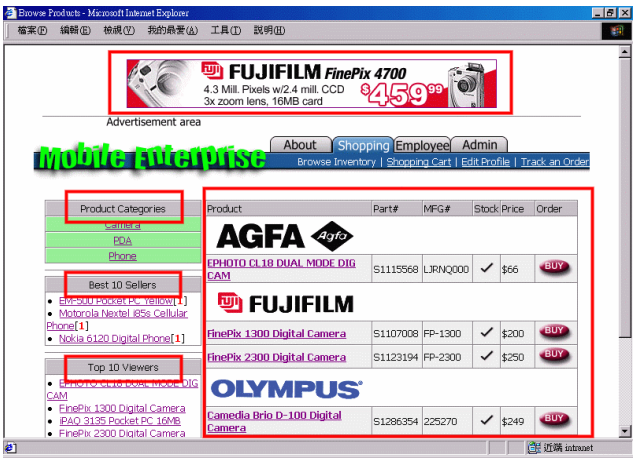

Figure 10 Browse Categories and advertisement

Figure 11 Browse Categories and advertisement MIDlet version

Another feature of the system is the implementation of the shopping carts and check out mechanism. We can save connection cost by browsing products offline. We first download all product information from server and store on the Java Phone. Customers can browse and purchase just as WAP Phone does. The only different is that all jobs are executed offline. When customers want to check out, the system will first to ask to connect to the Internet and post the product list from the shopping cart to server in order to complete the purchase. In case of out of stock, the system will check if that product can be replenished. If the product

cannot be replenished, the system responses to the client the product is not for selling right now.
Figure 13 and Figure 12 shows the shopping cart information on Web browser and Java Phone.

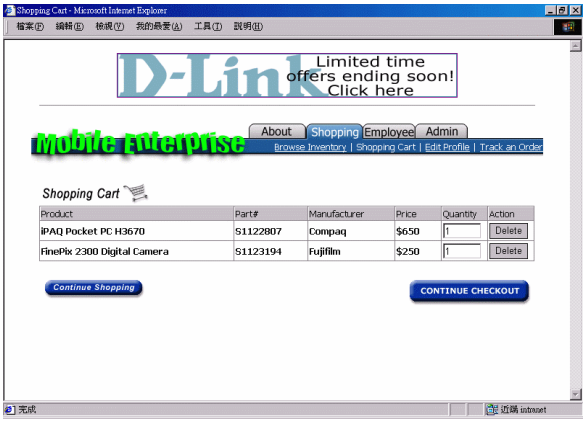

Figure 12 Shopping cart Web version

Figure 13 Shopping cart MIDlet version

4.2 Employee Support System

Once Virtual Store accepts the customer's orders, it passed the order to the internal process. The Employee Support System simulates many jobs that needs to be process in the enterprise, such as order processing, warehouse processing, replenish the stock, and check inventory. There are many kinds of employees in an enterprise. One may be a manager responsible for verify a purchase order, another may be responsible for checking customers' orders. The sales man may need the inventory information and company's yellow page information. The warehouse workers may responsible for inventory checking and product packaging. It is difficult to make all employees to use PC for their jobs. Especially for those need highly mobility. In this sub system, we analyzed each employee's job feature and determine if it needs a mobile device to help his job. Figure 14 shows the Employee Support System Use Case Diagram. We will implement a personal center and provide some PDA like programs, such as Address book; Memo, Date book and To Do List to helps employees organized his jobs.

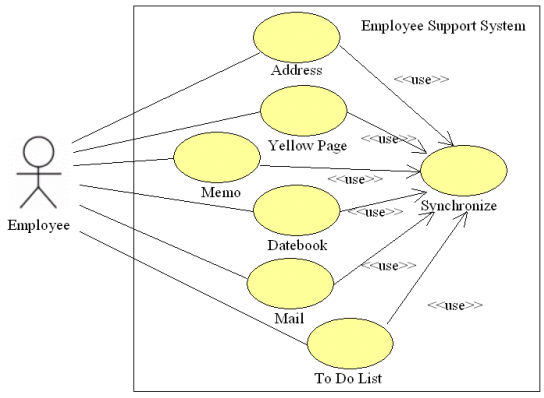

Figure 14 Employee Support System Use Case Diagram

The Employee Support system helps enterprise to manage the order process and warehouse inventory checking. It also provides several useful MIDlet application for different employee to use. Manager and sales man can use a personal center application to manage his jobs and access the company's bulletin board. Warehouse workers can use the application to checking inventory quantity and synchronizes to the back-end server.
Figure 15 shows the system architecture. It also uses the XML documents for data exchange and synchronization format.

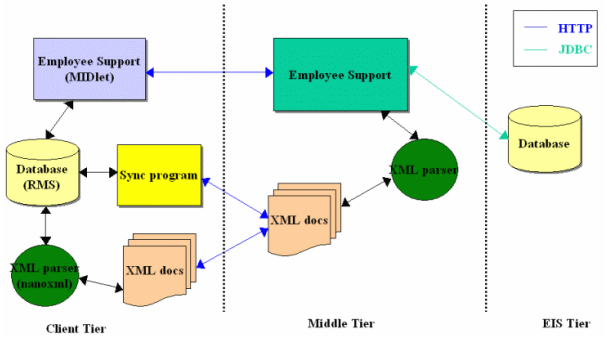

Figure 15 Employee Support System Architecture

Because there are many kinds of employees in an enterprise, we divide the employee into four types as describe below:

Manager: we provide a manager a personal center tools both in Web based (

Figure 16) and Java Phone applications (Figure 17). The personal center includes many useful tools that changed the Java Phone into a PDA like device. Manager can manage his schedule on it and records some important memo and To Do List. It can also receive the order verify message from enterprise if the manager should responsible for particular jobs such as purchase order verify.

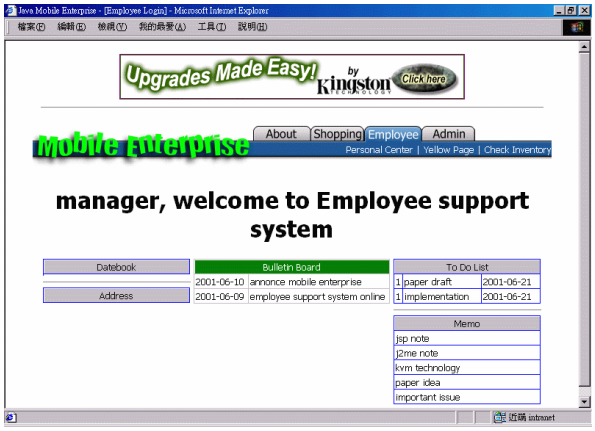

Figure 16 Employee Personal Center Web version

Figure 17 Employee Personal Center MIDlet version

Sales: a sales just like a manager who often out of offices. We also provide a set of PDA like application and add some additional feature for sales to use such as inventory querying and business achievement checking.

Customer order checker: this kind of employee is responsible for verify the customers' orders from Virtual Store. If some products are out of stock, he should new a purchase order to buy that product. If it needs the manager to verify, it should send a message to the manager and can be done by sending a short message to the manager's mobile phone. Because of this kind of employee do not need highly mobility, we don't provide any J2ME applications for them.

Warehouse worker: the workers belong to warehouse department usually need to check each products inventory and make sure that they are in safe quantity. Because they often move around the warehouse, it is not very convenient for those people to use PC to achieve this job. They don't need to learn complexity applications on PC. The only job for them is to check the inventory and type the amount into the application. It is a great example for mobile phone applications. First, leaders dispatch the job for each worker in the warehouse. Workers now can use their mobile phone to download the product information and check inventory with their mobile phone. When they get the number of each product, they can simply type the amount into the mobile phone application and the program will synchronizes to the back-end server. These kinds of employee does not need Web based application, so we just provide J2ME application for them in this system.

4.3 X-MIDlet System

As mentioned before, because the specification of OTA (Over-The-Air) is still under construction, thus we still cannot dynamically download application into the cellular phones. In this paper we would like to introduce how does our system realize the dynamic download. We will use XML technology to help to achieve the goal.

In this system, all applications are saved as an XML file. All these files are stored at the server. Another part is a MIDlet, which is an XML-based application manager. This manager manages the application downloading, executing, deleting on the cellular phone. The system is construct on the MIDP architecture. We add our own application class libraries on the MIDP profiles. There is also persistence storage on it, so we can save the applications downloading from the server. We create our own XML tags for application design. When the application manager activate an XML based application. First, it needs to parse the whole file then to create the user interface dynamically. It decreases the program difficult and gives another choice from the WAP development. The system flow is dividing into five processes

Step1: client makes a request to application download manager through X-MIDlet application manager.
Step2: download the application and save it to the XML-based application repository.
Step3: fetch application from the repository and execute.
Step4: parse the file and dynamically create application.
Step5: if the application needs to request service, it communicates with server to exchange data.

Because of the device restrict, traditional XML parser cannot fit our system. The XML parser's size becomes an issue. In this project, we choose nanoxml parser as our XML parser. The reason why we use is it has a smallest size of all the XML parser, only 6KB. But it still has some drawback. It does not support validating and the DTD is fully ignored, including <!ENTITY>. There is no support for mixed content (elements containing both sub elements and CDATA elements. But it just fits our need; we choose this as our client-side solution. We can also validate the application at the server side before download to the client.

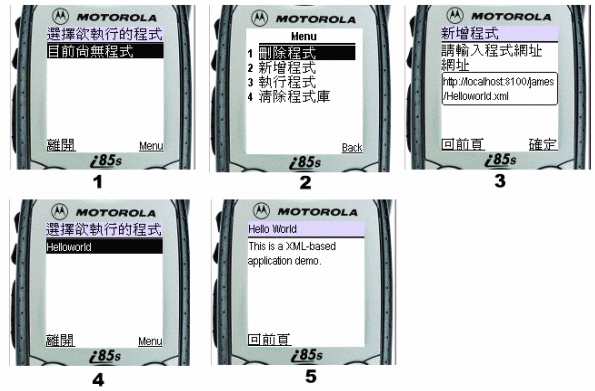

Figure 18 x-MIDlet execution examples

5. Conclusions and Future Research

Just because of the creation of J2ME technology makes wireless application development to have a total solution. Application developer can develop more powerful and useful systems under J2ME architecture. It is a complement to WAP technology. In this paper, we

demonstrate our virtual mobile enterprise. The mobile enterprise supports web browsers and J2ME devices to access its services. All uses can through a login mechanism to get appropriate services. A roll call system helps teachers to access and make a roll call through the Java phone. The X-MIDlet system demonstrates how to use XML format applications. All these systems show the power of J2ME technology. In our paper, we successful use J2ME technology to integrate new devices into existence services and achieve dynamic download applications into java phone. J2ME must be the trend of the consumer and embedded system development. Human life has tightly integrated with many intelligent devices. No matter its use WAP, J2ME or other wireless technology. But is it true that these devices really intelligent? Is there possible to put agent programs into all these devices. These agents can help to process daily routines and move to other devices to communicate with other agents to determine decision for us. We hope to focus on this issue and combine J2ME technology in the future.

References

[1] WAP Forum,http://www.wapforum.org
[2] Eric Giguère, "Java 2 Micro Edition PDG," *Addison-Wesley*, 2000.
[3] Bill Day, "J2ME Archive," http://www.billday.com/j2me/index.html, 2000.
[4] Sun Microsystems, "Mobile Information Device Profile White Paper," 2000.
[5] Sun Microsystems, "Over The Air User Initiated Provisioning Recommand Practice," 2000
[6] Sun Microsystems, "KVM White Paper," http://java.sun.com/cldc/wp/KVMwp.pdf, 2000.
[7] Marc De Scheemaecker, "NanoXML," http://nanoxml.sourceforge.net/index.html, 2000.
[8] William J. Pardi, "XML in Action," *Microsoft Press*, 1999.
[9] Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto, "XML and Java Developing Web Applications," *Addison-Wesley*, 1999.
[10] Jim Waldo, The Jini Technology Team, " The Jini Specifications Second Edition," *Addison-Wesley*, 2001
[11] Ken Arnold, etc, "The Jini Specifications Second Edition," *Sun MicroSystems*, 2000.
[12] Muthaiyan Chitrarasu, "Jini by Example," http://www.cswl.com/whiteppr/tutorials/jini.html, 1999.
[13] Neil Rbodes, Julie Mckeeban, "Palm Programming," *O'reilly*, 1999.
[14] Yu Feng and Jun Zhu, "Wireless Java Programming with Java 2 Micro Edition Chapter10 and Chapter12," *Sam's Publishing*, 2001
[15] Karl Avedal, etc, "Professional JSP," *Wrox*, 1999.
[16] Paul Tremblett, "Instant Java Server Pages," *Mc Graw Hill*, 2000.
[17] Danny Ayers, Hans Bergsten, "Professional Java Server Programming," *Wrox*, 1999.
[18] Jason Hunter, "Java Servlet Programming," *O'reilly*, 1999.