The Comparison of WML, cHTML, and XHTML-MP in m-Commerce

Jongwook Woo
California State University/ Computer Information Systems, Los Angeles, USA
Email: jwoo5@calstatela.edu

Minseok Jang
Kunsan National University/Computer Information Engineering, Kunsan, Korea
Email: msjang@kunsan.ac.kr

***Abstract*—In order to use mobile devices for business application, mobile applications and standard have grown fast – so called *m-Commerce* that normally uses *WAP* protocol to access remote web site wirelessly. WAP protocol has moved from *WAP 1.0* to *WAP 2.0* in order to lessen the developer's load by simplifying and forcing language independency. *WML* has been used to provide web-like user interface to mobile device on *WAP 1.0*. And, *cHTML* has been used exclusively in *NTT DoCoMo* service. However, *WAP 2.0* chooses *XHTML-MP* by considering existing *WAP* languages. The paper introduces and compares *WML*, *cHTML* and *XHTML-MP* in syntax and on experimental results. Besides, *m-Commerce* architecture on *n-tier architecture* is illustrated. The financial transaction system on online game is implemented in those languages as a mobile application to be compared.**

***Index Terms*—m-Commerce, WAP 2.0, XHTML-MP, WML, cHTML**

I. Introduction

e-Business system has been popular, which includes the popular terminology *e-Commerce*. *IBM* defines *e-Business* as the leveraging of network capabilities and technologies in order to achieve and maintain the huge advantages for customers, suppliers, partners, and employees [9]. *e-Business* activities can be classified into three categories based on end-users of transactions: Intra-business, Business-to-consumer, and Business-to-business. *Intra-business* activity is to share company information and computing resources among employees on the intranet: for example, knowledge management. *Business-to-business* activity is to improve inter-organizational partnerships and relationships: for example, supply chain integration. *Business-to-consumer*, the most common activity, is to provide services to consumers who are out of organizations: for example, customer resource management (*CRM*), *e-Commerce*, and web auctions etc [6]. And, financial transaction report system that is introduced in this paper is the example of *CRM*. Accountant needs to analyze the gain and loss of a product and then to see the report for net cash and net earned revenue that are computed based on the customers' subscription data accessible through the intranet.

n-tier architecture for *e-Business* system has been presented because businesses have to improve efficiency by integrating data and applications across the enterprise. Besides, the highest levels of performance and availability must be maintained for the critical businesses. In order to enable high performance, scalability, and availability to businesses, *n-tier* architecture partitions systems and software to more flexible blocks that have their roles [1]. Section 3 of this paper introduces *n-tier* architecture in detail.

As mobile devices and wireless telecommunications have grown these days, *m-Commerce* industry has been popular. *m-Commerce* can be defined as *e-Business* with mobile device. *m-Commerce* is the same as *e-Business* in its fundamental concept and actually its architecture is extended from *e-Business* architecture. But, it needs wireless environment to connect mobile device to the legacy system and to develop its client logic. *W3C* presented wireless access protocol *WAP* and languages for *WAP* protocol. In *WAP 1.x*, *WML* is the standard language but it has different syntax from *HTML* that has been the language for web browsing. Thus, *NTT*, Japanese Telecommunication Company, presented *cHTML* for *DoCoMo* wireless service, which is the subset of *HTML*. However, it is not *W3C* standard. Therefore, for *WAP 2.0*, *W3C* selected *XHTML-MP* as the standard, which is similar to *XHTML*.

In this paper, *WML*, *cHTML,* and *XHTML-MP* are used and compared to build client logic on *WAP 2.0*. *WML* is the extension of *XML* (eXtensible Markup Language) as an acronym of Wireless Markup Language. *XHTML-MP* (XHTML Mobile Profile) is the subset of *XHTML* (eXtensible HTML) that is the latest *HTML* standard for web. And, *WAP* gateway (or server) is used for *WAP* communication in Java between the wireless client logic and the legacy system in *J2EE* (*Java 2 Enterprise Edition*) as business/ data access logic [11].

In this paper, section 2 is Related Work. Section 3 *n-Tier* Architecture describes *n-Tier* architecture on *e-*

Business and *m-Commerce*. Section 4 describes the mobile application, financial transactional report system. Section 5 explains the architecture of the system on *WAP 2.0*. Section 6 compares syntax of the languages and presents experimental result. Finally, section 7 is Conclusions.

II. Related Work

Read *et al*. introduce the wireless protocol and languages at the moment. The Wireless Application Protocol *WAP* with *XHTML* and *WML* is illustrated. Besides, they describe the properties of *i-mode* mobile service with *cHTML* and *J2ME* (Java 2 Micro Edition) as tools for mobile application [13]. The paper just introduces the current techniques and trends in wireless applications.

Wang *et al.* describe *WAP* and *i-mode*. Then, they provide the sample *m-Commerce* applications simulated on *Nokia Toolkit 4.0* [14]. The Wang's paper has the similar architecture to our paper's because it is implemented on *WAP*. However, the application of our paper is implemented not only with *cHTML* but also with *WML* and *XHTML-MP*. It is integrated with the dynamic web contents in *J2EE* back-end system and even connected to *DB* of the application. However, Wang's application just communicates with *WAP* server on *i-mode*. Besides, our application is to access and display the financial transaction report.

Duncan *et al.* present their clinical data access system as a wireless healthcare application. It uses Palm Proxy server as a *WAP* server and *PDA* (Personal Digital Assistant) as a *WAP* browser. It uses existing back-end system built in Microsoft *ASP* language on Microsoft *IIS* web server but ours run on *J2EE* [15].

Mendonca *et al.* provide the health care mobile system that is executed on web server with *Palm PDA*. It is to provide health care information by reducing the errors [16]. Their back-end system is *CGI* that has less performance than our systems in *J2EE*. And, Raymond and Eneider's applications are wireless healthcare systems.

III. n-Tier Architecture

e-Business and *m-Commerce* are built on n-tier architecture. Thus, this section illustrates the architecture and its history.

The traditional *Client-Server* architecture has a mainframe that includes core applications and data. The mainframe is accessed from thick clients that are big applications. We can call it *2-tier* architecture that has many loads between client and server because of their tight interoperations for its presentation logic, business logic, and data access logic. This tight interoperation has generated many issues in the current high volume business systems. It is not scalable because it should replace the entire system when its capacity is exceeded. And, it is not flexible because its presentation logic, business logic, and data access logic are tightly coupled. If the developer wants to modify its business logic, he/she should modify the entire logics. Besides, the developer must adapt or modify the business logic when it is integrated with the *WWW* (World-Wide-Web) or other applications [1].

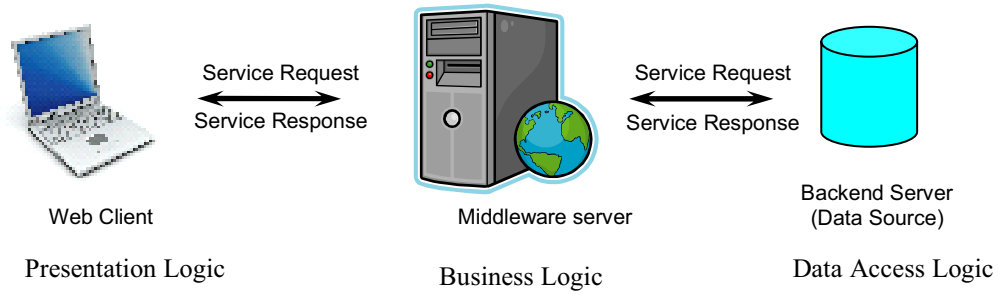


Figure 3.1. *n-tier* Architecture

The *n-tier* architecture has addressed the issues of the *2-tier* architecture and become the solution of the current *e-Business* systems on *Internet* and *WWW*. It partitions application functionalities into *n* independent layers, mainly three layers as in Figure 3.1. Thus, it becomes easier to integrate with the existing business systems. The layer 1 is the presentation logic that is typically hosted on Web server with web browser. The presentation logic is to send the request of client and receive its response from business logic. The response is normally dynamic or static web pages formatted to present the client. The layer 2 is hosted on mid-tier (middleware) server as business logic. It includes the business functions that are the main of the *e-Business* applications on *n-tier* architecture. It produces the response of the request from the client and provides it to the client. If the request is related to access data, it will pass the data access request to the back-end database server. The layer 3 is hosted on the back-end database server as database access logics. It is to handle the request of data source from the business logic. It has the functions to access data source such as plain file, XML file, database, or repository etc. Since business logic is separated from presentation logic and database access logic physically, each layer can be scalable and upgradeable independently. And, even if a layer is

modified or replaced, the application of other layers does not need to be recreated. Besides, each layer can be implemented with clustered servers for its logic. The clustering enables high-performance computing, availability, and scalability [1]. Therefore, the current *e-Business* systems are implemented on *n-tier* architecture.

m-Commerce application can be implemented on the similar architecture of Figure 3.1. In *m-Commerce*, presentation logic on *WAP* (Wireless Access Protocol) can be built in *WML*, *cHTML*, or *XHTML-MP etc*. *WAP* is the standard created by *WAP* forum in order to bring the *WWW* to wireless devices. Client device can be mobile phone or palm pilot. In order to make *WML*, *cHTML*, or *XHTML* parseable and executable, *WAP* server (or *WAP* gateway server) is needed. *WAP* server is to convert *WAP* data to *http* compatible data (or the other way). Thus, middleware server on *WAP* is composed of *WAP* server, web server, and application server.

A. WML

WML 1.x is defined in the *WAP 1.0* specification. *WAP* sites are written in *WML* as web sites are in *HTML*. *WML* is similar to *HTML* which has tags in plain text format. However, *WML* has unique tags for *WAP* document. The latest *WAP* standard is *WAP 2.0* and it defines *XHTML MP* as its markup language, which is described in the next section. Besides, *WAP 2.0* also supports *WML*. Even though *WML 1.x* is the old technology, there are still many mobile devices that only support *WML 1.x*.

When a mobile device sends a request to the *WAP* application running on the application server by selecting the system's *WAP* address as locating a *WML* file, the request is first routed through the *WAP* server where it is decoded, translated to *HTTP*, and then forwarded to the appropriate *URL*. After executing the business logic in Java classes (or other classes, for example, in *.NET*) referred by *WML* codes, its execution result will be generated. The execution result of the response is then re-routed back through the *WAP* gateway, translated to *WAP*, encoded, and forwarded to the mobile client. Thus, the mobile device can display the data result responded from the *WAP* address. This proxy architecture allows application developers to build services that are network and terminal independent [11].

B. cHTML (Compact HTML)

cHTML is the acronym of *Compact HTML*. It is simply the subsets of *HTML* 2.0, 3.2, 4.0 and 4.01 and used for *iMode* service that is supported by *NTT DoCoMo*. *cHTML* does not support *CSS* (Cascading Style Sheet) so that it does not need to be well formed and that each browser displays different views for the same *cHTML* code. It is the proprietary dependent langauge for *iMode* service. It does not support scripting language [19]. *cHTML* is the subset of *HTML* so that it is simple but it is proprietary dependent. *cHTML* is moving to *WAP 2.0* to be integrated to *XHTML-MP*.

C. XHTML MP

XHTML is the acronym of Extensible HyperText Markup Language as the latest version of *HTML* that extends *HTML* as a family of *XML*. Thus, *XHTML* has the strength of *HTML* in the look of a document and the strength of *XML* in the meaning of a document. There are several *W3C* recommended *XHTML* versions: *XHTML 1.0*, *XHTML basic*, and *XHTML 1.1*. *XHTML 1.0* is to reformulate *HTML* to *XML*-like. *XHTML* basic is to modulate *XHTML* to achieve mobile applications. *XHTML 1.1* is the larger module that can be easily combined to other *XML* documents.

WAP2.0 is the latest mobile service specification created by *WAP* forum. *NTT DoKoMo* and *WAP* forum join together to present the next wireless internet access - *WAP2.0* - by combining the features of *WML*, *XHTML Basic*, and *cHTML*. *XHTML-MP* is defined in *WAP2.0*. *XHTML-MP* is the subset of *XHTML* and it is the superset of *XHTML basic* with other *XHTML* elements and attributes. Even though *XHTML-MP* does not have the useful features such as events, variables, and script of *WML*, the great advantage of *XHTML-MP* is that web and *WAP* world now share the same document. Thus, in order to build *WAP* application, the developer can simply use the existing web documents in *XHTML* or simply modify them.

IV. Application for M-Commerce: Financial Transaction Report

This section illustrates an application that is implemented in *J2EE* and extended to wireless application by building *WML* and *XHTML-MP* clients. For business, it is important to display and analyze the gain and loss of a project. Many companies analyze their business loss or gain with the factors such as net cash and earned revenue. This section describes the basics of cash application and revenue and what are their mathematical formulae for an online game project. The online game supported by the financial system in our project has the millions of transaction with the hundreds of thousands of customers. Customer in the online game needs to subscribe it by determining its billing cycle as monthly, quarterly, semi-annually, and annually.

The financial system built on computer can provide well organized information to accountants for the millions of transactions. If there is no such a system, it will be the nightmare for the accountants. The accountants used to use the simple application such as *Microsoft Excel* for the financial system. However, the excel file cannot handle the complicated data as the online game transaction system. Therefore, our transactional system is implemented in *J2EE* to compute complicated data on web for the system. The accountants need to analyze customers' net cashes and earned revenue for a period by collecting the transactional data such as subscription amount and tax *etc*. The transactional system can display the transactional data and its net cashes and earned revenue for a period and it has two reports as

follows.

First, there is cash application report to calculate a net cash of each customer and its summary. Since there are millions of transactions in the game, an accountant needs data in certain period. Cash application report needs each game account's subscription amount, credit card fee charged, online payment fee charged, and tax charged in order to calculate its net cash. The formula to calculate the net cash is as follows:

Net Cash = (Subscription Amount) - Tax – (Credit Card Fees) – (Online Payment Fees) (Formula 4.1)

For each customer, since the online game system stores its subscription amount and tax and fees charged, its net cash can be calculated by formula 4.1. However, the user – accountant - needs to filter the millions of customers by selecting periods as start and end dates and collect only small amounts of customers.

Second, there is revenue report to calculate earned revenue for each customer by its billing cycle in a period selected by the accountant. The report needs each game account's gross earned revenue, credit card fee charged, online payment fee charged, and tax charged in order to calculate its net earned revenue. Its formula is similar to cash report formula 4.1. However, it needs to calculate daily revenue rate before calculating earned gross and revenue. The daily revenue rate is calculated as the total subscription rate divided by the number of days in the month(s) spanning the subscription as follows:

Daily Revenue Rate = (subscription amount) / (total days) (Formula 4.2)

For example, there is a customer who subscribes the game with $14 subscription amount from May 20 2007 for a monthly subscription. Thus, the customer's Net Cash is $12.25 (14 – 14 X 0.08 – 14 X 0.03 – 14 X 0.015) where tax is 8%, Credit Card fee is 3%, and Online Payment Fee is 1.5%. His account is expired on June 19 and he has 31 total days for the subscription. Thus, its daily revenue rate is $0.4516 (=14/31). Revenue will be earned on a daily basis. The rate of revenue earned will be based on the number of days in the month(s) spanning the subscription. Based on the daily revenue rate, earned revenue as of a date can be computed as follows:

Gross Revenue Earned As of Date D = (D – (subscription date)) X (Daily Revenue Rate) (Formula 4.3)

For example, for the revenue earned as of May 31 2007 of the above example, total days between May 20 and May 31 are 11 days. Thus, its gross earned revenue is $4.9676 (= 11 X 0.4516).

Let's see another example. For an annually subscription that begins on March 14th 2007 with the subscription amount $150, the user want to see the revenue earned as of May 14th 2007. First, the daily rate will be calculated as follows:

Daily Revenue Rate = 150 / 365 = $0.4110
Total days between March 15 and May 14 are 61 days (17 days for March, 30 days for April, and 14 days for May).
Its Gross Revenue Earned as of May 14, 2007 is $25.071 (=0.4110 X 61).

We can compute Net Earned Revenue with Formula 4.3 and other fees as follows:

Net Earned Revenue = (Gross Earned Revenue of (Formula 4.3)) - Tax – (Credit Card Fees) – (Online Payment Fees) (Formula 4.4)

Thus, the customer's Net Earned Revenue is $21.93 (25.071 – 25.071 X 0.08 – 25.071 X 0.03 – 25.071 X 0.015). Formula 4.1-4.4 are implemented in *Java* for *financialReport System*.

V. The Architecture of the Financial Report on WAP 2.0

The financial report system is built on Tomcat server that supports *WAP 2.0* and *Java* servlet/JSP and *MySQL* Database server. The system is mainly developed in *J2EE* version 1.5 on *JBoss* version 4.0.3 application server.

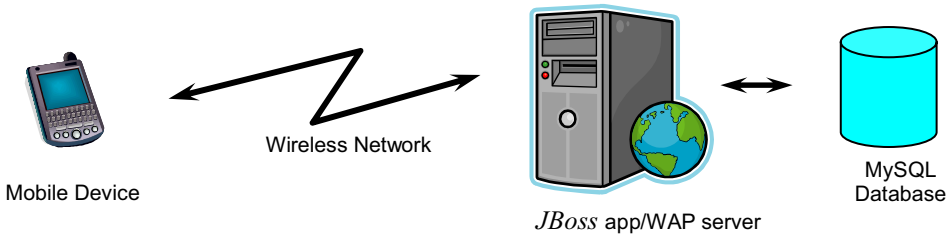


Figure 5.1 *m-Commerce* Architecture

JBoss server supports *WAP* that acts as the bridge between the mobile network containing mobile clients and the computer network containing application servers as shown in Figure 5.1. Its database server is *MySQL 4.1*. The mobile device requests *WAP* of *JBoss* server for financial report and receives its response wirelessly.

JBoss server executes business logic and data access logic in *J2EE*. The financial report logic described in section 4 is built in Java and packaged to *edu.calstatela.hipic.financialReport*. The business logic is composed of the utility classes to calculate the net cash and net earned revenue based on Formulae 4.1-4.4 by using the billing type of a customer for a period entered by the user.

The system has two main *JavaBean* classes for Data Access logic such as *CashApplicationReport* and *RevenueReport* with several supporting classes. These *JavaBean* classes have data access logics to connect DB server and to join several tables to generate the properties that are used for each report.

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML
1.3//EN"
"http://www.wapforum.org/DTD/wml13.dtd">
<wml>
 <card id="cashReport" title="cashReport">
   <do type="accept" label="Go">
     <go href="cashReportResult.jsp" method="post">
         <postfield name="sDate" value="$(sDate)"/>
         <postfield name="eDate" value="$(eDate)"/>
     </go>
   </do>  <p>
  <fieldset>
Start: <input type="text" name="sDate" maxlength="6"
format="*N"/>
End: <input type="text" name="eDate" maxlength="6"
format="*N"/>
 </fieldset></p>
</card>
</wml>
```

Figure 5.2 cashReport.wml

For example, *CashApplicationReport* is composed of *JDBC* statements to access the DB and to join USER_INFO and TRX tables. Thus, the properties of *CashApplicationReport* class are *trx_id*, *accountName*, *billStartDate*, *gross*, *creditcardFee*, *onlinePaymentFee*, *tax*, *settlementStatus*, and *netCash* with *set* and *get* methods of them. Among them, *netCash* property is calculated with other properties' values by Formulae 4.1 as shown in section 4. The properties of *RevenueReport* class are *grossEarnedRevenue*, *creditcardFee*, *onlinePaymentFee*, *tax*, and *netEarnedRevenue* with *set* and *get* methods of them. *grossEarnedRevenue* and *netEarnedRevenue* are computed by Formulae 4.2-4.4 in Section 4. The SQL commands retrieve data from Database and construct the properties of *RevenueReport* class, which maps tables and correspondent objects These SQLs are built in each Java class using *PreparedStatement* of *JDBC* library.

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML
Mobile 1.0//EN"
"http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Cash Report Input</title>
  </head>
  <body>
     <form method="post" action="cashReportResult.jsp" >
        Start: <input type="text" name="sDate" maxlength="6" />
         End:  <input type="text" name="eDate" maxlength="6" />
         <input type="submit"/>
          <input type="reset"/>
      </form>
  </body>
</html>
```

Figure 5.3 cashReport.xhtml

The client logic of the system is initially built in *WML* (Wireless Markup Language) and *cHTML* such as *cashReport.wml*, *cashReport.chtml*, *revenueReport.wml*, and *revenueReport.chtml*. Then, the clients are implemented in *XHTML-MP* for *WAP 2.0* such as *cashReport.xhtml* and *revenueReport.xhtml*.

As shown in Figure 5.2-4, *cashReport.wml*, *cashReport.chtml* and *cashReport.xhtml* files take the input parameters from the user and pass them to *jsp* file *cashReportResult.jsp*. The *jsp* file is developed to access *Java* classes of the data access and business logics. Thus, it retrieves data from *DB* tables with *SQL* command and generates a *JavaBean* class that contains data retrieved. And, it displays the result into the *LCD* panel of the mobile device that represents Cash report as in Figure 6.1 - 6.2. Thus, the mobile device can display the data result responded from the system's *WAP* address.

```
<html>
  <head>
    <title>Cash Report Input</title>
  </head>
  <body>
     <form method="post" action="cashReportResult.jsp" >
        Start: <input type="text" name="sDate" maxlength="6" />
        End:  <input type="text" name="eDate" maxlength="6" />
      <input type="submit"/>
      <input type="reset"/>
     </form>
  </body>
</html>
```

Figure 5.4 cashReport.chtml

In order to represent Revenue report, there are *revenueReport.wml* and *revenueReport.xhtml* files that pass the input parameters to *revenueReportResult.jsp* file, which are similar to Figure 5.2 - 5.5. The data displayed on the mobile device by these files are billing cycle, gross revenue, online payment fee, credit card fee, tax, and net earned revenue.

VI. COMPARING WML, CHTML, AND XHTML-MP

This section compares *WML*, *cHTML* and *XHTML-MP* for the two issues. The first is how to implement input pages of *m-Commerce* applications, which show the differences of the syntaxes in terms of programming language. The second is to display the output pages that we can compare as the result view in both approaches.

A. Syntax

WML has the features that *XHTML-MP* and *cHTML* do not have. *XHTML-MP* and *cHTML* do not support events like *ontimer*, *onenterbackward*, *onenterforward* and *onpick* of *WML*. *XHTML-MP* and *cHTML* cannot declare variables as *WML* does. *WML* provides client side scripting with *WMLScript* language. In the future, *XHTML-MP* will have a script language called *ECMAScript Mobile Profile (ECMP)* that is designed for *XHTML-MP*. In *WML*, input element has format

attributes to determine the length of characters and its type that the user can enter. *XHTML-MP* needs to format the input with *wap-input-format* property of *WAP-CSS (Cascading Style Sheet)*. In *WML*, in order to get data, *anchor* link with "go" option is needed. However, in *XHTML-MP*, "submit" button is enough without *anchor* link [17-18].

```
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//WAPFORUM//DTD XHTML
Mobile 1.0//EN"
 "http://www.wapforum.org/DTD/xhtml-mobile10.dtd">
// jsp codes to get the parameters for start date sDate and end
date eDate
// And to retrieve data from the DB by the parameters and create
ResultSet rs for cash report
…

 <html xmlns="http://www.w3.org/1999/xhtml">
  <head>
   <title>Cash Report Result</title>
  </head>
  <body>
    <% while(rs!=null && rs.next()) { %>
        <p>Date: <%= rs.getBillStartDate() %></p>
        <p>TRX: <%= rs.getTrxID() %></p>
        <p>Acct: <%= rs.getAccountName() %></p>
        <p>Gross: <%= rs.getGross() %></p>
        <p>Credit: <%= rs.getCreditCardFee() %></p>
        <p>Online Fee: <%= rs.getOnlinePaymentFee %></p>
            <p>Tax: <%= rs.getTax() %></p>
            <p>Net Cash: <%= rs.NetCash() %></p>
<% } %>
  </body>
</html>
```

Figure 5.5. cashReportResult.jsp

XHTML-MP and *cHTML* are originated from *HTML*. Thus, some tags are common and for the simple applications like in this paper, the documents are almost same as in Figures 5.3 and 5.4. The document in *XHTML-MP* contains more tags related to *XHTML* than in *cHTML*. Besides, *cHTML* does not support scripting language and *CSS*. And, *XHTML-MP* is *W3C* standard but *cHTML* is proprietary dependent.

(a) Cash Report Input (b) Cash Report Result

Figure 6.1 Cash Report on WML

B. Experimental Results

The financial report system is implemented with servers such as *JBoss 4.0.3* application/WAP server and *MySQL Database* server. The mobile device and application is simulated with *Phone Simulator v7* that is freeware developed by *Open Wave* [20]. The applications are written in *WML, cHTML,* and *XHTML-MP*.

(a) Cash Report Input (b) Cash Report Result

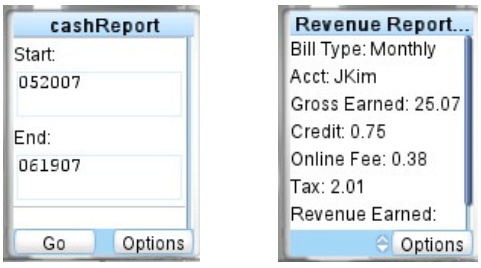
Figure 6.2 Cash Report of WML on Nokia Series 40

(a) Cash Report Input (b) Cash Report Result

Figure 6.3 Cash Report on cHTML and XHTML-MP

Figures 6.1-6.5 show the experimental results of the Cash and Revenue reports illustrated in the previous section in the *Phone Simulator v7*. Depending on the input dates and billing cycle, the report should display different results.

The experiment is not to evaluate the efficiency of the system because the system depends on the existing back-end application in *J2EE*, *JBoss* Application Server, and *MySQL* DB server. The experimental result is to show that transactional and financial system can be expanded to mobile environment and implemented in *WML, cHTML* or *XHTML-MP*. Then, we actually want to compare them.



(a) Revenue Report Input (b) Revenue Report Result

Figure 6.4 Revenue Report on WML

The user enters the start date and end date to retrieve data for the cash report as in Figures 6.1-3 (a). Since *cHTML* and *XHTML-MP* are originated from *HTML*, in our example, the input and its result have the same look.

The user interface built in *WML* needs two phases input process for some mobile devices such as *Nokia 40 series* shown in Figure 6.2 (a) but in *Phone Simulator v7* of Figures 6.1 and 6.3 (a), it is the one phase process. In *Nokia 40 series* of Figure 6.2 (a), after entering the input data, the user needs to choose "go" button. It is inconvenient to the user, especially using the key of the mobile device - it may not be the problem to the traditional web user who uses desktop keyboard. However, the user interfaces in *Phone Simulator v7* including *WML*, *cHTML* and *XHTML-MP* of Figure 6.1-3 (a), after entering the input data, the user can choose either "go" or "submit" button respectively as the legacy web user interface.

(a) Revenue Report Input (b) Revenue Report Result

Figure 6.5 Revenue Report on cHTML and XHTML-MP

Then, after submitting the dates, its cash report of customers between those dates is displayed on the *LCD* panel of the mobile device, which is composed of each customer's Bill Start Date, Transaction ID, Account Name, Gross Amount, Credit Card Fee, Online Fee, Tax, and Net Cash calculated as in Figures 6.1-3 (b). The result displays the same page with the same data because all *WML*, *cHTML* and *XHTML-MP* pages display the same *JSP* page, *cashReportResult.jsp*. For the revenue report as in Figures 6.4 and 6.5 (a), the user enters the start date and end date and then selects its billing cycle to retrieve data. Same as cash report, the user interface in *WML* needs two phases process in some mobile devices but the user interfaces in *Phone Simulator v7* including *WML*, *cHTML* and *XHTML-MP* runs on one phase process. Then, after submitting the dates and its cycle, its revenue report of customers between those dates by the cycle is displayed on the LCD panel of the device, which is composed of each customer's Bill Cycle, Account Name, Gross Earned Amount calculated, Credit Card Fee, Online Fee, Tax, and Net Earned Revenue calculated as in Figures 6.4 and 6.5 (b). Again, its result page is the same *JSP* page, *revenueReportResult.jsp* as input pages in *WML*, *cHTML* and *XHTML-MP* request the same *JSP* page with the input from the user. Besides, *cHTML* and *XHTML-MP* have the same look of revenue input pages.

Table 6. 1 Comparing WML, cHTML, and XHTML-MP

	WML	**cHTML**	**XHTML MP**
Events Support	Yes (ontimer, onenterbackward, onenterforward etc)	NA	NA
Variable Declaration	Yes	NA	NA
Scripting Support	WMLScript	NA	ECMP near soon
Object Support	NA	NA	NA
Table Support	Yes	NA	Yes
CSS support	NA	NA	WAP-CSS
Coding Simplicity for XHTML developer	Difficult	Easy	Easy
Submitting input data (Form)	Indirect for some mobile devices (One or Two phases)	Direct (One phase)	Direct (One phase)
Displaying the result JSP page	Same as Others	Same as Others	Same as Others

Table 6.1 compares the applications built in *WML*, *cHTML* and *XHTML-MP* in the view of the user and developer as well as the language characteristics. As described in section *VI.A*, *WML* supports events and variable declaration statement but *cHTML* and *XHTML-MP* do not. *WML* and *XHTML-MP* supports table and scripts *WMLScript* and *ECMP* respectively. *XHTML-MP* supports *CSS* with *WAP-CSS* but others do not. However, all do not support object. In order to implement input document, *cHTML* and *XHTML-MP* are much easier for the legacy *HTML* developer because *cHTML* and *XHTML-MP* are originated from *HTML*.

For the simplicity to submit input data as described above, in *Phone Simulator v7*, *WML*, *cHTML*, and *XHTML-MP* only need one phase process but *WML* in some mobile devices needs two phase processes that are inconvenient for the user of mobile device such as cellular phone. However, the result displays the same page with the same data as the input pages request the same JSP pages: *cashReportResult.jsp* or *revenueReportResult.jsp*. Besides, *cHTML* and *XHTML-MP* input pages have the same look. However, *cHTML* is proprietary dependent so that it may have platform dependency issue.

VII. CONCLUSIONS

e-Business has been adopted for business as *Internet* became the part of our life. Besides, as wireless communication has grown up, *m-Commerce* gets more important and popular. And, *WAP* has been the standard for *m-Commerce* application. Thus, many organizations

have studied the way how to provide contents to mobile devices so that some organizations presented Markup languages. *NTT DoCoMo* presented *cHTML* that is the subset of *HTML* so that it is easy to build by the legacy *HTML* developer but it is proprietary dependent. *W3C* presented *WML* that is similar to but different from *HTML*. Therefore, it is not convenient to use *WML* for the traditional *HTML* or *JSP* developers because the developer needs to learn *WML* as well as *HTML*. Thus, *XHTML-MP* has been introduced by *WAP 2.0* community, which is a sibling of *XHTML*.

The paper illustrates and compares *m-Commerce* application on *WML*, *cHTML*, and *XHTML-MP*. The application, financial report system, has been implemented on *n-tier* architecture for its *m-Commerce* system with *WAP*, which is extended from the legacy financial report system in *e-Business*.

The paper also presents the *m-Commerce* architecture and how it is implemented on *WML*, *cHTML*, and *XHTML-MP*. *cHTML* and *XHTML-MP* are easy to build because it has the similar syntax of *HTML* used in the legacy *e-Business* world. And, It is easy to adapt web *XHTML* document to *WAP cHTML* and *XHTML-MP* The developer just modifies or uses the existing *XHTML* document for *cHTML* and *XHTML-MP*. Besides, for some mobile devices, in order to submit the user's data, user interface in *cHTML* and *XHTML-MP* is much easier than one in *WML*. The user interface in *WML* needs two phase processes that are composed of data entry page and *"go"* page. However, the user interface in *cHTML* and *XHTML-MP* only needs one phase process by submitting form page.

REFERENCES

[1] e-Business Center , "Building a Better e-Business Infrastructure: N-tier Architecture Improves Scalability, Availability and Ease of Integration", Intel® e-Business Center White Paper, http://dme.uma.pt/jcardoso/Teaching/ASI/N-tier%20Architecture/N-tier%20Architectures-Intel.pdf, 2001

[2] Microsoft Corporation, "Technology Overview for .NET Framework 1.1", http://msdn2.microsoft.com/en-us/netframework/aa497336.aspx

[3] D Rahmel, ".NET Framework", McGraw Hill, ISBN 0-07-,219466-9, 1st edition, February 14, 2002

[4] David Stutz, Ted Neward, Geoff Shilling, "Shared Source CLI Essentials", Oreilly, ISBN 13: 9780596003517, First Edition March 2003

[5] Middleware Company, "The Petstore Revisited: J2EE vs .NET Application Server Performance Benchmark", http://www.middleware-company.com/j2eedotnetbench, Oct. 2002.

[6] Indran Naick, Luca Amato, Jason K O'Brien, Jim Nicolson, Tsutomu Oya, "Design Considerations: From Client/Server Applications to e-business Applications", http://www.redbooks.ibm.com/redbooks/SG245503.html, Dec 1999

[7] Nokia, "Nokia Mobile Browser Development SDK 4.0", http://www.forum.nokia.com, 2002

[8] Nokia, "Nokia WAP Gateway simulator 4.0", http://www.forum.nokia.com, 2002

[9] Brian R. Smith, Charles Ackeifi, Thomas G. Bradford, Prabhakar Gopalan, Jennifer Maynard, Abdulamir Mryhij, "IBM e-business Technology, Solution, and Design Overview", http://www.redbooks.ibm.com/redbooks/SG246248.html, August 2003

[10] Sun Microsystems, Inc, "Java 2 Platform, Enterprise Edition (J2EE)", http://java.sun.com/j2ee/, 2007

[11] Open Mobile Alliance, "WAP Forum", http://www.wapforum.org/, 2007

[12] IBM, "IBM WebSphere", http://www-306.ibm.com/software/websphere/, 2007

[13] K Read, F Maurer, "Developing mobile wireless applications", Internet Computing, IEEE, Volume: 7, Issue: 1, pp81- 86, Jan/Feb 2003

[14] Wang, J.J. Song, Z. Lei, P. Sheriff, R.E, "Design and Evaluation of M-Commerce Applications", Communications, 2005 Asia-Pacific Conference, pp 745-749, Oct. 2005

[15] RG Duncan and MM Shabot, "Secure remote access to a clinical data repository using a wireless personal digital assistant (PDA)", Proceedings in AMIA (American Medical Informatics Association) Symposium, 2000

[16] EA Mendonca, ES Chen, PD Stetson, LK McKnight, "Approach to mobile information and communication for health care", International Journal of Medical Information;, Vol 3(7-8): pp631-638, Aug 2004

[17] WWWa, "XHTML", http://www.w3.org/MarkUp/, 2007

[18] Developers' Home, "XHTML Mobile Profile / XHTML MP Tutorial", http://www.developershome.com/wap/xhtmlmp/, 2007

[19] NTT DoCoMo, "Overview of i-mode Compatible HTML", http://www.nttdocomo.co.jp/english/service/imode/make/content/html/about/index.html, 2007

[20] Openwave, "Openwave Phone Simulator", http://developer.openwave.com/dvl/tools_and_sdk/phone_simulator/, Openwave Developer Network 2007

Prof. Jongwook Woo is currently an Assistant Professor at Computer Information Systems Department of California State University, Los Angeles. He received the BS and the MS degree, both in Electronic Engineering from Yonsei University in 1989 and 1991, respectively. He obtained his second MS degree in Computer Science and received the PhD degree in Computer Engineering, both from University of Southern California in 1998 and 2001, respectively. His research interests are Integrated Information Systems, static analysis in Objected-Oriented language, Parallel and Distributed Computing, e-Business and bioinformatics applications in n-Tier Architecture, and load-sharing algorithm in IPv6.

Prof. Minseok Jang is currently a professor of Computer Information Engineering Department at Kunsan National University, which is located in Kunsan, Korea. He received the BS, MS and the PhD degree in Electronic Engineering from Yonsei University in 1989, 1991, and 1999 respectively. His research interests are Wireless Network, XML, VoiceXML, and e-Business.