

Delivering Mobile Enterprise Applications on iMMS Framework

Jun Shen, Pei Sun, Changjie Guo, Ying Yin, Song Song
IBM China Research Laboratory
4/F, HaoHai Building, No.7, 5th Street, ShangDi
Beijing 100085, P.R.China
{shenjun, sunpei, guocj, yinying, soong}@cn.ibm.com

ABSTRACT

Mobile messaging services like SMS, MMS and wireless instant messaging services have become the major revenue source of the wireless data services. Compared with the online web access model, messaging access is more preferred in the wireless environment because messaging systems work in an asynchronous way and can tolerate intermittent connectivity. Current mobile messaging focuses on consumer applications, e.g., person-to-person communication. To leverage mobile message usage in mobile enterprise applications, iMMS (Interactive Multimedia Messaging Service) is proposed in this paper, which adds interactive features into MMS standard to provide rich and friendly user interface, local interaction and offline operation. In this paper, a message package based programming model is introduced to support interactive messaging application development. A design of an iMMS framework based on this programming model is presented. The architecture of the framework is also discussed. Finally a mobile workflow solution is given to show the implementation of a mobile enterprise application on the framework.

Categories and Subject Descriptors

D.4.4 [Operating Systems]: Communications Management – *message sending, network communication.*

General Terms

Management, Design.

Keywords

Interactive multimedia messaging service, messaging programming model, mobile enterprise application.

1. INTRODUCTION

Although browser based application like web application is very popular over Internet today, it has not achieved great success in the wireless network environment as expected because of the following limitations[1]:

MDM 2005 05 Ayia Napa Cyprus

(c) 2005 ACM 1-59593-041-8/05/05...\$5.00

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

- Availability: browser based applications can be used only when the network is connected. However for a mobile wireless device, the permanent connection to the wireless network can not be guaranteed. In another words, mobile wireless devices usually work in an intermittent network connection mode. As a result, a mobile user may not be able to access backend applications at the time they want via a mobile device.
- Usability: because browser based applications work in a click-wait-refresh model, after clicking a hyperlink on the web page, the user has to wait for the return of the requested web page. This process can be time consuming. It would become worse and intolerable when the wireless network performance is poor, thus deteriorating the user's wireless experience.
- Network efficiency: in the browser based applications, an entire HTML file has to be sent to a client even if there is only a little change on the web page since the last access. This increases the amount of network traffic and the user's network usage cost.

Compared with browser based applications, messaging services have won great success in the mobile application market since they score over web applications in terms of simplicity, accessibility and ease-of-use. Messaging is also an ideal channel for delivering mobile applications to end users. However, most current messaging services focus nearly on person-to-person (P2P) communications and very limited consumer applications. In order to leverage mobile message usage in enterprise, an enhanced messaging service, iMMS (Interactive Multimedia Messaging Service) is proposed in this paper. It adds interactive features into Multimedia Messaging Service (MMS)[4][5][6] standard to provide richer and more friendly user interface, local interaction and offline operation.

In this paper, we will first introduce the iMMS definition and its programming model. Then a detailed description of the iMMS framework is presented. The design of a mobile iMMS enterprise workflow based on this framework will be explained in section 5. The conclusion and our future work will be given at the end of this paper.

2. INTERACTIVE MULTIMEDIA MESSAGING SERVICES

iMMS[2] is the extension of MMS by adding interactive features. Although MMS supports rich content such as longer texts, color

pictures, animations, even audios and video streams, it doesn't solve the user interface problems that SMS and MMS are facing. Both SMS and MMS deal with only the message presentation to the end user with either pure text based or multimedia content based. The message contents are still static and less user interaction is involved. Compared with the browser user interface where the user can fill the forms, make selections, click buttons to send requests to the web applications, the MMS user interface is rather poor because the current MMS standard does not support those interactive elements. By means of the current MMS features it is difficult to extend its usage into more diverse and sophisticated business application scenarios.

In order to address the issue of extending MMS usage, iMMS enhances MMS interactive capability in three aspects:

- Rich Presentation

Synchronized multimedia integration language (SMIL)[7] is a simple language to describe the timing and synchronization relationships among the multimedia objects within an MMS message but it lacks the capability to provide the logical relationship among these objects and supports less user interaction. To solve the issues addressed above, iMMS leverages existing Internet technology in MMS standard by adopting XHTML[8] as its presentation language.

- Local Interaction

To extend MMS interactive capabilities, a set of visual controls are introduced in iMMS for providing an interactive user interface between a user and an MMS client. For example, forms—which support multiple kinds of controls, such as text input, buttons, and selection lists—are important because they are the enablers for the end user to 'talk' with backend applications. Behind this local interaction, there is also a communication interface defined between an MMS client and a back-end server. With the help of iMMS technology, it would be easier to implement many new kinds of interactive messaging applications that go far beyond those built with simple notification and button pressing.

- Disconnected Operation

The iMMS local interactive feature also supports disconnected operation. When an iMMS terminal is disconnected from a remote server such as losing the network signal, the user can continue to use received iMMS messages on the mobile terminal and submit a request to the server. The request will be sent out automatically when the network connection is recovered. Unlike the browser based applications which require the always-on network connection, MMS is a messaging system which also works under the intermittent connected network environment. The messaging user does not have to be aware of the message transmission status before the whole message has been delivered and received, the communication delay is imperceptible. This reduces the application requirement and dependence on the network performance and makes MMS as convenient and friendly to be used as SMS.

As one of the implementations for an iMMS message, shown in Figure 1, an iMMS package is designed to include three kinds of objects: a profile file, data files, and XHTML files including embedded objects such as JavaScript and images. The profile file is a text file which describes the basic information of the message

such as message ID, caption, status, and the name of the root of the XHTML file. The data file and XHTML are adopted in order to separate the message presentation and message data. The data file is used to store message data for two purposes. One is for dynamic message building, i.e. the message could be rendered on the client when there is only an updated data file without a profile or presentation file being sent to the client. The other is to store the user input and user local interaction. The XHTML file is used for message presentation and an embedded JavaScript is added to implement the local control logic. iMMS uses the same message encapsulation protocol and communication protocol to process and transmit iMMS package as MMS.

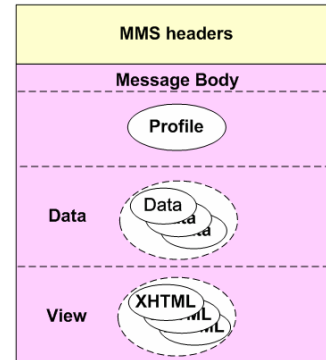


Figure 1. iMMS Package Structure

3. iMMS PROGRAMMING MODEL

Due to the different requirements under different network connection conditions, the iMMS programming model is different from the web programming model. A browser based application is an online program while an iMMS application is an offline program. As shown in Figure 2, the web application programming model is a web page based programming model. An application is constructed by a set of web pages like a tree. Users access the application by browsing the pages along the tree. As for messaging applications, the tree is divided into several sub-trees according to its functions and business logic, and each sub-tree is packaged into a message and delivered to an iMMS client. On the client side, a user can browse the message and perform the sub-function locally or communicate with the iMMS middleware to perform other sub business logic.

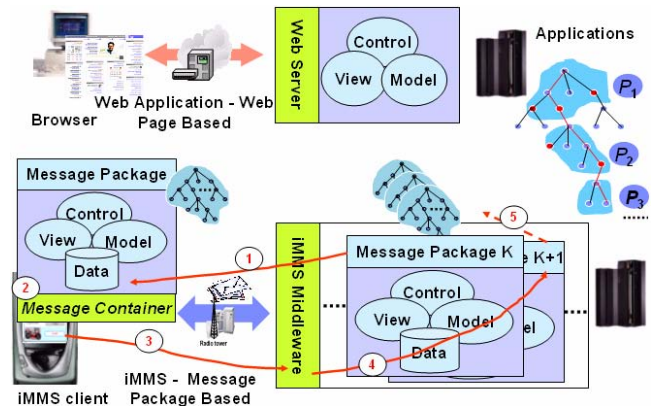


Figure 2. iMMS Programming Model

We called this iMMS programming model as a message package based programming model. A message container is designed on the client side to implement this programming model. When an iMMS package is sent to the client, the client decodes the package to perform the logic and returns the response to the iMMS middleware. Then the middleware will parse the response package, execute the application logic, generate and send the replied iMMS package to the client. This interaction cycle will be performed continuously until the required job is done.

4. iMMS FRAMEWORK

An iMMS framework has been designed to support interactive messaging applications. The framework includes an iMMS client and an iMMS application server. We've implemented the client prototype on Windows Mobile 2003 phone edition.

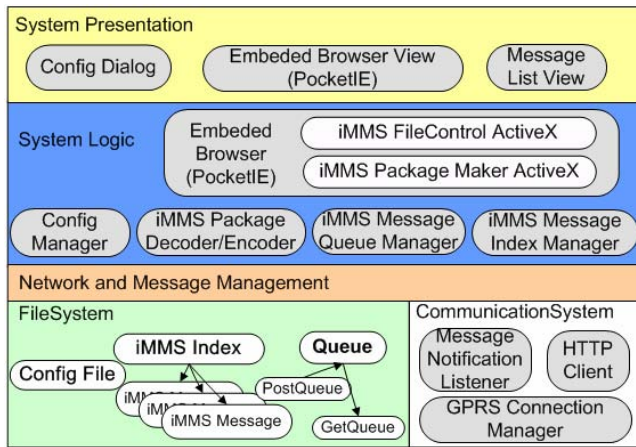


Figure 3. iMMS Client for PocketPC 2003 Phone Edition

As illustrated in Figure 3, the client architecture is divided into three levels:

- System Presentation

User interfaces (UI) are provided at this level for an end user to browse message list, read message abstract and content. It also provides the interface for the iMMS user to set system configurations. An iMMS List class is implemented to browse the

basic information of a message such as message receiving time, subject, status, and the message sender address. An XHTML browser should be designed to render the iMMS content since iMMS message adopts XHTML as the presentation language. In this prototype, Pocket IE is embedded in the iMMS client to provide this functionality. And the interaction capability is implemented by the Pocket IE by invoking JavaScript embedded in XHTML files. Also a configuration dialog is designed to set the system parameters such as the URL of an MMS center, user name and password, HTTP timeout, dialing policy, SSL enable or not, the threshold value of memory size, etc.

- System Logic

One important feature of iMMS is to enable the iMMS client to perform business logic locally on the client side. The presentation content of iMMS is XHTML files and the business logic is described with JavaScript in XHTML. Pocket IE is embedded in the iMMS client to interpret XHTML and JavaScript. In addition, an iMMS parser is included at this level for encoding and decoding an iMMS package.

- Network and Message Management

Network communication provides the support for SMS, GPRS and HTTP communications. Like MMS, iMMS adopts SMS as a notification trigger for retrieving an incoming message. When a notification SMS arrives at a mobile terminal, the iMMS client is invoked to retrieve the iMMS package from the MMS center indicated by the URL information in the SMS notification message.

Message management is responsible for managing three kinds of messages: the SMS notification message, the received iMMS Package and the post iMMS Package. Two messaging queues, GetQueue and PostQueue, are designed to ensure the reliable SMS receiving and Post message sending. The queues are stored in the PocketPC's file system to prevent the information loss even when iMMS client is in abnormal status. An index file is used to record every received iMMS package. Each record includes the following information: the absolute path of the message stored on the device, receiving time, subject, status, sender address, etc. The index file is used for the client to perform operations like adding or deleting a message.

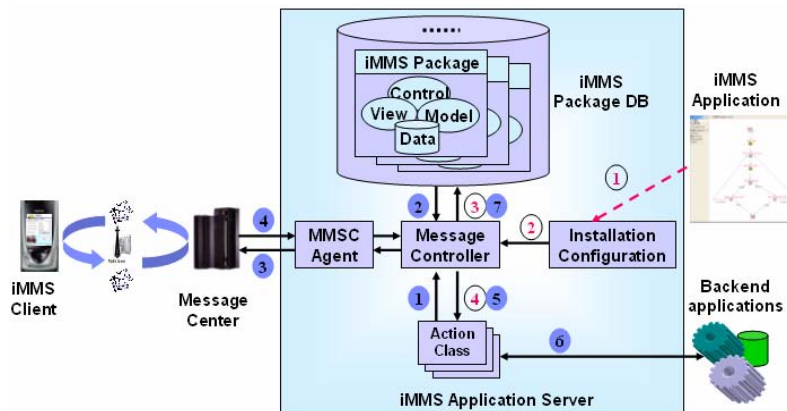


Figure 4. iMMS Application Server Architecture

The iMMS Application Server prototype is implemented on a J2EE platform to provide an iMMS runtime system as shown in

Figure 4. Five components are designed to implement the application server:

- (1) Message Controller: it's the major thread of the application server to start and initialize the application server, parse iMMS packages, and invoke the related components.
- (2) MMSC Agent: it is the communication interface to send and receive MMS messages to/from a message center, which provides SMS and MMS communication channels.
- (3) Installation & Configuration: it is the deployment component of the application server which reads an installation file, unpacks the file and stores iMMS packages into iMMS package database, as shown in Figure 4. Then the Message Controller reads the deployment file and starts the initialized action of the application.
- (4) iMMS Package Storage: it manages the installed iMMS packages and stores the session data for each application on the server.
- (5) Action component: it parses iMMS client requests, executes the related action class, updates the session data, and manages the action processing thread pool.

An iMMS application is composed of three basic components:

- (1) iMMS message packages which are used for iMMS message presentation and interaction.
- (2) Message action classes which perform the application logic to respond a user's interaction requests.
- (3) A configuration file which describes the timing sequence among the iMMS packages.

An example process of an iMMS application deployment and execution is like the following:

- (1) An installation file is read by Installation Component. Then it is decomposed and stored in the package database.
- (2) The Message Controller reads the deployment file and starts the initialized action class.

- (3) The initialized action sends an iMMS package to an iMMS client.
- (4) A user reads the iMMS message, inputs data and clicks submit button to send his request to the server after receiving an iMMS message.
- (5) The Message Controller parses the request, calls the corresponding action class to process the user request.
- (6) The selected action class either performs the server side logic or invokes backend applications, then returns the result data to the Message Controller.
- (7) The Message Controller selects the next iMMS message package from the package database, generates a new iMMS package using the result data from the Action Class.
- (8) The Message Controller delivers the new iMMS package to the mobile client through MMSC Agent, and waits for another cycle of interaction.

5. MOBILE WORKFLOW SOLUTION

Workflow is a typical enterprise application. Most of the existing workflow applications are web or Domino based applications which can not support mobile phone very well for the reason of limited wireless network performance, web page size, JavaScript and Java applet supports, etc. Therefore a mobile Domino workflow solution is designed on the iMMS framework to improve the mobile user experience by providing almost in time message delivery, rich local interaction, and offline operation.

The architecture of the iMMS mobile workflow solution is illustrated in Figure 5. Three components have been developed: iMMS Workflow Templates, Workflow Connector, and iMMS Workflow Adaptor for Domino.

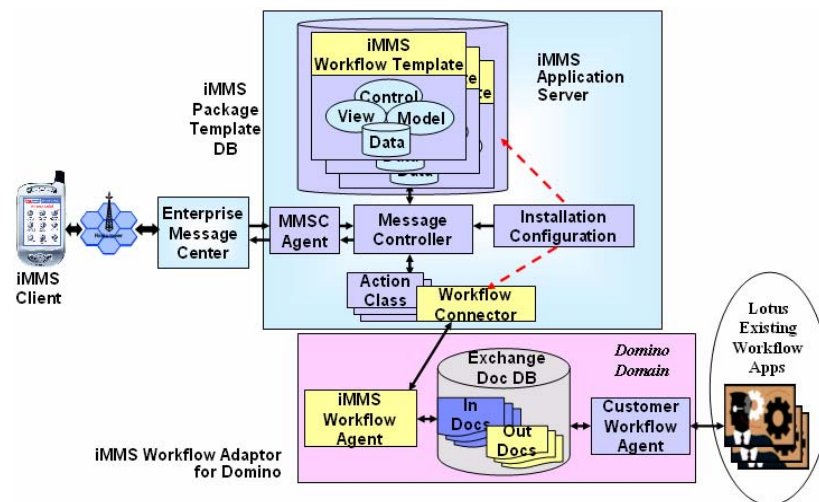


Figure 5. Mobile Workflow Solution Architecture

iMMS Workflow Templates is used to define iMMS packages. Workflow Connector is responsible for delivering information between the iMMS users and the workflow adaptor. These two components are deployed on iMMS Application Server. iMMS Workflow Adaptor is for automatically detecting the workflow activities. It forwards the mobile activities to Workflow Connector,

and sends users' response back to the existing workflow applications to enable the workflow activities to move on to the next step. An Exchange Doc Database is designed in this adaptor to isolate iMMS Workflow Agent and Customer Workflow Agent. The two agents exchange workflow data by accessing In-Doc and Out-Doc documents in the database. Because different workflow

applications have different implementation approaches and communication interfaces, the Customer Workflow Agent should be redesigned according to the architecture of existing workflow applications while keeping the iMMS Workflow Connector and Agent unchanged. A new iMMS workflow adaptor should also be re-designed to support other type of workflow applications, e.g. Exchange and WBI (WebSphere Business Integration).



Figure 6. Mobile Workflow Scenario

Figure 6 shows an iMMS mobile workflow scenario. When an urgent business workflow task is submitted by an employee, an iMMS message is pushed to his manager's iMMS client. The iMMS message contains both the task information and the possible actions for that task. The iMMS client parses the iMMS message and renders the workflow task on the mobile device. The iMMS message, as illustrated in Figure 7, contains the control elements such as a text input field and some buttons to help the mobile user to interact with the back end application efficiently. This mobile solution also allows the mobile user to accomplish the workflow tasks in an offline model which means that mobile users can save the related workflow as processing data on their mobile devices and retrieve them late when there is a need. Once the user submits the request, all the work for data transmission will be handled by the iMMS client. Because the MMS system guarantees the delivery of iMMS messages, each user's submission could be regarded as a success delivery. There is no need for the user to worry about the network connection status.

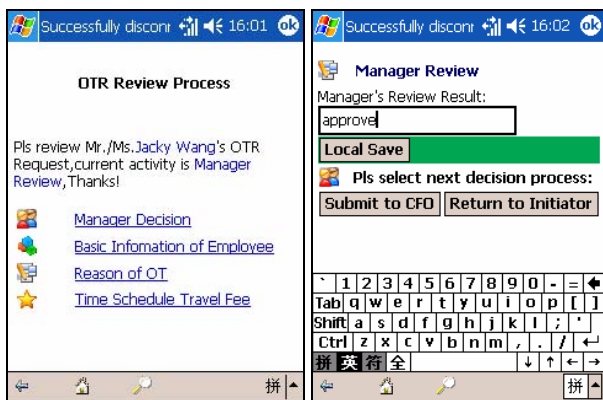


Figure 7. Mobile Workflow User Interface on PocketPC

6. CONCLUSION AND FUTURE WORK

In general, iMMS is an enhancement to the existing MMS with valuable interactive features added. It enriches message presentation while still keeping the messaging notification and

offline messaging operation capabilities. iMMS not only improves the mobile user experience, but also opens a new approach of deploying diverse and complex messaging based mobile applications, both for consumer and for enterprise applications.

The message package based programming model and the corresponding iMMS framework are introduced in this paper. As one of the typical mobile enterprise solutions, an iMMS based Domino mobile workflow is described as well. This iMMS based mobile solution not only enriches mobile user experience by MMS interactive capability, but also shows an example of deploying iMMS based applications over the iMMS framework.

The ease of use of the development toolkit is one of the key factors for the popularity of iMMS applications. It will be one of the focus points of our research. Also a migration tool that could automatically or semi-automatically convert the existing web applications into iMMS applications will be another potential research topic.

7. REFERENCES

- [1] Lai Jin, Takashi Sakairi, "An e-Business Framework for Developing Rich and Reliable Client Applications", Systems, Man and Cybernetics, 2003. IEEE International Conference on, Volume: 5, 5-8 Oct. 2003, Pages:4294 - 4299 vol.5.
- [2] Jun Shen, Pei Sun, Jianming Zhang, Song Song, "iMMS - Interactive Multimedia Messaging Services", IBM Journal of Research and Development, Vol. 48 No. 5/6 September/November 2004.
- [3] M. Blount, V. Perret, D. Yeh, A. Purakayastha, M. Moser, Y. Duponchel, D. Bourges, M. Graf, "Managed Portal Appliance: An Experiment in Extending the Reach of Web Applications", Proceedings of the 2004 IEEE International Conference on Mobile Data Management.
- [4] "WAP MMS architecture overview," WAP-205-MMSArchOverview-20010425-a, *WAP Forum*, <http://www.openmobilealliance.org>.
- [5] "WAP MMS client transactions," WAP-206-MMSCTR-20010612-a, *WAP Forum*, <http://www.openmobilealliance.org>.
- [6] "WAP MMS encapsulation protocol," WAP-209-MMSEncapsulation, *WAP Forum*, <http://www.openmobilealliance.org>, Sep. 14, 2000.
- [7] W3C Recommendation: "Synchronized Multimedia Integration Language (SMIL 2.0)," Aug. 7, 2001. URL: <http://www.w3.org/TR/2001/smil20>.
- [8] "XHTML Mobile Profile 1.1", WAP-XHTMLMP-V1_1-20020904-C, *WAP Forum*, <http://www.openmobilealliance.org>.
- [9] Lars Novak and Magnus Svensson, "MMS - Building on the success of SMS," *Ericsson Review*, vol. 78(2001): 3, pp. 102-109.
- [10] Backweb-The Offline Infrastructure Company, <http://www.backweb.com>.
- [11] Curl Corp., <http://www.curl.com>