Mobile Services for Java-enabled Devices on 3G Wireless Networks

Dr. Silvano Maffeis, CTO, Softwired AG
silvano.maffeis@softwired-inc.com
http://www.softwired-inc.com/

1 Introduction

The usage of mobile devices has outpaced that of PCs, with mobile phone usage surpassing PC ownership. Gartner group estimates that by 2003, there will be 1 billion mobile phone users and by then, the mobile phone will have become the "de-facto standard consumer e-commerce device".

In January 2001, NTT DoCoMo launched its iAppli service, allowing mobile users to download Java applications on their iMode phones. By October of the same year, 7 million users had signed up for the service.

Nokia announced plans to deliver 50 million Java-enabled devices by the end of 2002, whilst the ARC analyst firm estimates that 180 million Java-enabled handsets will be in use in 2002, and more than 400 million in 2003.

The mobile handset industry is transitioning from growth of their user base as the primary driver of unit volume demand, to shorter product lifecycles and increasing demand for richer interactive applications on the device. Companies such as Siemens, Motorola, NTT, Nokia, Compaq, and HP are working on a new generation of mobile device, the so-called "communicator". A communicator is a mobile device bigger than a phone, but smaller than a Pocket PC. It combines both voice and wireless data capabilities.

Figure 1: Some of the available communicators.

A related concept is that of the "smart phone". A smart phone has the same compact size of a phone and is able to run Java applications (MIDlets) and to play MP3 files.

Figure 2: Some of the available smart phones.

Communicators and smart phones will be used not only for running typical PDA applications, such as Calendar, ToDo, Calculator, and so on, but also for running interactive, network-aware applications: Multiuser games, picture messaging, real-time quotes, location services and maps, file sharing, audio and video streaming, chatting, mobile commerce, and mobile finance. These applications will allow for a much richer user experience than WAP: they are highly interactive, secure, and place "intelligence" right on the device, in the form of Java applications. That "intelligence" allows the user to keep on working with the application even when there is no network coverage, or to click on a map to spot traffic congestions, or to render games and animations.

Importantly, interactive mobile applications require advanced "middleware infrastructure software" for connecting the Java applications running on the mobile devices (communicator or smart phone), to the "service" applications running in a data center, by using a 2.5 or third generation wireless network.

2 Mobile Headaches

Developing this new type of mobile application is challenging for the following reasons:

- **So many mobile operating systems to pick from**. The mobile operating system market is fragmented, and will remain in this state in the mid to longer term. The contenders in this market are Microsoft's PocketPC, PalmOS, Symbian, Embedded Linux, as well as JavaPhone operating systems. A company wanting to seize a substantial user base for a mobile service might thus need to support more than one mobile operating system. Adapting the software for various platforms can be very expensive and time consuming, unless the right middleware tools are used.
- **So many wireless standards**. GSM-Data, GPRS, UMTS, CDMA, W-CDMA, CDPD, Wireless LAN, Mobitex, you name it! There are so many incompatible wireless network standards. Rolling out a mobile service worldwide, or even in one single country, requires your software to support multiple networks. Again, without the right middleware, such endeavor is going to be expensive.
- **Wireless issues**. Do you synchronize your PDA over mobile phone, or have you tried to surf the Web when riding on the train? Let's face it, wireless networks have some very user-"**un**friendly" characteristics: Shadows in network coverage, variations of the available bandwidth, sudden disconnections, or intermittent communication links between your mobile device and your server. The right wireless middleware will help your mobile services cope with these problems, and will give the user the impression that the network is very fast and "always there"!
- **User expectations**. Users have been disappointed by WAP whilst expecting from a communicator the same "quality of service" they get from a PC connected to the Internet via cable. This means, applications on a communicator should not throw error windows at the user's face when network coverage is lost, and the application should always be responsive to the user's input. Also, mobile commerce transactions should neither be lost nor duplicated if they are submitted while the device is out of coverage. So be prepared for high expectations and for calls to your support lines when your mobile application doesn't work as reliably as the user wants.
- **Vendor independence**. Of course, your mobile applications should not tie you to a single hardware manufacturer, software vendor, or network operator.
- **Time-to-market**. Communicators and smart phones have started to appear months ago. GPRS is a reality in many countries already. Your competitor has started to develop tools for mobilizing his work force. Setting up a mobile applications platform on which your company can host mobile services takes time for training, system integration, software development, testing, and deployment. So you might have to start now, if you want to roll out the first applications within the next 12 to 18 months!

3 Meeting the Challenge

Mobile services for communicator devices require that you develop software to run on the handheld as well as server software to run in the data center. Moreover, you need to link-in existing information systems, and to figure out how to connect the mobile client applications to the server applications. Doing that in an acceptable time frame requires the right middleware tools. Only middleware allows you to develop mobile services by addressing the problems we outlined under "Mobile Headaches".

3.1 What is Middleware?

Middleware is a "glue" layer of software between the network and the applications. This software provides services such as data transportation, service roaming, directory lookup, and security. Middleware works much like the "postal system", it allows applications to communicate more easily and effectively. Not using middleware would be as if one would have to deliver the mail oneself!

In the mobile world, two classes of middleware are of particular interest:

- **Data synchronization middleware**: For synchronizing addresses, e-mail, and databases on mobile devices. This type of middleware is appropriate for "**data centric**" applications. The application on the mobile device stores data in a local database. The data synchronization middleware synchronizes that data with the data located on a server. Synchronization typically requires user intervention, and is neither suitable for interactive applications, nor for transmitting time-critical information such as stock quotes.

- **Message-oriented middleware (MOM)**: Applications on mobile devices can communicate with other applications (mobile or not), by exchanging so-called messages. This works much like e-mail, and is used to transport information from one application to another, in a fast, reliable, and efficient manner. MOM is for "**interaction centric**" applications and works much like a high-speed postal system. The advantage of MOM is that messages are delivered immediately over a wireless network, provided there is network coverage. If not, messages are stored on the device, and are automatically forwarded as soon as network coverage is granted. Therefore, MOM allows the development of highly interactive applications, whilst allowing mobile application to be operated also in "disconnected mode". According to Gartner, Wireless-MOM will emerge "as the dominant form of communication middleware for linking mobile and enterprise applications"[1].

3.2 Picking the Right Standards

Minimized vendor dependence means reducing the overall business and technology risks of rolling out a mobile service. Therefore companies should stick to established standards when choosing wireless tools. Today, mobile middleware tools are highly vendor dependent, with few exceptions. For data synchronization middleware, SyncML is the de-facto standard. For Wireless MOM, "Wireless JMS" (Java Message Service) is the de-facto standard.

An elegant and future-proof solution is to deploy Java technology on both the mobile devices, as well as on the server side. On the mobile devices, Motorola, Nokia, Siemens, and other device manufacturers have chosen the Java-2 Micro Edition (J2ME). On the server side, the Java-2 Enterprise Edition (J2EE) is the platform of choice. In a mobile world, Wireless JMS provides the natural link between these two technologies.

3.3 The Mobile Java Applications Platform

By combining J2ME, Wireless JMS, and J2EE we obtain a standards-based, end-to-end Java solution, for delivering interactive services to mobile devices. The resulting architecture is depicted in the illustration.

---

1 *Do MOM, ORBs and Data Access Middleware Suit Mobile?* Gartner Report T-14-3936.

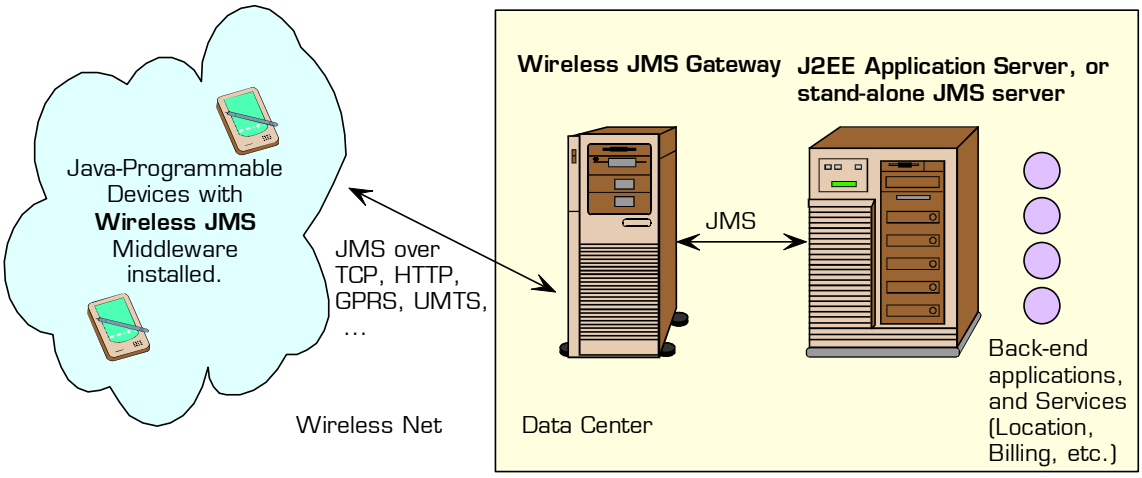

Figure 3: Standards-Based, Mobile Java Applications Platform.

In this architecture, the server-side of mobile applications is deployed on a J2EE applications server such as BEA WebLogic, IBM WebSphere, or JBoss. Alternatively to a full-fledged J2EE server, a stand-alone JMS server such as iBus//MessageServer or SwiftMQ can be deployed. The J2EE server is used also for integrating location-, profiling-, and billing services into the platform. A J2EE application server can provide the right "connectors" for accessing existing information systems (ERP, CRM, Logistics, etc.).

The client-side of the mobile service is deployed directly on a Java-enabled device. Also on the device, a Wireless JMS middleware component such as Softwired's iBus//Mobile is deployed. The middleware component communicates with the Wireless JMS gateway on the server side of the system.

4 Application Areas

The application areas of this Mobile Applications Platform are manifold:

- **Business-to-Employee** (B2E) systems**,** such as mobile workforce automation, logistics, supply chain management, CRM, and so forth. These require customized mobile client applications, offline and online operation, mobile transaction processing, as well as applications to remain responsive even when network coverage is not granted for longer periods of time.
- **Business-to-Consumer** (B2C) systems, such as M-Banking, M-Finance, and M-Commerce. Only Wireless JMS can guarantee the exactly-once delivery of the transactions issued by a mobile user.
- **Machine-to-Machine** (M2M) communication as in embedded systems, remote-administration of vending machines, telematics, automotive computing, and manufacturing control. Messaging middleware supports the immediate delivery of signals and other important data between embedded devices, from embedded devices to servers, etc.

- **Peer-to-Peer** (P2P) applications, such as gaming, online gambling, file sharing, chatting, wireless dating, and so forth. Wireless JMS allows for more interaction, and thus for more fun!

5 Summary

Mobile applications for Java-enabled devices require powerful middleware tools in order to deliver on the promises of communicator devices and "smart phones". Only message-oriented middleware can ensure that mobile applications will function reliably on wireless networks, which are always subject to wireless coverage "gaps" and "holes". The companies that will have a leading role in provisioning interactive services to these new types of devices are those that have started putting together "mobile Java applications platforms" today. From the point of view of vendor neutrality, time-to-market, and positive user experience, we believe that such mobile Java applications platform should be based upon a "Wireless JMS" middleware, and upon the J2ME and J2EE standards.

Wireless MOM Resources

For further information about using Java messaging middleware technology for building mobile Java applications:

- **Software Download**: iBus//Mobile Standards Based Mobile Java Applications Platform (free evaluation):
  http://www.softwired-inc.com/
- **White Paper**: "JMS for Mobile Applications and Wireless Communication".
  http://www.softwired-inc.com/pdf/technology/4931_11.pdf
- **Book**: Chapter 12 of the Book "Professional Mobile Java Programming", Wrox Press, ISBN: 1861003897.

About the author:

Dr. Silvano Maffeis is CTO at Softwired AG. He can be reached at silvano.maffeis@softwired-inc.com. Softwired is a privately held software company headquartered in Zurich, Switzerland. Softwired is an expert in mobile Java application platforms, in communications middleware, as well as in developing mobile applications for Java-enabled devices. For more information, visit http://www.softwired-inc.com/