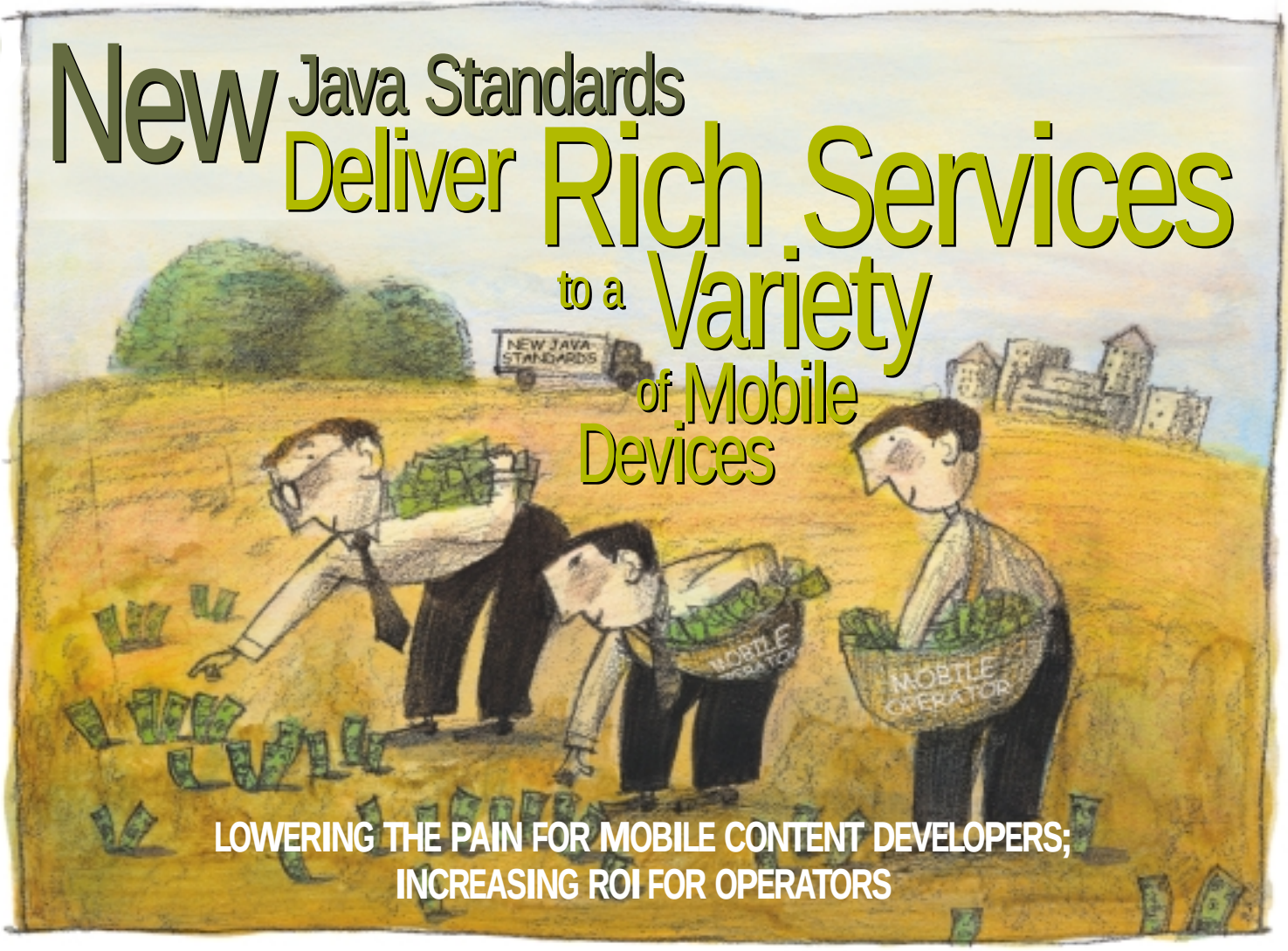
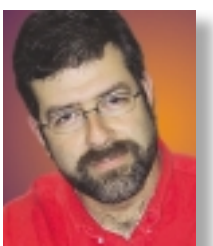
New Java Standards Deliver Rich Services to a Variety of Mobile Devices



LOWERING THE PAIN FOR MOBILE CONTENT DEVELOPERS; INCREASING ROI FOR OPERATORS



by Peter Bernard

Peter Bernard is vice president of Insignia, responsible for directing the definition, implementation, and evolution of Insignia's products for wireless carriers and mobile device manufacturers. During the last 13 years, Peter has held a variety of pioneering positions in the mobile enabling technology industry and has been granted a patent and has patents pending in several areas including power management and mobile security.

Peter.Bernard@insignia.com

Over the past year, members of the Java Community Process (JCP), an independent organization dedicated to developing standards for Java technology, have been hard at work ratifying new Java technology standards that make Java a more unified platform and a more portable environment.

If you take a quick look in any electronics shop or at any wireless operator Web site, you'll see the ever-expanding array of wireless devices reaching consumers. Operators are trying to reach the broadest possible base of customers, appealing to mobile professionals, gamers, teens, and anyone in between. The strategy for deploying mobile services is to leverage as many of these platforms as possible, delivering a range of mobile services to as varied a base of consumers as possible. But getting all the various devices and services to work together has, in short, become a developer's nightmare.

That's because even though Java technology was originally positioned as a "Write Once Run Anywhere" platform, it has often unfortunately been deployed without this capability because the list of operator and device-specific extensions designed to fill the functional holes in earlier Java specifications has become too exhaustive. To deal with this problem, developers had to re-write their Java applications using proprietary APIs to incorporate sound, PIM access, storage card access, and other features into their devices. Some manufacturers were even including proprietary APIs in their devices, choosing which device profiles and which optional packages to support. To make matters even more confusing, Java technology itself comes in various "flavors."

Some proprietary extensions will always be present in mobile Java devices because of innovations that inevitably take place and are simply too new to be standardized. But over the past year, members of the Java Community Process (JCP), an independent organization dedicated to developing standards for Java technology, have been hard at work ratifying new Java technology standards that make Java a more unified platform. They have also worked to make it a much more portable environment.

Today, most of the proprietary extensions developers used for first-generation mobile Java devices are being replaced with the new open standards. The new standards allow developers to focus their energies on a well-defined and open mobile Java platform that enables them to use their development time to their own advantage.

This article will describe the recently approved Java standards from MIDP 2.0 to Mobile Media API to PDA Profile. It will also show how the standards provide rich access to services on devices from feature phones to PDAs. See Table 1 for a summary of key areas of functionality that are currently being specified and implemented by the Java Community Process (JCP).

Feature	JSP#	Comment
Sound aPI	JSR 135	Mobile Media - MDI, Wave, etc.
Video API	JSR 135	Mobile Media - MPEG-4, etc.
Game API	JSR 118	MIDP 200Game API
SMS Messaging	JSR 120	Java Wireless
3D Grpahics	JSR 184	Java 3D API for Mobile Devices
Security	JSR 118	MIDP 2.00 HTTPS Support & API
Networking	JSR 118	Security

Table 1: Key areas of functionality that are currently being specified and implemented by the JCP

JSR 120 (JAVA WIRELESS MESSAGING)

The main features provided by the Java wireless API will be Short Message Service (SMS), Unstructured Supplementary Service Data (USSD), and Cell Broadcast Service (CBS). This JSR will enable Java developers to programmatically send and receive SMS messages, which is a necessary feature for many multiplayer games and other interactive services. This wireless messaging API provides standard access to wireless communication resources, allowing third-party developers to build intelligent connected Java applications.

JSR 135 (MOBILE MEDIA API)

The Mobile Media API JSR 135 specification provides a high-level interface to sound and multimedia capabilities of a device running J2ME. It enables multimedia functionality in J2ME applications and can be used by content developers to provide a wide range of new products and services.

JSR 135 addresses the need for a scalable set of multimedia services for various types of devices. For low-end devices, basic sound capabilities can be implemented, and for higher-end devices, video playback and graphics can be provided. Although the JSR 135 is not compatible with Java Media Framework (the Java multimedia API for desktop devices), Java developers will immediately recognize the similarities and will use it productively.

JSR 135 can be used by a wide range of MIDP applications, such as for games, animated messages, screen savers, custom user interfaces, and product visualization. It is important because it will eliminate the requirements of many OEM class libraries, removing a potential source of fragmentation from the J2ME standard.

JSR 184 (INTERACTIVE 3D API)

JSR 184 defines an optional J2ME MIDP interactive 3D graphics package that is efficient, compact, and designed for limited resource/graphics-powered devices. This package will sit alongside CLDC and MIDP with other optional packages like JSR135 .

A wide range of applications can use the 3D API, including: games, animated messages, screen savers, custom user interfaces, and product visualization. The 3D API will be easy for developers and content providers to use. It is targeted at both low-end and medium/high-end MIDP devices. Devices that utilize software implementations of floating point and 3D operations, as well as devices utilizing hardware floating point and DSP capabilities will be supported in a fully scalable manner.

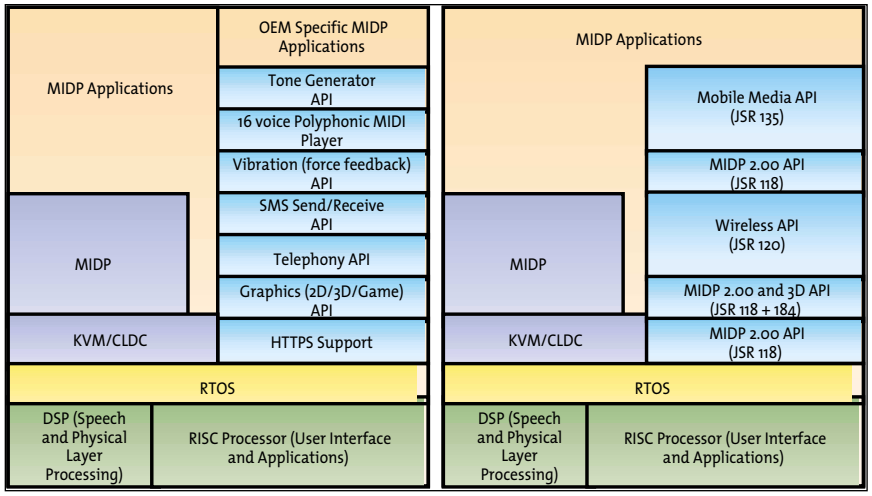
JSR 185 (JAVA TECHNOLOGY FOR THE WIRELESS INDUSTRY)

In the past two years, a number of JSRs that apply to the wireless communications industry have been initiated. A primary focus has been on the use of J2ME technologies in the wireless handset: MIDP (JSR 37), MIDP 2.00 (JSR 118), and Wireless Messaging API (JSR 120). In addition, a number of complementary efforts are currently underway with the development of JSR 135.

J2ME technologies have been very successful in the wireless industry. The MID Profile is incorporated into a wide variety of products. Not only are handset vendors supporting it, but implementations for PDAs are also available, and implementations for other devices are certain to follow.

What is not clear, and is not within the scope of any existing JSR, is how the various technologies associated with the MID Profile work together to form a complete handset solution for the wireless services industry. JSR 185 will describe an overall architecture that would include which optional packages fit with which profiles; how an end-to-end solution for interoperable Java applications could work; and how the migration of applications can occur to which profiles as the devices become more capable.

In addition, it would be beneficial if coordination of new JSRs and recommendations for new optional packages could take place within the context of the wireless development community as a whole. JSR 185 will provide an overall architectural description of a wireless client software stack. An integrated reference implementation (RI) and technology compatibility kit (TCK) bundle will be provided for the described technologies.



Tables 2 and 3: Once the implementation is readily available, developers will have a feature-rich J2ME platform standardized across many devices

> These new Java standards enable a variety of mobile services to be scaled across a variety of devices and to a variety of consumers.

JSR 75 (PDAP)

The PDA Profile (PDAP) will extend the functionality of MIDP by providing additional graphics APIs and the ability to access Personal Information Management (PIM) data as well as removable storage card access. The device requirements for memory will be increased allowing a minimum of 512KB total memory (ROM plus RAM) available for Java runtime and libraries. To accommodate applications developed using the Abstract Windowing Toolkit (AWT), a subset of AWT will also be included in the PDA Profile.

MIDP 2.00

MIDP 2.00 is the next generation of the MID Profile and should be available to the market in early 2003. Many new APIs have been added including functionality for application delivery, UI, multimedia, security, and networking. The Over-The-Air (OTA) provisioning is formally being added to the MIDP 2.00 specification and enhancements have been made to enable reliable delivery service. The whole area of MIDlet management has been improved and provides additional billing options for carriers and greater control for end users.

The user interface has also been updated with features that will provide developers with a much richer set of APIs. MIDP 2.00 adds custom item support, layout control, and graphical enhancements (including transparent image support). MIDP 2.00 will also include a rich set of Game APIs for 2D games and a Sound API based on a subset of JSR 135. This will allow for richer more compelling MIDlets than are possible today, which can be created without using proprietary OEM class libraries.

Although MIDP 2.00 contains many new features and capabilities, optional Java packages will still be necessary in many device types, as the profile does not try to be all things to all devices. This will give handset manufactures the ability to create devices with different functionality at different price points, keeping costs down, but increasing the capabilities of the platform.

KEY MIDP 2.00 ENHANCEMENTS

Where MIDP 1.00 showed the industry how well-suited Java technology is for the wireless market, MIDP 2.0 addresses the feedback from the developers, carriers, handset manufacturers, consumers, and competitors, by including features that truly enable revenue-generating applications and services.

One of the key complaints from developers about MIDP is the lack of functionality. Security on MIDP depends largely on a "sandbox" method to secure devices against viruses and other malicious code. Downloaded programs run in a quarantined area from which access to other software on a device is restricted. MIDP 1.00 limited the functionality of the API in certain areas, because additional security above and beyond the sandbox is needed. Some industry experts have said, "This sandbox approach to security is no security at all. It's a joke, so just forget it."

What is needed is a domain-based security framework that can ensure that Java applets delivered to a mobile device come from a legitimate source. MIDP 2.00 will support a public-key encryption system. A developer writing a Java MIDlet, such as a game or a currency converter, would add a digital signature using a unique, private encryption key. When the applet arrives at the device, the private key would be matched with its corresponding public key to verify that it was from a trusted source.

MIDP 2.00 will use the new security functions to create the concept of trusted and untrusted applications. Untrusted applications will use the same "sandbox" model as MIDP 1.00, and depending on the security policy of the device, will be limited to which APIs it can access. Trusted applications will use digital signing to allow the device to clearly identify the source of the downloaded MIDlet. Using this security scheme, permission can be granted to applications in a variety of ways. See Tables 2 and 3, which show that once the new Java technology standards are implemented, developers will have a feature-rich J2ME platform standardized across many devices.

Conclusion

These new standards enable a variety of mobile services to be scaled across a variety of devices and to a variety of consumers. And with the recent ratification of the CDC/Personal Profile standard, developers can also now leverage desktop Java development of enterprise applications onto PDAs, as a logical successor to PJAVA. Those same devices can be enabled with a CLDC/PDAP implementation to run mobile services content from an operator – the same content that can be provisioned and run on low-power feature phones.

This new level of portability, although not perfect by any means, will enable mobile content developers to significantly lower their development pain threshold. It will also enable operators to drive a significantly higher return on investment (ROI) of their data services infrastructure. But most of all, it will enable consumers to enjoy rich content for devices used at work and for entertainment.