Moving Java into Mobile Phones

George Lawton

As mobile technology matures, handheld-device vendors are looking for ways to make their products more functional, and Java is one approach they are turning to. This is particularly the case with smart cellular phones, which are using Java to help add new capabilities.

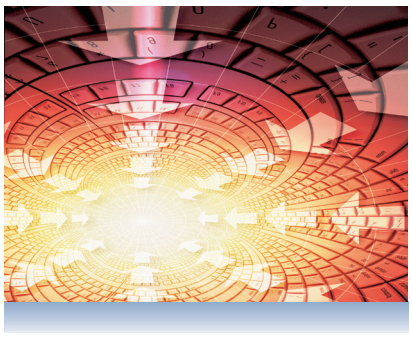
In smart phones, Java functions as a layer between the operating system and the hardware, or runs parallel to the OS within a separate chip.

In the past, the key constraint to running Java on mobile devices has been their processing, memory, and power-consumption limitations. However, new mobile hardware and software developments are reducing these limitations.

Thus, industry observers expect Java use in mobile devices, which is already supported by many vendors, to explode during the coming years. Nick Jones, a fellow at Gartner Inc., a market research firm, said Java will become a de facto standard on mid-range and high-end cellular phones. He predicted that at least 80 percent of mobile phones will support Java by 2006, although some may also run on other technologies, such as Microsoft's Pocket PC operating system.

According to Jones, mobile-device manufacturers' desire for an aftermarket is driving interest in Java as a mechanism for easily adding software to devices.

Java also permits applications to work across platforms. This is important in the mobile-phone market,



which features many platforms. However, questions about Java's performance and a dearth of Java-based applications for cellular phones, particularly in Europe and the US, remain as obstacles to the technology's widespread adoption in mobile devices.

DRIVING JAVA USE IN HANDHELDS

Work on Java-enabled handheld devices began several years ago, but completion of the Java 2 Platform Mobile Edition (J2ME) and support from device vendors and cellular-phone-service providers have driven the recent level of interest, explained Eric Chu, Sun Microsystems' group product manager for industry marketing.

Adoption levels

Korea's LG Telecom in became the first service provider to deploy Java in September 2000. Since then, users have deployed between 18 million and 20 million Java-enabled telephones, said Sun spokesperson Marie Domingo.

Companies such as Nextel in the US, NTT DoCoMo in Japan, and British Telecom are now selling Java phones. And, said Ben Wang, manager of systems development for Sprint PCS, 80 percent of the new phones the company sells will be Java enabled after the big rollout next month. Nokia alone plans to ship 50 million Java phones this year and 100 million next year. In fact, 15 handset makers either are or soon will be selling 50 models of Java phones.

Advantages

According to Sun's Chu, one of Java's major benefits for cellular phones is support for packet-based networks running TCP/IP. Using TCP/IP makes it easier to write applications that communicate directly with the phone, rather than relying on an intermediate technology such as the wireless application protocol (WAP). Also, Chu said, Java, unlike WAP, supports pictures and colors. In addition, he explained, the Java environment provides good security because it includes a sandbox that limits downloaded code's access to the rest of a host system.

Moreover, Java's ability to work with different platforms is important in the fragmented cellular-phone market. This capability lets a Java-enabled phone run applications and services written for other mobile platforms and also lets software vendors save time and money by writing a single, Java-based version of an application to run on multiple platforms. And Java-enabled phones and servers could communicate directly with each other, thereby enhancing interactive applications.

Java enables smart-phone users to download applications directly from the Internet. Similarly, Java lets users download Java applets that customize their devices in various ways, such as with special ring tones or improved caller ID. This lets users get new features more easily. In the past, users had to buy new phones, run new applications remotely using WAP, or download programs first downloaded to a PC.

Meanwhile, there are many Java developers, which makes it easier for

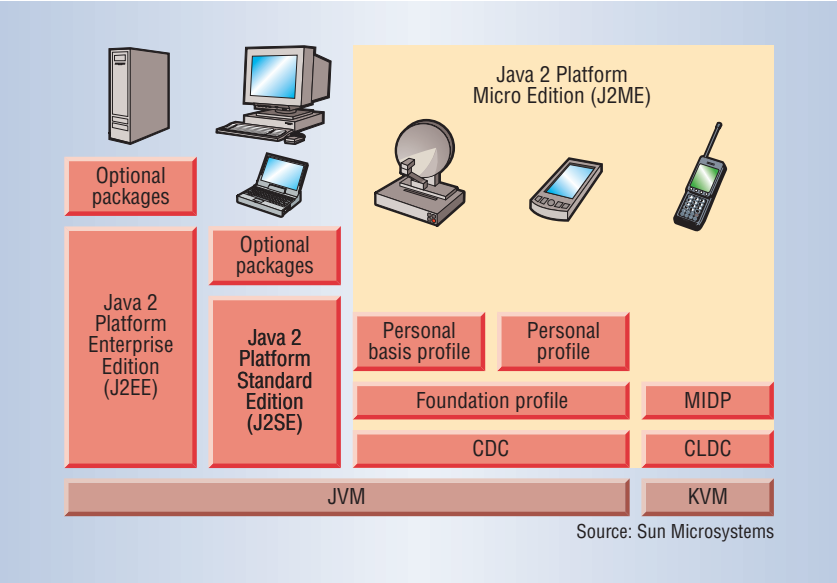


Figure 1. Sun's three primary Java platforms are each designed primarily to run on a different type of machine. The Java 2 Platform Enterprise Edition is designed for servers; the Java 2 Platform Standard Edition for workstations, PCs, and laptops; and the Java 2 Platform Micro Edition for PDAs, smart cellular phones, and other smaller systems. J2EE and J2SE use the full Java virtual machine (JVM). J2ME also works with the slimmed-down K virtual machine (KVM), the connected limited device configuration (CLDC), and the mobile information device profile (MIDP).

vendors of Java-enabled mobile devices to find people to write their software.

MAKING JAVA WORK IN HANDHELDS

Sun, which designed and manages development of Java, is in the forefront of making the technology work in handheld devices. However, other vendors have also become active in this area.

Sun Microsystems

Sun and a group of partners created J2ME to make Java work on smaller devices. J2ME includes some core Java instructions and APIs but runs more easily on small devices because it has a smaller footprint than the Java 2 Platform Standard Edition (J2SE) or Enterprise Edition (J2EE), shown in Figure 1, and has only those features relevant for the targeted devices. For example, J2ME's graphics and database-access capabilities are less sophisticated.

J2ME generally incorporates the connected limited device configuration (CLDC), which is implemented on top of operating systems and serves as an interface between the OS and Java-based applications. The CLDC generally uses the K virtual machine (KVM), a slimmed-down, less-functional version of the Java virtual machine (JVM) for small devices. The J2ME mobile information device profile (MIDP) sits on top of the CLDC and provides a set of APIs that define how mobile phones will interface with applications.

Other vendors

Several vendors besides Sun are creating Java-based technologies for handheld devices. Hewlett-Packard makes the MicroChaiVM (http://www.hp.com/products1/embedded/products/devtools/microchai_vm.html), a cloned JVM that doesn't have Sun's licensing fees and usage restrictions. Several vendors, including Ericsson and HP, plan to use MicroChaiVM-based phones.

Other approaches help Java technologies designed for larger computers work on mobile devices.

For example, SavaJe developed the SavaJe OS, which supports Java applications in a mobile environment by optimizing J2SE libraries for common mobile CPUs. Mathew Catino, SavaJe's cofounder and vice president of marketing, said Java applications typically spend 80 to 90 percent of their time executing the libraries. Therefore, he explained, optimizing the libraries enables applications to run 10 to 20 times faster.

Zeosoft has developed ZeoSphere Developer, which permits the creation of mobile applications that support Enterprise Java Beans, Sun's Java-based software-component architecture. This could simplify the development of complex enterprise applications that communicate and run across servers (via J2EE), PCs (via J2SE), and mobile devices (via J2ME).

Software development tools

Application developers can use existing tools to create Java programs for handheld devices by limiting their code to libraries and APIs supported by J2ME.

However, J2ME includes only a limited number of development libraries, noted Jacob Christfort, chief technology officer of Oracle's Mobile Division.

Also, said Gartner's Jones, enterprises might shy away from J2ME because of the poor user interface designed for small device screens, the primitive threading model, and minimal native data-handling facilities. In essence, he explained, the design approach that lets J2ME work on small devices sometimes makes it inappropriate for large-scale enterprise uses.

To address these concerns, several vendors have released or will soon release development toolkits or toolkit extensions to help developers more easily meet enterprise applications' needs. The new approaches include Sun's Forte for Java Programming Tools, the Oracle 9i Application Server

Wireless architecture toolkit, and the Sprint PCS Wireless Toolkit.

Because of J2ME's shortcomings, Jones said, corporate applications will probably be based on the larger-footprint J2SE as mobile devices get more processing power.

Regardless, said John Montgomery, product manager with Microsoft's .NET Development Group, current Java tools are too primitive and difficult to use for most developers.

Server-side handheld Java

Another Java-enabling approach would link handheld devices to Java applications and services on servers. AT&T Wireless, BEA Systems, IBM, Nokia, NTT DoCoMo, Sun, and other companies have created the Java-based Open Mobile Architecture for linking cellular phones and servers. The project would augment J2EE, designed primarily for servers, so that it would support standards that mobile devices can use with Internet-based information. The standards include XHTML (for displaying Web pages on mobile devices), SyncML (for synchronizing data between mobile devices and other machines), WAP 2.0 (to access Internet content and services), and the multimedia messaging service (for handheld messaging).

IMPLEMENTATION IN HARDWARE AND SOFTWARE

Java technology can be implemented in software or in hardware on either a specialized Java acceleration chip or a core within the main processor.

Software implementations tend to run less efficiently because systems must translate each Java instruction into native instructions that the CPU can run. Separate hardware chips are more efficient but represent additional device components and cost. Java cores integrate some of both approaches.

Software approach

In the software approach, a device's CPU runs the Java code. David Rogers, marketing manager for Intel's PCA Components Group, said his company has developed techniques for speeding up the software process, which used to bog down when the CPU switched from instructions it could accelerate to instructions it couldn't.

In addition, Intel and other software-based Java proponents say the latest mobile processors can run Java fast enough to compete with hardware-based approaches.

Analyst Markus Levy with MicroDesign Resources, a semiconductor-industry research firm, disagreed. He said, "People are spending a lot of energy fine-tuning the software-based approaches. For some people that may be good enough, but if you really want the most efficient implementation you need a hardware-based approach."

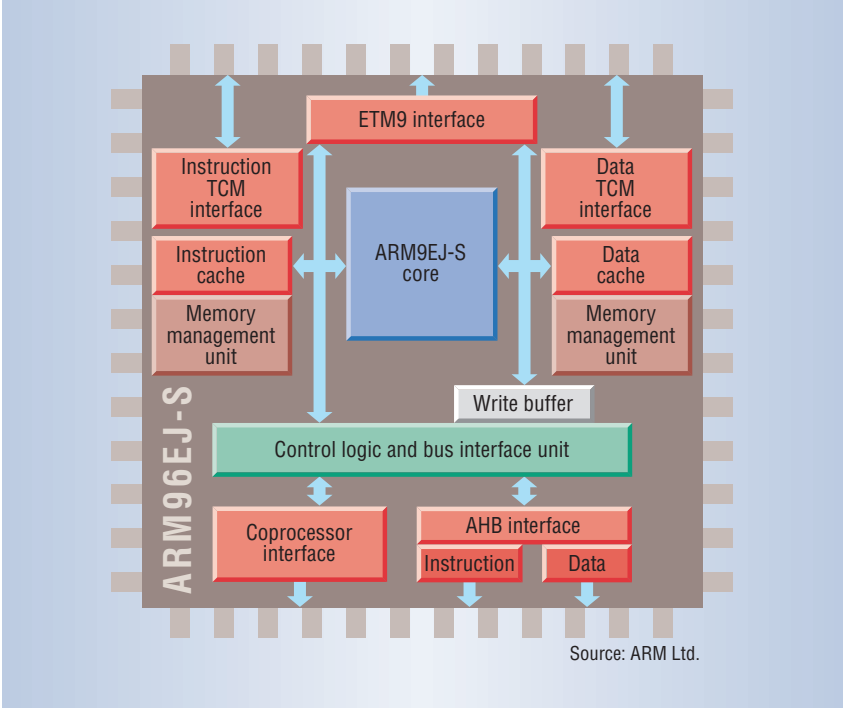


Figure 2. ARM Ltd.'s ARM926EJ-S chip includes the company's Jazelle technology in its ARM9EJ-S Java-enabled processor core. In addition, the chip includes separate ETM (embedded trace macrocell), data TCM (tightly coupled memory), and AHB (advanced high-performance bus) interfaces.

Java hardware

Companies such as ARC Cores, ARM Ltd., Aurora VLSI, Digital Communications Technologies, inSilicon, and Zucotto Wireless are developing hardware that runs Java, either as Java coprocessing cores for integration into CPUs or as stand-alone Java chips.

Both hardware-based approaches promise to increase Java-based application performance and, by running more efficiently, reduce power demands on battery-dependent cellular phones.

Different companies' chips execute different subsets of the Java instructions. For example, ARM's Jazelle chip, shown in Figure 2, executes about 68.2 percent of all possible Java instructions, while Aurora's DeCaf runs about 95 percent. Running a bigger set of Java instructions provides more functionality but makes a chip cost more and consume more power.

Joan Pendleton, Aurora's cofounder and chief architect, said there are two classes of acceleration. The first, used by most vendors, translates Java byte-

code into native processor instructions. The second directly executes Java bytecode, which offers better performance but requires a larger footprint because of the additional circuitry necessary to run the software in hardware.

Levy predicted that Java cores will be more popular than stand-alone Java processors. This approach's primary constraint is that developers must use a system-on-chip approach to create their products. Putting multiple functions on a chip is more expensive to develop, but the elimination of additional chips reduces device costs. Stand-alone Java chips are less expensive to design but lead to higher device costs.

CONCERNS AND CHALLENGES

Mobile Java is still a relatively new technology. Many industry watchers say the technology has kinks that still need to be worked out.

For example, Gartner's Jones expressed concern about vendors' differing Java implementations. He said some developers are complaining about having to manually optimize their Java games for different cellular phones.

And although there are many Java developers, there are fewer who have experience working with J2ME and writing code for small, resource-constrained devices.

Overall, said Microsoft's Montgomery, "J2ME is an interesting set of engineering compromises, but I would argue exactly the wrong set of compromises. It is too big for the smallest devices but too small to have the features you want on the smartest devices."

Performance

Jones said that mobile Java can be somewhat slow because the KVM is not particularly fast. However, he added, the KVM should become faster in the future, particularly as phones with more memory can run just-in-time compiler technology, which enhances performance. "In five years," he said, "[performance] will be a nonissue."

Another problem, said Levy, is a lack of standards to objectively measure performance across platforms. Levy has thus launched a Java-processor group within the Embedded Microprocessor Benchmark Consortium (http://www.eembc.org/). The group expects to release its first benchmark by next month.

Not enough applications

There are currently some mobile-Java applications, including games and weather and traffic maps. However, Jones said, there are not enough desirable mobile-Java applications yet. The reason is not the technology, he said, but instead the lack of an effective business model and a commercial infrastructure that would enable developers to profit from their work.

Industry observers say mobile Java still has kinks that must be worked out.

The growth of publishing intermediaries that would certify and sell mobile-Java software may eliminate this problem.

HANDHELDS AND THE FUTURE OF JAVA

Jones said Java is doing well on back-end servers because Java-based applications can easily be redeployed as companies buy new servers. However, he noted, client-side Java use has faded considerably because many enterprise-application developers turned to Visual Basic to work within the corporate environment, which is typically Microsoft-based. Thus, the battle for the mobile platform is important to Sun.

However, Sun's Java initiatives for cellular phones are facing stiff competition from various sources, including Microsoft's wireless efforts, the Symbian operating system, Linux, and Qualcomm's binary runtime environment for wireless (http://www.qualcomm.com/brew/).

"We are still in a phase of market confusion and have not yet gotten to a state of market consolidation," Jones explained.

According to Jones, J2ME will attract more application developers as it becomes a richer and less constrained environment. A survey by Evans Data, a market research firm, found that wireless developers who have used Java expect to use the technology a bit more in 2003 than they will this year.

Java will also become even more attractive as smart phones get more processing power and vendors design better APIs for color screens, higher quality sound, intellectual-property protection, and user-location capabilities, he added. However, he cautioned, these extra features would give vendors more opportunity to create their own Java implementations, which could fragment the application-development environment.

Sprint PCS's Wang said the initial focus of mobile Java will be on games, multimedia, and ring tones. Over time, Levy added, Java will become a de facto standard built into smart phones.

SavaJe's Catino predicted that Microsoft and Java-based technologies are likely to coexist in phones during the coming years. Third-party vendors could help this process by developing software-integration techniques that would combine the two environments in devices. ■

George Lawton is a freelance technology writer based in Brisbane, California. Contact him at glawton@glawton.com.

Editor: Lee Garber, Computer, 10662 Los Vaqueros Circle, PO Box 3014, Los Alamitos, CA 90720-1314; l.garber@computer.org