The Wireless Explosion

A new generation of wireless devices that include cell phones, pagers, PDAs, and other consumer devices are rapidly entering the market, and analysts are predicting that the demand for applications is set to explode. Millions of US users will subscribe to wireless services in the coming years, and Europe is moving even faster. As applications developers stop to consider the new opportunities that this growth will create, they are also asking fundamental questions about the available tools and programming environments they will need.

Peer-to-peer (P2P) computing readily lends itself to the more dynamic environments of wireless devices. Characterized by their ability to create, join, and interact with peer groups, and to post advertisements to offer and solicit resources, P2P applications can dynamically find what they need — an approach more suited to the wireless “lifestyle.”

Project JXTA for J2ME™ is the response of the JXTA developer community to this need. Designed to offer a rich programming environment for creating applications using the P2P computing model, JXTA for J2ME technology promises to accelerate development for those wishing to participate in the wireless applications marketplace.

The Challenges of Resource-Limited Platforms

Perhaps the most defining characteristic of wireless devices is the limited set of resources they have. While time and technology will most certainly help resolve this problem, the resource constraints for today’s wireless devices are daunting:

- Persistent storage is limited and shared by all applications
- Runtime heap is small
- Network bandwidth is very limited and latency is high
- Processor performance is modest
- Electrical power is confined to that available from small batteries most of the time

Clearly, these constraints limit the kinds of applications that wireless devices can host independently. Because they do have the ability to communicate with other systems, however, their greatest potential lies in their ability to act as portals into networks where more comprehensive computing and storage resources can be found. Simply put, wireless devices will have to rely on networks to get much of their work done.


A discussion of wireless applications must begin with Java™ technology, which is already delivering on its promise of platform-independent applications for Web environments, small businesses, and large enterprises. Today, thousands of Java applications have been deployed.

Java clearly has a role to play in wireless. The many vendors of mobile devices and the incompatibilities that will inevitably result can be mitigated by the unifying force of the Java programming language – a point long ago recognized by both industry and the developer community.
Recognizing that some tailoring was required to more carefully match Java environments to their target platforms, Sun Microsystems began working with the developer community to regroup its Java technologies into three editions: Standard (J2SE™ technology), Enterprise (J2EE™ technology), and Micro (J2ME technology). Each edition incorporates tools that are carefully matched to their intended use:

- Java virtual machines tailored to fit comfortably inside their target computing platform and APIs that are specialized for each type of environment
- Tools for deployment and device configuration
- Specifications for profiles — the minimum set of APIs needed for a particular environment — and requirements for the Java virtual machine functions required to support those APIs

J2ME technology specifically addresses the vast consumer space, which covers a range of smaller platforms that include smart cards, pagers, set-top boxes, and wireless devices. Like all Java editions, the J2ME platform maintains the qualities critical to Java, including portability of code, safe network delivery, and upwards compatibility with J2SE and J2EE platforms.

J2ME includes special profiles designed to support the limited resources of smaller devices. The Connection Limited Device Configuration (CDLC) and Mobile Information Device Profiles (MIDP), are core class libraries and specialized APIs designed to work in the constrained environments of wireless devices. Despite their small size, CDLC and MIDP offer developers the tools they need to create powerful wireless applications that can interoperate with Java solutions running on workstations, servers, and mainframes:

- CLDC targets small, resource-constrained devices, such as mobile phones, personal digital assistants, and small retail payment terminals. CLDC is suitable for devices with 16/32-bit RISC/CISC microprocessors and controllers with as little as 160 KB of total memory.
- MIDP-1.0 is a set of Java APIs that are targeted at mobile information devices, such as cellular phones and two-way pagers. The MIDP-1.0 specification addresses issues such as user interface, storage persistence, networking, and application model.

Project JXTA – A Breakthrough for Peer-to-Peer Computing

In April of 2001, Sun Microsystems unveiled Project JXTA, a community development project to create a new programming platform designed to solve a number of problems in modern distributed computing, especially in the area broadly referred to as peer-to-peer (P2P) computing.

Project JXTA was conceived with a set of objectives intended to address the shortcomings of the peer-to-peer systems already in existence or under development:

- Interoperability. JXTA technology is designed to enable interconnected peers to easily locate each other, communicate with each other, participate in community-based activities, and offer services to each other seamlessly across different P2P systems and different communities.
- Platform independence. JXTA technology is designed to be independent of programming languages (such as C or the Java programming language), system platforms (such as the Microsoft Windows and UNIX® operating systems), and networking platforms (such as TCP/IP or Bluetooth).
- Ubiquity. JXTA technology is designed to be implemented on any device with a digital heartbeat, including sensors, consumer electronics, PDAs, appliances, network routers, desktop computers, data-center servers, and storage systems.

Project JXTA began with a small group of engineers at Sun Microsystems, along with the participation of a small but growing number of experts from academic institutions and industry. JXTA technology is now distributed under an open-source license, and as such, is being co-developed by a larger community of users interested in P2P computing.
Project JXTA for J2ME Adds P2P Support for Wireless Devices

The Java and JXTA Developer community, understanding that peer-to-peer technology could have important consequences for wireless computing, set about defining a set of requirements for creating wireless JXTA peers built on the foundation created by J2ME and MIDP. They began with a set of simple goals:

- Be interoperable with JXTA on desktops and workstations
- Provide a P2P infrastructure for small devices
- Be simple and easy to use by developers
- Be small enough to be used with cellular phones and PDAs
- Allow the creation of applications that provide a good user experience
- Be MIDP-1.0 compliant

The result of this effort, incubated by Sun Microsystems and then quickly transferred to the JXTA development community, is Project JXTA for J2ME. JXTA for J2ME will allow a MIDP-compliant device to participate in P2P activities with JXTA peers running on larger platforms.

MIDP-1.0 provides the needed APIs that are already recognized by the Java development community, and thus acts as a firm foundation for the creation of wireless JXTA peers. Despite this, MIDP-1.0 does have constraints such as limited libraries, lack of an XML parser, support for outgoing HTTP only, and no security support, and so these limitations are reflected in JXTA for J2ME’s capabilities.

JXTA Relays

Because of platform resource constraints, and the corresponding limitations in MIDP-1.0, JXTA for J2ME peers can only act as edge peers. That is, they cannot assume the role of more sophisticated peers that offer services to other members in the peer group. From a purely practical point of view, wireless peers must principally act as windows into the network for their users. As a result, the heavier lifting (searching for resources, performing computationally-intensive tasks, etc.) must be done by other members of the peer group. Indeed, even some of the basic tasks associated with peer group membership must be assumed by other peers called JXTA relays (see figure).

JXTA relays are proxies for small devices, taking care of most of the tasks required to maintain their status as a full-fledged member of a peer group:

- Provide interoperability with JXTA protocols
- Act on behalf of JXTA for J2ME peers as a proxy to perform user, group, and peer discovery; to create pipes and groups; to join groups; and to communicate.
- Filter JXTA traffic
- Trim and optimize advertisements

The role of the wireless peer and JXTA relay becomes more clear with a simple example of how a fundamental JXTA operation — like discovery — works. A wireless peer sends a discovery query to a JXTA relay. The relay then propagates it to the JXTA network on behalf of the wireless peer. All of the responses are collected by the relay, which trims them to optimize transfer efficiency to the low-bandwidth wireless device. Meanwhile, the wireless peer periodically polls the relay for all the in coming responses.

Figure 1: Wireless devices that are members of a JXTA network connect through JXTA relays, and can communicate independently of underlying network protocols and network carriers.

JXTA relays are proxies for small devices, taking care of most of the tasks required to maintain their status as a full-fledged member of a peer group:

- Provide interoperability with JXTA protocols
- Act on behalf of JXTA for J2ME peers as a proxy to perform user, group, and peer discovery; to create pipes and groups; to join groups; and to communicate.
- Filter JXTA traffic
- Trim and optimize advertisements

The role of the wireless peer and JXTA relay becomes more clear with a simple example of how a fundamental JXTA operation — like discovery — works. A wireless peer sends a discovery query to a JXTA relay. The relay then propagates it to the JXTA network on behalf of the wireless peer. All of the responses are collected by the relay, which trims them to optimize transfer efficiency to the low-bandwidth wireless device. Meanwhile, the wireless peer periodically polls the relay for all the in coming responses.
Relays are not a new idea for JXTA. Relays are already used to access JXTA peers behind firewalls and are the means by which networks using NAT allow peers to communicate with the outside world.

**JXTA Relays Do Not Compromise the P2P Vision**

Unlike the client-server model, wireless JXTA peers are not required to establish and maintain a static relationship with designated JXTA relays. Two independent JXTA for J2ME peers can be connected to different relays and still discover each other and communicate. In addition, a wireless peer can dynamically change its relay or have multiple relays. In the future, a wireless peer will be able to search for JXTA relays and then configure one of them to be its default. The relays will provide a service to the wireless peer, just as they might offer database search and retrieval or computational services.

**Optimized Protocols Reduce Traffic Volume**

The JXTA relay filters all the incoming queries from the network and responds to the queries on behalf of wireless peer. In addition, for each JXTA core protocol, like discovery, the JXTA relay trims incoming responses to the bare minimum. For instance, if a wireless peer initiates peer discovery, an incoming response is a Peer Advertisement. Instead of sending the entire advertisement to the wireless peer, the relay sends only the peer ID, reducing the number of Elements in the advertisement to one.

**Simple API**

The API offered to applications by JXTA for J2ME is designed to hide complexity for the casual developer, while allowing low-level access for the advanced programmer. The API consists of just three classes:

- **Message.** Methods to create and manipulate JXTA messages.
- **Element.** Methods to construct and manipulate the basic components of JXTA messages.
- **PeerNetwork.** Operations that can be invoked on the JXTA network.

The API avoids the use of Interfaces, Factories, Listeners, Threads, and Inner Classes to reduce its size. Having an API represented in just three classes allows new programmers to start writing applications for JXTA for J2ME very quickly. This low conceptual weight reduces the barriers to entry.

Specifically, the following features are scheduled to be supported in the first release of JXTA for J2ME. (The item in parenthesis following the feature are the **PeerNetwork** methods used to implement the feature.):

- **User Discovery (pipes).** Each JXTA for J2ME application will be able to maintain and search for a limited list of users for that application — a buddy list on a per-application basis. (**search**)
- **Group Discovery.** Each application running on JXTA for J2ME will be able to discover a JXTA group and join it. (**search**)
- **Peer Discovery.** JXTA for J2ME peers will be able to discover other peers. (**search**)
- **Create Pipes.** Create both point-to-point and propagate pipes. (**create**)
- **Create Groups.** Create groups to limit the scope of discovery. (**create**)
- **Join Groups.** Join a given group according to the JXTA specification. (**listen**)
- **Communicate.** Exchange messages with other peers. (**send, poll**)

**The Promise of Small Device Peer-to-Peer Computing**

As mentioned earlier, the peer to peer model is especially appropriate to the characteristics of wireless devices. Their mobility implicitly requires a method of discovering network resources — a key feature of P2P networks. Likewise, their limited resources means a greater reliance on network services in order to create useful applications — again a central characteristic of P2P computing.

As a result of the efforts of Project JXTA for J2ME, wireless applications can be designed now using P2P computing technology. The opportunities for this technology seem especially strong in the areas of gaming, financial services, and instant messaging, although ultimately, the possibilities will be limited only by the imaginations of developers.

**Additional Resources**

For more information about Project JXTA, visit their web site at [http://www.jxta.org](http://www.jxta.org). There you will find a number of resources including documentation, JXTA source code, sample applications, and technical discussions.

For more information about Project JXTA for J2ME visit their page at [http://jxme.jxta.org](http://jxme.jxta.org).

Sun Microsystems, Inc.  Page 4  March, 2002