W H I T E P A P E R

Jini™

What is Jini?

S U M M A R Y

Jini is a Sun R&D project inspired by Bill Joy that dramatically expands the power of Java technology. Jini technology enables spontaneous networking of a wide variety of hardware and software -- anything that can be connected to a network.

Jini allows people to use networked devices and services as simply as using a phone today – plug-and-participate via a network dialtone. The goal of Jini is to dramatically simplify interaction with networks.

Jini takes advantage of Java technology. Jini consists of a small amount of Java code in class library form and some conventions to create a "federation" of Java virtual machines on the network, similar to the creation of a community today. Network citizens such as people, devices, data and applications within this federation are dynamically connected to share information and perform tasks.

MAJOR TRENDS -- NETWORKS EVERYWHERE

The world is getting networked. Today, for example, a network is literally a requirement for a business to be successful. Business networks are expanding to include direct interaction with suppliers and customers. Interacting with wireless networks is becoming almost commonplace.

Businesses and consumers are demanding more interaction with the network. The business traveler wants to arrive at a hotel, plug in, and not only interact with his or her work environment back in the office, but also interact with the local hotel services such as the printer or fax machine. Using a cellular phone or laptop computer, a parent may want to access a camera in their home, just to see how things are going. People will want to move, connect and utilize immediately local and customized services.

In the near future, we'll see Networks proliferating into many other environments. For instance, there will be networks in the home, connecting audio/visual equipment such as televisions and stereo equipment to home office computers and peripherals to control networks such as security surveillance and temperature control thermostats. High bandwidth mediums such as Cable and ASDL will transport entirely new services into the home.

Networks will reach into the car, as service providers provide the driver more and more services while traveling. In addition to navigation systems, proximity services such as points of interest and listings for local restaurants will emerge on the driver's screen. Car maintenance will be automated as the car keeps in touch with remote diagnostic equipment and will inform the driver when a problem with the car is emerging.

BUSINESS OPPORTUNITY - NETWORK SERVICES

The business opportunity that Jini enables is a new class of Network Services. For instance, product manufacturers will offer new services on network enabled products. For example, a disk can be viewed as a storage service to the network, offering storage, automatic backup to a tape and other brand new services. A networked camera might offer new types of imaging services such as security. These new services enable the manufacturer to become a new type of Service Provider to the network.

Jini also enables traditional Service Providers to offer new classes of services. For example, a media service provider may want to provide newspaper printing service onto a consumer's home printer. A wireless provider may want to offer proximity services via the cellular phone.

Jini also simplifies the management of existing services. In the case of overnight delivery, Jini simplifies how the distributed workforce can easily connect into the network. In a retail bank, the Jini-enabled computers and peripherals simplify the system administration of the branch. For a wireless provider, Jini enables a cellular phone to announce the phone's capabilities to the network such as screen size and processing power, allowing delivered services to be tailored to the phone.

Of course, Java software developers will develop new applications and services to take advantage of all these networks and new types of products and services.

THE PROBLEM IS ...

The problem is that in today's environment, networking is much too complex. For instance, attaching a PC to the network and accessing a networked printer is far too complicated. Only an experienced System Administrator can deal with the complexity of loading drivers, figuring out the configuration files, etc... Clearly, we can not expect the average consumer to manage today's style of network.

Networks today are also too brittle and inflexible. A slight change in a network can cause havoc that can be impossible to fix.

Adding capacity to the network, such as disk storage, is also complicated. To add a disk drive, for instance, typically one must open a chassis, deal with setting jumpers and navigate through a maze of confusing setup questions. Even experts have problems with this.

In fact, from the consumer's perspective, what is needed is a simple way to plug in hardware and software into a networked environment and then use the available services immediately: like plugging in a phone today. Today, when a consumer buys a phone from the store, he or she does not need to configure the phone. The consumer calls the telephone Service Provider and a service is made available. Later, the consumer plugs in the phone and accesses the phone service. Spontaneous networking.

THE VALUE OF JINI

Jini is about simplifying interactions with networks.

From the consumer's perspective, the consumer plugs into the network attachable devices and software as simply as plugging in a phone today.

From the traditional Service Provider's perspective, Jini simplifies the management of Services Delivery. Devices announce not only value added services, but also attributes and capabilities of the device to the network. Service providers can now tailor services to the device. Of course, Jini holds the promise to open a floodgate of new Networked Services as well.

From a product manufacturer's perspective, Jini opens whole new markets. Because Jini simplifies the ability for devices to announce value added services to the network, products can compete not as commodity items, but rather as products differentiated by value added services.

From the Java Programmer's perspective, Jini simplifies the task of writing distributed applications to the point where any Java programmer can write applications and services to take advantage of the new Jini-enabled devices. Therefore, instead of employing limited expert resources to write distributed application, any Java programmer can develop services for a Jini enabled network.

JINI ORIGINS

Pre-1994, Bill Joy presented a proposal to Sun Labs where he presented three main concepts:

1) a language that would run on all platforms,

2) a virtual machine to run this language, and

3) a networked system to allow the distributed virtual machines to work as a singular system.

In 1995, the language and virtual machine were introduced to the market as the Java Programming language and Java Virtual Machine. The system context, however, was kept in the Sun R&D for continued research and development. This system context is Jini.

JINI DEPLOYMENT STRATEGY AND PARTNERS

Sun deployed a broad strategy to provide Jini technology into the market. Basically, Jini is relevant to any company that provides products and/or services to a networked environment. This list includes traditional device manufacturers and service providers to software developers.

HOW WILL JINI BE LICENSED?

To drive innovation for the vision of Jini and to gain rapid acceptance in the market the Jini source code will be open to the developer community, similar to Netscape's Mozilla model. To guarantee compatibility and quality a mark for commercial products is being considered. The specific details around the licensing model are still being finalized. A draft of the proposed license will be available in August.

JINI TECHNOLOGY OVERVIEW

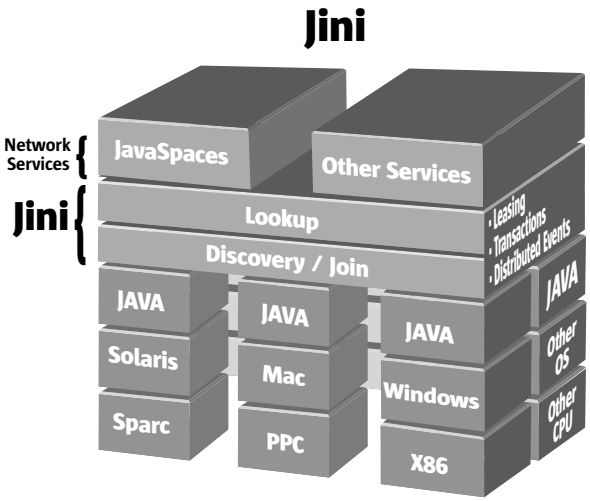
The Jini technology can be separated into two categories: Infrastructure, and Distributed Programming. In addition, Network Services will be provided to run on top of Jini.

INFRASTRUCTURE

Jini Infrastructure addresses the fundamental issue of how devices and software connect to and register with the network.

The first Infrastructure element is called *Discovery* and *Join*. Discovery and Join solves the difficult problem of how a device or application registers itself with the network for the first time with no prior knowledge of the network.

The second Infrastructure element is *Lookup*. Lookup can be thought of as a bulletin board for all services on the network.



DISCOVERY AND JOIN

The first task that a device or application needs to do once plugged into the network is to discover the network and have the network discover the device or application. We use the word *Discovery* and *Join* because the device or application can not know anything about the network in advance.

Discovery works as follows:

1. When a Jini enabled device is plugged into the network, it drops a 512 byte multicast Discovery packet onto the network on a well-known port. Among other information, this packet contains a reference back to itself.
2. The Jini lookup listens on the well-known port. When the Discovery packet is received, Lookup uses the device's interface to pass an interface to itself back to the plugged in device or application.

The device or application has now discovered the network and is ready to upload all its characteristics to the Jini Lookup. The aspect of uploading characteristics is the Join aspect of Discovery and Join.

The device or application now uses the interface received to the Lookup during the Discovery phase to Join the network. Characteristics uploaded into Lookup include all value-added services provided by the device or application, such as drivers, help wizards, attributes, etc...

LOOKUP

Lookup is the Network bulletin board for all services on the network. Lookup stores not only pointers to the services on the network, but also the code and/or code pointers for these services.

As an example, when a Printer registers with the Lookup, it will load the Printer Driver or an Interface to the driver into the Lookup. When a client wants to use the printer, the driver and driver interface get downloaded from the Lookup unto the client. In this way, drivers do not need to be loaded in advance on the client.

The printer might also load other value-added services into the Lookup. For instance, the printer might store attributes about itself, like if it supports postscript, or if it's a color printer. The printer might also store help wizards that will run on the client.

If a Lookup is not available on the network, then a Peer Lookup will be used. A Peer Lookup can be enabled when a client desiring a service cannot find a Lookup on a network. In such situations, the client can send out the same Discovery and Join packet used by a lookup to request that any service providers register. Service providers will then attempt to register with the client as though it were a Lookup.

DISTRIBUTED PROGRAMMING

Jini Distributed Programming adds to Java additional functionality required for building distributed systems. Specifically, Jini Distributed Programming provides leasing, distributed transactions, and distributed events.

LEASING

Leasing is analogous to leasing an apartment. When one wants to lease an apartment, they negotiate a certain amount of time to use the apartment. Likewise, in Jini, objects negotiate leases with each other. For example, when a device uses the Discovery and Join protocol to discovery the network, it registers for a certain leased time. Before the lease expires, the device must re-negotiate the lease. In this way, if the device is unplugged, once the lease expires, the device's entry in the Lookup will be removed automatically. This is how distributed garbage collection is done.

DISTRIBUTED EVENTS

In a single computer, events are guaranteed to be received by the receiving party and the sequence is guaranteed to occur in order.

However, in a distributed environment, distributed events might be received either out of order, or an event might get lost.

To facilitate distributed events in a Java environment, Jini provides a simple Java API to facilitate distributed events. For instance, when a distributed event occurs, the event carries an event number and sequence. Using this information, the receiving party can check if an event was lost (a sequence number is missing) or if an event was received out of order (the sequence number is out of order).

DISTRIBUTED TRANSACTIONS

In a distributed Java environment, what is sometimes required is a very lightweight way to insure that all events occur in a transaction before the entire transaction is actually committed (also called two phase commit).

To facilitate this style of distributed computing, Jini provides a simple Java API. This API enables an object to start up a transaction manager that will manage the transaction. Every object that will participate in the transaction registers with the transaction manager.

As a transaction occurs, if one of the participating object says that an event in the transaction did not occur, then this information is communicated back to the transaction manager. The transaction manager then tells all the participating objects to roll back to the last well known state. Likewise, if all objects complete their part of the transaction, then the entire transaction is rolled forward.

NETWORK SERVICES ON JINI

On top of the Jini Infrastructure and Distributed Programming, Network Services will be provided to facilitate distributed computing. An example Network Service is JavaSpaces.

For more information on JavaSpaces, see
http://java.sun.com/products/javaspaces

In the future, we expect many other Network Services to be built on top of Jini.

For more information on Jini technology, see
http://java.sun.com/products/jini