Why Jini Now?

THE NETWORK IS THE COMPUTER™

Sun Microsystems, Inc.
2550 Garcia Avenue
Mountain View, CA 94043 USA
415 960-1300 fax 415 969-9131

Revision 01, August 1998

Please
Recycle

Why Jini Now?

We've all seen the picture: A computer room circa 1963 with its raised floor, a dozen tall skinny boxes representing the mainframe, a Teletype terminal, a bulky line printer or two, and the pièce de résistance—a cleancut gentleman loading a mag tape onto one of the 3 or 4 tape drives. He looks out of the picture toward you, smiling.

The picture seems quaint to us now—how unlike contemporary computing.

But in the intervening 35 years, what has changed in the picture? Not much—just the scale. The speed of computation has increased by a perhaps factor of 3000, the size of feasible computations has also increased dramatically by a factor of perhaps 1000, and the size of the computer has decreased dramatically: The contemporary computer—the one that's 1000–3000 times more powerful than the one in the picture—sits on a desk and is used routinely in the office and at home. But a block diagram of the major components, their roles, and how they work together hasn't changed at all. We've merely shrunk the computer and sped it up, but we haven't advanced it as a tool. This is significant.

Other significant things have changed as well—the smiling cleancut gentleman has disappeared, sort of. That man was the *system administrator* (*sys admin*) who was responsible for making sure that the gargantuan computer worked properly, that programs it needed to operate were kept up to date and in good working order, and that the programs you wanted to run were loaded and executed as you specified. The problem is that he hasn't disappeared entirely: If you work in a small office, have a home office or a moderately sophisticated home computer system, or if you work in a department with limited resources, most of his duties have fallen on your shoulders. You make sure the computer is working properly; you keep the programs you use loaded and up to date; and you make sure that all the components are in working order, and communicating properly with each other. In larger organizations, the smiling man is still called a *sys admin*, but his job (or her job) is to install your computer when it first arrives and to fix it or arrange to have it fixed when it breaks down—and you share him or her with 20 or more other people.

What else has changed? Computers have become ubiquitous, and they have disappeared. They've disappeared by donning coats of automobile hide, telephone skin, stereo fur, and microwave scales. Quite a few of the things we use have a computer in it, and all of these computers are easy to use, while the only thing we think of as a computer is hardest to use of all. There is nothing magical or paradoxical about this: Throughout history the best technology has always sunk below our view within tools and toys that do something we need or something we want.

By disappearing they dominate our lives. But also their ubiquity is quite visible on the Internet and the Web. With Web browsers we visit computers all over the world—sometimes in peoples' dens and bedrooms, sometimes in corporate halls the other side of sunset. When we work the Web, aren't we operating under the model that there is exactly *one* computer in the world?

And when we connect to a site with wonderful animated graphics, we've actually installed and successfully run a program that a Swede or an Aussie but someone far away has put together in her or his spare time. It's a simple piece of code written in Java, but what we've done was once considered remarkable. And even today, the simplest program we can buy from the best professional software developer can take many minutes or hours of devoted attention to install and run.

These three facts (you are the new sys admin, computers are nowhere, the one computer is everywhere) should combine to improve the world of using computers as computers— by making the boundaries of computers disappear, by making the computer be everywhere, and by making the details of working with the computer as simple as plugging a DVD machine into your stereo system.

What is Jini?

Jini is a new system architecture that tries to make the improvements these observations suggest: Jini brings to the network the facilities of distributed computing, network-based services, seamless expansion, reliable smart devices, and ease of administration.

Here's the Jini vision: When you walk up to an interaction device that is part of a Jini system, all of the *services* on the Jini system are as available to you as if they were on your own computer—and services include not only software but hardware devices as well, including disk drives, DVD players, VCRs, printers, scanners, digital cameras, and almost anything else you could imagine that passes "information" (digital streams) in and out. Adding a new device to a Jini system is simply plugging it in.

We use the term *system* for a couple of reasons. First, we say Jini is a system because Jini really isn't a computer nor is it a network of computers. Not only does it accept other sorts of devices (entertainment devices, MIDI devices, etc.) than we normally expect on a computer or computer network, but there is no single computer that is the computer. The system appears as a set of services that are available—some are software implemented and others hardware, but the interfaces (user and programmatic) always presents simply and uniformly the services available regardless of how they are implemented or where they are.

The second reason we say Jini is a system is because Jini itself is not simply a set of point technologies but a new architecture for computing, sort of like a new block diagram for what a computer could be.

And we use the term *service* because there is nothing inherently interesting about computers to people who want to use them, just as there is nothing inherently interesting about stereo components except what they do, what services they provide. A CD player, an amplifier, some speakers, and a CD make music—that's all we care about once it's all working. What most people care about from their computing systems is what they can do with them: read and compose e-mail, prepare papers and presentations, do financial work, find information on the Web, or be entertained. Each of these things is a service, and we care about them, not about the types of CPUs, the number of registers, the speeds of the buses, or the word size.

In other words, Jini is a system architecture (hardware, software, and network) that supports the notion that a computing environment is a network-connected set of computing, storage, display, entertainment, and IO devices. In Jini, devices can be added or subtracted, and doing so may alter some of the capabilities of the system, but it will not alter its identity or basic usability.

Putting together Jini requires a few things:

- an infrastructure which operates as a dynamically distributed system
- a common language and implementation that enables low-overhead communication between distributed objects
- a lookup service (which identifies objects that supply those services)
- add-in and subtract-out protocols which are implemented on each device—we call this the *discovery* protocol

Jini Software Details

The first realization of Jini ties together machines on which Java objects are running, perhaps in different virtual machines. Jini enables such objects to work together as though they were on a single, very powerful computer. Jini enables such objects to

be activated and tracked, to communicate with each other, and generally to be managed. Jini provides a user with access to all of the facilities on the collection of networked machines in a location-transparent fashion.

Jini Environment

Jini works in a networked environment, and we assume the following hardware trends will continue:

- processors will continue to get faster, smaller, and cheaper. This trend will continue and permit us to assume that a powerful microprocessor will be available in even the simplest device.
- memory will continue to get faster, larger, and cheaper. This trend will allow reasonable memory to accompany the powerful processors mentioned above.
- network bandwidth will continue to expand. In particular, Jini assumes that networks of between 100 megabits per second and 1 gigabit per second will be common soon.
- storage devices will continue to increase in capacity per unit cost.
- entertainment devices will continue to increase their "digital component" so that Jini systems will increasingly blur the distinction between consumer electronics, computers, networks, the Web, and entertainment devices.

In designing Jini, we have assumed the following about trends in the software environment:

- Java will be a major implementation language, but will not be the only implementation language in use for either new or maintained (legacy) code
- legacy applications— written in a variety of languages and requiring a variety of underlying operating systems—will still be important
- the Web will increase in importance for application and entertainment development (and not just the existing Web, but the Web and Web browsers as a paradigm for human-computer-entertainment interaction)

Jini Functionality

Jini provides resources in which to run Java objects, communication facilities between those objects, and the ability to find and exploit resources on the network.

Jini uses separate Java virtual machines to run Java objects. Running objects in separate virtual machines has two major benefits:

- objects have a degree of isolation not possible within a particular virtual machine
- objects can have different access controls and resource allocation

Jini provides a communication layer between objects in the various virtual machines that enables those objects to work together. Using an enhanced version of the Java Remote Method Invocation system (RMI), Jini enables method invocations to take place across virtual machine and physical machine boundaries. RMI enhancements enable activation of objects being called and the use of multicast to contact replicated objects, providing high availability and high reliance objects to be easily implemented in the Jini framework.

Jini uses lookup service allowing resources connected by the communication infrastructure to be found.

Jini provides a mechanism for other Jini-enabled devices (such as disk drives, printers, and computers) to join into the overall Jini system. When a device joins a Jini system, its services are added to the lookup service. Symmetrically, when a Jini-enabled device leaves a Jini system (by being removed or by becoming unreliable), its services are deleted from the lookup service. The mechanism fpr joining is called *discovery.*

Jini Benefits

By providing a well-established distributed computing platform which takes advantage Java virtual machines running on a variety of platforms, users will see performance and reliability gains from applications designed to use Jini. Resources already provided in the Java language will see performance and reliability gains by being activated on unencumbered machines where possible.

Jini provides the possibility to *compose* systems to meet specific requirements rather than relying on a general-purpose system. Particular service requirements for a task or a group can be put together because Jini provides a low-impact way of customizing not only your software but your hardware configuration. The services view of work enables devices and software to be managed uniformly.

Equally important, though, is that managing resources available on a Jini system is much simpler: Each Jini-enabled device has enough information stored on it to enable hot-plugging capabilities. By simply plugging in a device, all of its Jini-related resources become available without intervention. Each device contains a user interface for configuring and customizing the device.

This implies that the total cost of ownership of a computer system will decline as fewer system administrators are needed. To be sure, in the short term a system administrator will still be required for even a medium-sized network, but the tasks that he or she will be performing will be those associated with the enterprise in which the network is involved, not routine resource maintenance. Further, systems for small business or departments and home use can be administered more reliably and at a much lower cost with Jini.

Finally, Jini begins to bring together the realms of computing and home networks including entertainment and personal/family management. Jini devices can include not only computers, printers, scanners, and disks, but VCRs, DVD players, CD players, MIDI devices, the Web, and even broadcast receivers. As long as a device can attach to the network and can appear as a Java object, it can be a Jini device.

Changing Trends

Jini is based on a number of already-existing technologies, and so it represents a small technological step but a large intellectual one. Video recorders were once considered useful only to television studios and filmmakers. But, in the 1970s people found that VCRs provided a way to alter how they used TVs, how they interacted with their families, and how they entertained themselves. For better or worse, the modern world will never be the same. What started out as a specialized, exotic device turned into one of the most common pieces of home technology. Similarly, computers and networks, until now the domain of specialists and professionals, can make the leap into the world of ordinary life where they can finally make the difference so many in the past have imagined.

In this relatively technical section we will look at the technologies that are converging to make Jini possible.

Network Models

From their first appearance in the 1950s until today, computers have been expanding in connectivity. From the 1950s until the early 1970s there were almost exclusively isolated mainframes; from the early 1970s until the early to mid-1980s there were additionally minicomputers sometimes connected by proprietary networks or the ARPANET; from the early 1980s until the later 1980s and early 1990s there were additionally workstation servers and PC clients connected by LANs; from the early 1990s until now there were workstation servers and thin clients connected by LANs, WANs, and the Internet; and right now we are seeing the concept of servers and clients disappearing and the network actually becoming the computer.

Each of these transitions has been fueled by a couple of forces: price, control, data transfer speed, and the role of the network.

Mainframe → Minicomputer

For many years the mainframe was the center of business computing: Payroll, record-keeping, inventory, and accounting were all handled by a large central computer.

Control of the mainframe was in the hands of a centralized computing agency—the MIS department—which did all the backups, custom programming, upgrades, etc. The price of the machines was very high, which implied they were used for "important", noninteractive tasks.

In the late 1960s and early 1970s several manufacturers produced what were termed "minicomputers". The minicomputer's main effect was to bring computing into departments and divisions, where it was once confined to corporate headquarters.

The mini was physically a bit smaller than the mainframes, considerably less expensive, and had interactive capabilities. Corporate data at that time resided partially in the mainframe and partially in the mini, and approximately 20% of the backup and maintenance load moved into the departments. Maintenance of the computers was still performed centrally. The total cost of ownership was still high and perhaps a bit higher than for the pure mainframe world.

The most dramatic effect, though, was to begin the computerization of industry by making computers available at the departmental level with interactive computer terminals and some communication software. E-mail and electronic memos began to change the culture of the workplace to one where the "virtual" workplace began to emerge.

Of significance was the battle that began between the central MIS groups and the departments and individuals that would play out over the next several decades. Early MIS groups enjoyed a monopoly on computing within a company. The mini was the first opportunity for the departments to challenge what they saw as the tyranny of the MIS group. Forward-looking departments saw benefits of computing in their own areas of responsibility outside the administrative role the computer played in the entire enterprise. With the minicomputer departments began to hire their own programming staff to do customized programming and some maintenance.

A significant change that pushed this transition was that many people in the departments had a computer background from their high school and college educations, and the new entrepreneurial spirit of the 1980s meant that department heads more often treated their departments as small business, exercising more and more autonomy.

Mini → PC / Server

In the mid- to late 1980s, several trends came together to weave a change in corporate computing. The first was the ascendance of the personal computer. This took place because the personal computer began to have enough productivity software that departments put one on every desk. Productivity software includes word processing, e-mail, document preparation, accounting, project management, and simple database packages. The trend was accelerated by the appearance of LANs.

The other trend was the reduction of size of what was effectively next evolution of the minicomputer: the workstation. The power of the workstation increased to where it largely replaced the minicomputer, and the purchase price of the workstation approached the PC. Many departments had both workstations and informal computer support groups, and some had their own MIS groups to do custom programming and support.

Workstations ran on the same network as the PC's and had enough horsepower that the dream of the departments could begin to be realized: department-level computing where the business of the department could be codified.

The price of PC's dropped enough that, by the early 1990s, individuals were buying their own PC's for working at home. At first this was supported by transferring files with floppy disks, but advanced to modem-based connectivity once the problems of high-speed data transfer over analog phone lines were handled.

Notice that the source of processing power had shifted for the first time from a centralized CPU center to local machines. All the important productivity software came off the shelf and ran very well on the PCs.

Data transfer speed increased from something close to 56 kilobits per second in the early 1980s for custom or proprietary connections to as much as 10 megabits per second in the early 1990s so that cross-mounted file systems could provide a way to informally share information within and between departments.

The MIS groups hit their low during this transition, because almost every one of their responsibilities had been taken over by departments and individuals while they were still held accountable for integration failures.

PC/Server → Thin Client

The most dramatic change in the computing industry was the switch to the Web from the PC/server style of computing. In the late 1980s, the first implementation of the World Wide Web appeared. From then until Netscape popularized the Web, there had been slow but steady growth in the use of the Web and in the number of Web sites. When browsers became commonplace, businesses that had invested in three-tier and client-server applications realized that the browser could be viewed as a very thin client and the Web server as the server in an application. Java as a mechanism to create usable user interfaces within any browser and CGI scripting on the server side completed the picture.

The possibility of thin clients means that the cost of being connected into a business is very low if requirements are only modest. That is, any computer that can run a Web browser can be used to conduct business and exchange e-mail and documents.

Of importance is the fact that the thinner the client, the more the focus of computing returns to a centralized place, though that place is likely to be a department and not the enterprise. On the other hand, with the speed of data transfer and the existence

of so-called legacy connections, the client is increasingly represented by a Web browser, the server by a workstation, and the data by a mainframe-supported database.

The cost of ownership decreases because the client is not required to run very many programs, and the cost then depends on how many clients are connected to a server and therefore how powerful the server must be or how many servers are required. Nevertheless, because the client is still a PC, its horsepower is overabundant, and so there is waste, causing pressure for a less expensive solution.

This form of computing requires Internet and Intranet connections to be fast and reliable, putting pressure on the servers and infrastructure overall.

Early thin-client systems were simple forms-based HTML interfaces. Soon after the Web began to take off, Java was introduced as a way to write small applications that would run on the client machine after being downloaded to a browser. Today this means that the client machine cannot be trivially simple and poorly powered (as a computational engine), but neither does it need to support general-purpose computing in quite the same way: virtual memory and hence fast disks are probably not necessary, nor is a large amount of disk storage.

Fixed Control → Programmable Control

There is one last trend that fits into the picture of why Jini makes sense today. It is represented by the change from fixed-control devices to programmable control devices. This trend started with musical instruments—both performance and composition—and is moving on to consumer electronics devices.

Until the 1970s, musicians played their instruments and got whatever sounds they could out of them based on their talents and virtuosity. Sometimes a small device would come on the market that would either alter the sound of an instrument or provide an existing “user interface” to take on a variety of sounds. Later there were “stomp boxes” which were used by electric guitar players who plugged the output of their guitars into a box and the output of the box went to either another box or the amplifier. The box would have an on/off switch activated by a footswitch sitting on top of the box which was placed on the floor, and hence the name “stomp box.”

At the same time—in the 1960s—the first modern synthesizers were developed. These were keyboard instruments where pressing a key would activate a tone generator whose waveform could be altered by filters and envelope generators to provide unique and, at the time, unusual sounds. Musicians would “program” the synthesizer to play a particular sound or set of sounds depending on which keys were played by altering a set of plugs or patches.

What followed was MIDI (Musical Instrument Digital Interface). MIDI provided an interface specification that synthesizers and other musical instruments could implement that enabled control signals to be sent to the instruments. This enabled computers to be hooked up to musical instruments, and real programming was possible.

By the 1980s, stereo equipment manufacturers were producing their own version of stomp boxes—components that would plug into each other: turntables, CD players, preamplifiers, amplifiers,equalization boxes, noise-reduction units, speakers, tuners, tape recorders and tape players. There were even switch boxes that would enable you to switch from sending signals to speakers in the living room to sending signals to speakers on the patio.

Similarly, VCRs and TVs were manufactured that could accept several input sources. By switching the inputs, it was possible to play a tape in the living-room VCR and watch it on the bedroom TV.

The connection is easy to make: Consumer electronics is nothing more than components or boxes that pass signals in and out and which can be connected together differently for different purposes. MIDI is a standard way of passing control information to components or boxes that pass signals in and out and which can be connected together differently for different purposes.

In fact, this is what the IEEE 1394 standard for consumer electronics is: an interconnect mechanism and an interface to enable consumer electronics devices (including MIDI) to be dynamically hooked up under computer control.

MIDI-controlled devices represent the blending of devices that perform a useful function with digital control that directs digital or analog information passing. When we look to bring this idea to the systems world, we get Jini.

Thin Client → Network Is the Computer

If we look at the hardware trends described above, we see a couple of things:

- computing (CPU) power is getting strong enough that almost any general purpose CPU chip can perform the calculations needed for business computing
- disks are getting very small, very fast, and very inexpensive.
- RAM is finally getting inexpensive
- the network is getting faster and more reliable
- displays are getting a little better

The implications of these trends is interesting: The computations that the hardware can perform are vastly larger and more complex than before. Numerical computation has gone through the roof. Simulations, graphics, reasoning—all of these are several orders of magnitude better than they were just a few decades ago. These gains are important, but they are more important to the professional

programmer or professional user than to the majority of people using computers for ordinary productivity reasons—these are the largely hidden gains of the industry and field of computer science. These gains are seen by looking at CPU performance, RAM size, and disk size.

In the days of mainframes and minicomputers, each company or department shared a single computer with a single space of applications that could be run, data that could be shared, and, in most cases, a community of workers and colleagues. There was no question about whether you could run a program your colleague could or whether data formats were compatible.

In the days of workstations and personal computers, people gained the luxury of guaranteed access to reliable computation. Although this opened up new possibilities for how to work, it also served to isolate people and resources into small islands that might communicate only in a rudimentary way. The gains were still important: If each user had his or her own computer, other users cannot impose a heavy load on a shared computational resource, and individuals could upgrade their own computers with smaller cost than in a centralized context.

In the 1980s we connected computers via networks and began using graphical user interfaces. In functionality, convenience, and power, this opened up a world of possibilities so that people could work and act in ways only imagined before. In particular, the two apparently diverging advantages of a common workspace and guaranteed, incrementally upgradeable resources could converge. No longer would it matter what you, your department, or even your company had purchased for computer equipment, you could find information and interact and in some cases use devices in the next office, down the hall, in the branch office, or on the other side of the globe. With a network, connecting new devices can be simple—you would no longer need to open up a cabinet or a closed box, you could simply plug it in. With the Web, the whole face of applications and interaction changed.

But in the intervening 15 or so years since the early 1980s, the network and display technology advanced hardly at all compared to advances in the other hardware areas, and the dream of reuniting separated computers into a shared workplace never materialized. The experiences of typical end-users has not got much better than they were in the days of isolated workstations and personal computers, except insofar as applications take advantage of the increased compute horsepower.

In fact, in many cases connecting computers together in networks today has decreased their reliability and sometimes their performance as well, precisely because the design of the network was an add-on, making the entire network appear as a peripheral to any particular computer, and making any good sharing rely on the ministrations of the sys admin.

Let’s look at the software trends:

- Java combines the best of static and dynamic languages providing a clean, new language that is easy to learn and use, and that provides new functionality for Web-based computing.

- Java's implementation technology means that code can be run on any computer and, in fact, can be transported as is and run somewhere else. This means that the interpretation of data residing on one computer no longer depends on programs that can run only on that computer. Computer boundaries make no difference to Java either in executable format or data format. Therefore, the Balkanization of computers which happened after timeshared mainframes fell out of favor can be reversed.
- Java's security model means that code can be trusted. In the past, a sys admin or the computer's owner would decide whether a program was trusted when he or she installed it on an individual computer. Java enables dynamic loading of code.
- Java and Web browsers are making the case that the same user interface can run on every computer.
- Object technology is mainstream, having been thoroughly explored and used in advanced development shops by early adopters. With Java, many of the rough spots in earlier languages and language implementations have been smoothed out.
- Current multi-tier and thin-client architectures have paved the way for general distributed computing.
- The Web has legitimized the concept that the actual computer is unknown that is performing the computation observed in the user interface.

An architecture based on these trends could change the face of what will formerly called the world of computing.

Jini

When we put all these things together, we get Jini. The computer-based technology is converging from one side, the component-based view is converging from another, *plug 'n' whatever* is converging from yet another, and the view of connecting up things on a network (including the Internet and the Web) and using them is binding the whole thing together.

Jini's view is that everything is a service, everything is connected without a lot of setup trouble, and controlling them is uniformly accomplished because every service is really an object with a particular interface. Jini systems will contain computer and consumer devices along with a variety of external sources—the Net, broadcast, cable, satellite, and landline.

Jini is the first step in this direction.

Conclusion

Not to be forgotten is the smiling gentleman who takes care of your computer. Today he doesn't wear a suit, but with Jini the sys admin will be still smiling because it isn't necessary to perform frustrating maintenance on network systems whenever a component is added or removed, or capabilities change.

Jini represents the best hope for consolidating the world brought forth by the revolution spurred by the Web, Java distributed computing, and consumer electronics. In the end, computers will become what those who worked with them years ago imagined: ubiquitous, almost invisible, as easy to maintain and expand as your stereo system—in fact, part of it.