USENIX
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION

The following paper was originally published in the

Proceedings of the Embedded Systems Workshop

Cambridge, Massachusetts, USA, March 29–31, 1999

AirJava: Networking for Smart Spaces

Kevin L. Mills
National Institute of Standards and Technology

For more information about the USENIX Association:
Phone: 1 510 528 8649 FAX: 1 510 548 5738
Email: office@usenix.org WWW: http://www.usenix.org

AirJava: Networking for Smart Spaces

Kevin L. Mills
Information Technology Laboratory
National Institute of Standards and Technology

Abstract

Increasingly people work and live on the move. To support this mobile lifestyle, especially as work becomes more intensely information-based, companies are producing various portable and embedded information devices. Concurrently, some interesting pico-cellular wireless technologies promise to outfit these portable and embedded devices with high bandwidth, localized, wireless communication capabilities that can also reach the globally wired Internet. An impressionist painting emerges of small, specialized devices roaming among islands of wireless connectivity within a global ocean of wired networks. Each wireless island becomes a "Smart Space", where available services and embedded devices can be discovered, accessed, interconnected with portable devices carried onto the island, and then the combination of imported and native devices can be exploited to support the information needs of the current island inhabitants. In this paper, I outline three specific human-information interaction challenges that the research community must address in order to reap the benefits of specialized information devices within Smart Spaces. Before these research challenges can be adequately addressed, the research community must have some Smart Spaces with which to experiment. I describe ***AirJava***, which combines Java Jini with pico-cellular wireless technology to empower small devices to discover each other, to exchange programs, and to interact. While a technology like ***AirJava*** should emerge in the next five years, I propose a means of building ***AirJava*** adapters today so researchers can begin experimenting with Smart Spaces.

1. Introduction

Increasingly people work and live on the move. To support this mobile lifestyle, especially as work becomes more intensely information-based, companies are producing various portable and embedded information devices. Consider for example, portable digital assistants[1], cellular telephones[2], the CrossPad[3], the InfoPen[4], active badges[5], intelligent buttons[6], and the Internet Car[7]. Concurrently, some interesting wireless technologies, including Bluetooth[8], IrDA[9], and HomeRF[10], promise to outfit portable and embedded devices with high bandwidth, localized wireless communication capabilities that can also reach the globally wired Internet. An impressionist painting emerges of small, specialized devices roaming among islands of wireless connectivity within a global ocean of wired networks. Each wireless island becomes a "Smart Space", where available services and embedded devices can be discovered, accessed, interconnected with portable devices carried onto the island, and then the combination of imported and native devices can be exploited to support the information needs of the current island inhabitants. This painting suggests some wonderful potential outcomes, once a number of research challenges have been mastered. In this paper, I outline three specific human-information interaction (HII) challenges that the research community must address in order to reap the benefits of specialized information devices within Smart Spaces. Before these research challenges can be adequately addressed, the research community must have some Smart Spaces with which to experiment. I describe ***AirJava***, which combines Java Jini[11] with pico-cellular wireless technology to empower small devices to discover each other, to exchange programs, and to interact. While a technology like ***AirJava*** should emerge in the next five years, I propose a means of building ***AirJava*** adapters so researchers can begin experimenting with Smart Spaces today. ***AirJava*** adapters have been proposed to DARPA as one component of a Smart Spaces test bed.

2. Three Challenges for Smart-Spaces Researchers

I predict that within five years specialized devices can be worn or toted into Smart Spaces, and once there can discover and interact with embedded devices at the basic level of exchanging data and programs. If I am correct, then researchers should be tackling many issues today in order to exploit this capability when the time comes. From among the myriad issues to consider, I selected three to discuss here.

2.1 Removing the Tyranny of an Interface Per Device

As many specialized devices become available, human-information interfaces can be distributed across devices and interaction modes. I call such interfaces poly-device, poly-modal (PDPM) interfaces. Depending upon application requirements, user preferences, and knowledge about human awareness, about specific tasks, and about the type of information being conveyed, tomorrow's PDPM interfaces must coordinate interactions across devices and among modalities. Surely, some sort of distributed coordination bus will be needed to exchange interaction events. In addition, a model of interaction events will be needed, as well as rules for mapping between the interaction event model and mode-specific interactions. Given a fluid set of devices available in any particular Smart Space, software mechanisms must support the dynamic composition of interfaces from among software components. Not only must composition be supported, but also rules for instantiating the optimal PDDM interface for specific tasks, given available devices and modalities. Naming and identification will be a key issue, along with authentication and access control. Since information and interaction events will fly through the air, privacy will also be important. Other issues will arise regarding shared access to devices within a Smart Space.

Some researchers are already looking into a few of these concerns. For example, the DISCIPLE[12] project at Rutgers CAIP has integrated into a single desktop interface a range of multi-modal technologies, including gaze and gesture tracking, voice recognition and speech synthesis, along with the typical display, mouse and keyboard. Similarly, the Virtue[13] project at UIUC has developed and is experimenting with a multi-modal interface for immersive virtual environments. At CMU, the Experience-on-Demand[14] project has enabled individuals to collect multimedia records of their experiences and has begun to examine how such individual experiences can be collected into a searchable database. Several collaboration technology projects are developing and evaluating multi-party distributed event buses.[12,15,16] In addition, a few projects have begun to consider the implications of multimedia, cross-device drag-and-drop[15,17].

2.2 Moving Information for People.

Some researchers believe that we will carry all of our information with us as the miracle of hardware continues to bring us ever-increasing density in disk storage, along with cheaper and faster processors. I don't believe this to be the case because human activities continue to produce information at a prodigious rate. In fact, much of the information we produce is context-dependent. For example, we typically attend meetings to conduct specific tasks. Before, after, and during these meetings we create information. Some of this information we retain personally, while other information is shared among the meeting attendees and others outside of the group. Only a small fraction of this information is our own personal information. Surely, as we move to the next meeting on the same subject we will wish to have information from the last meeting available. I argue that context can often be inferred from a combination of user, location, and task. If so, then why should a user be required to ensure that the right information is available at the right place and time? Can't the information itself take on this responsibility? Imagine active information objects that can move, that can replicate themselves, and that can communicate as a group. Such active information should be able to track the location, state, and trajectory of information users, of object replicas, and of linked objects. In addition, active information objects should be able to plan the movement, replication, and transformation of information to serve the projected needs of its users. Active information objects must also be able to implement consistency, access, and sharing policies among replicated and linked objects.

A combination of commercial and research activities show some promise that a day will soon appear in which active information becomes possible and interesting. Clearly mobile code systems such as Python[18], Aglets[19], AGNI[20], and Java[21] hint at the possibility of distributed object systems that can replicate and move. From the research world, the BARWAN[22] and MASH[15] projects at UCB are developing scalable reliable multicast protocols, beaconing protocols, and transcoding algorithms that distributed objects can use to discover each other, to communicate, and to transform their presentation. In addition, the Networks of Workstations[23] and Active Services[24] projects at UCB promise a processing-capable network infrastructure that can provide a platform for mobile distributed objects to reside within the network and to move or copy themselves toward specific Smart Spaces as users begin to congregate.

2.3 Adapting Information Delivery Using Knowledge of People, Places, and Devices.

At present, networked-based computing works because people carry in their heads a reasonably good model of cyberspace. We know where computers and printers can be found; we know how information can be organized for storage on a disk. We know, but just barely, how to locate, download, configure, and execute various plug-

ins to display information in specific formats or to convert information between formats. In fact, sometimes I think our computers and networks should pay us because we sure do a lot of work for them. Suppose, on the other hand, that our computers and networks had a much better model of the world in which we live. Would it be possible for our computers and networks to help us more than we help them today? I suggest that researchers consider trying to build Inter-Space, a model that crosses the gap between physical and logical space as we perceive it and cyberspace as it exists in our computers and networks. Suppose we could couple sensor data with resource and scene description languages to model within our computers the physical and logical space (maybe physilogical space) that people perceive and understand. If we could, then our software might be able to exploit location, proximity, and visibility of resources to determine where to deliver specific services for us. In addition, our software might be able to adapt information presentation to the specific characteristics of available devices and services. In fact, more generally, if our software has a model of physilogical space that appears reasonably consistent with our own, then we might be able to encode into our computer heuristics similar to those that we now use when reasoning on our own about cyberspace.

Think about it. Sensors of all kinds are becoming cheaper and more capable. These include digital still and video cameras[25,26], digital sensors[27], eye-tracking devices[28], radio-frequency tags[29], and global positioning system chips[30]. In addition, a few projects, such as Dataman[31] at Rutgers and the BARWAN[22] and MASH[15] projects at UCB are beginning to explore location-based networking services. Research projects at UCB[32] and at MIT[33] are also investigating methods to construct geometric models automatically from still pictures and video images.

I discussed only a few among many fruitful topics that can be explored in the context of Smart Spaces. The main concern of the remainder of this paper is to explain how researchers can begin to investigate Smart Spaces today.

3. *AirJava*

Two emerging technologies, pico-cellular wireless and mobile code, promise to provide the networking foundation for Smart Spaces. I envision that within five years vendors will offer portable and embedded devices containing low-cost chips for pico-cellular wireless communications, coupled with virtual machine interpreters that support the exchange of executable programs among networked devices. The combination of these new technologies promises to revolutionize computing and networking, as we know it today. I propose that we can accelerate this coming revolution by designing and constructing ***AirJava*** adapters that can convert any existing computer-controllable device into a prototype Smart Spaces device. Further, I propose that ***AirJava*** adapters can be used within Smart Spaces test beds, and can be supplied to researchers investigating issues associated with Smart Spaces. ***AirJava*** adapters should also inform commercial developers about design issues surrounding Smart Spaces devices. Here is how ***AirJava*** could be realized, and how Smart Spaces test beds could be created.

3.1 Pico-cellular Wireless Technology

Industry is developing two, competing wireless "transceiver-on-a-chip" technologies that can provide relatively high bandwidth (300 Kbps to 1.2 Mbps) over limited ranges (10 to 50 meters) by exploiting the unlicensed frequency band around 2.4 GHz. Table 1 gives the specifications for one of these technologies, Bluetooth[8], while Table 2 gives the specifications for the other, HomeRF[10].

While these two radio frequency (RF) technologies compete for a similar market, no matter which one prevails, future portable and embedded computing devices will clearly come equipped with some RF technology that will permit relatively high speed communications over a restricted range. Such technology will provide the communications conduit for large numbers of transient devices to begin to talk. But how will these devices discover each other and establish links and what will the devices say to one another? That's where mobile code comes into the picture.

Table 1. Planned Specifications for Bluetooth Version 1.0

- Range 10 Meters in shirt pocket or briefcase
- Network Size: 8 devices per pico-net
- Data Rate: 300-400 Kbps
- Frequency: 2.4 GHz and 1600 Hops/sec
- Supports 3 near line-quality voice links
- Optimized for cell phones and mobile devices
- Multi-point to point connections
- Point-to-point TCI/IP support
- Low power standby mode
- Higher transmit power possible
- Based on a working prototype
- More: http://www.bluetooth.com
- Main players: Ericsson, IBM, Intel, Nokia, and Toshiba

Table 2. Planned Specifications for HomeRF Version 1.0

- Range: 50 Meters in home and yard
- Network Size: unlimited
- Data Rate: 1.2 Mbps
- Frequency: 2.4 GHz and 50 Hops/sec
- Supports 6 near line-quality voice links
- Optimized for home voice and data
- Native TCI/IP support
- Low power paging mode
- Lower transmit power possible
- Based on shipping 802.11 and DECT technology
- More: http://www.homerf.org
- Main players: AMD, Ericsson, HP, IBM, Intel, Microsoft, Motorola, National Semiconductor, and more
- Peer-to-peer networking

3.2 Mobile Code

Most people have heard about the Java[21] promise of write-once, run-anywhere software. Java makes this possible by requiring Java-compliant systems to implement an interpreter for a Java Virtual Machine (Java VM[34]). In fact, a number of initiatives are underway to design chips that run the Java VM in hardware. Additionally, Sun and IBM have been designing an operating system, JavaOS[35], intended to provide operating services native to Java programs by implementing a small kernel around a JavaChip[36]. No matter what the outcome of these explorations, it appears possible to implement the Java VM on a chip (Java or otherwise) with fairly substantial memory and reasonable speed. Combining a Java VM-on-a-chip with a RF transceiver-on-a-chip could provide an interesting basis for networking Smart Spaces devices, especially with the advent of Jini[11].

Jini is a Java-based networking technology recently announced by Sun. Jini enables devices, newly added to a network, to discover a lookup service and to deposit there some key information. This information can include a description of the device and its services, along with Java classes that can be used by others to communicate with the device. In addition, Jini enables programs and devices to discover other devices in an area, and then to download Java code that permits communication with the discovered devices. Along with Jini comes extensions to the Java VM to support event distribution among distributed Java VMs, and extensions to Java Remote Method Invocation[37] (RMI) to exploit multicast protocols.

As should now be clear, the ingredients exist to provide Smart Spaces devices with powerful networking functionality in a small, low-power package. Such a package would include a RF transceiver-on-a-chip, a hardware implementation of the Java VM, and enough memory to run the Jini discovery protocols, to hold Java classes for uploading to a Jini lookup service, and to execute Java classes downloaded from a Jini lookup service. My proposed ***AirJava*** adapters would demonstrate the feasibility of these ideas, while at the same time providing a means to prototype tomorrow's Smart Spaces today.

4. A Smart Spaces Test Bed Based on *AirJava* Adapters

The following paragraphs discuss how ***AirJava*** networking for Smart Spaces can be brought to life. Three steps are needed: design of the adapter, production of some prototypes, and demonstration of a Smart Space.

4.1 Design *AirJava* Adapter.

Design of an ***AirJava*** adapter must cover both hardware and software. Hardware for a prototype ***AirJava*** adapter must provide a reasonably capable CPU, a significant amount of flash memory, a modest amount of DRAM, a serial port, a parallel port, a universal serial bus port, and the ability to accept PCMCIA interface cards, including either a RF or IR interface. The packaging must be reasonably compact, and power must be provided with batteries. To support development, the adapter should also accept PCMCIA interfaces for a secondary storage device, such as floppy disk. Software for the prototype ***AirJava*** adapter must provide an operating system or run-time environment capable of executing a Java Virtual Machine and running Jini.

Two approaches to design appear feasible. The conservative approach would base the ***AirJava*** adapter on a small, portable computer, such as the Toshiba Libretto[38]. The Libretto provides a fully functional PC-like device in a reasonably compact form, powered with batteries. The Libretto can accept up to two PCMCIA cards. The Libretto can run Windows 95, Windows 98,

or Windows NT, and thus can easily execute a Java Virtual Machine and Jini. On the downside, the Libretto is a bit heavy (just under 2 pounds) and has a short battery life (around 2 hours). As an alternative to the Libretto, we could consider some of the newer WindowsCE[39] devices beginning to appear on the market. It is unclear whether WindowsCE will support the Java Virtual Machine; however, such devices are generally smaller and have longer battery life that computers in the Libretto class.

A less conservative approach would base the ***AirJava*** adapter on GUMPS[40], a GloMo Universal Module Packaging System developed for DARPA by USC-ISI. GUMPS provides a compact brick that includes a rigid-flex circuit board, a PCMCIA-format CPU, and up to seven PCMCIA cards. Four batteries, which can yield up to 10 hours of operation, power GUMPS. The GUMPS project also envisions a mini-brick (about the size of a pack of cigarettes) that would include a CPU plus up to five PCMCIA cards. The mini-brick would be powered on only two batteries, but an operating life of 20 hours is anticipated due to better power management. While the original plan for GUMPS called for a 33 MHz 486 CPU, as of 1998 a prototype GUMPS unit included a 100 MHz 486DX4 CPU. The 1998 GUMPS prototype also included 150 Mbytes of flash storage. Producing a design based on a GUMPS-like architecture would enable us to consider a range of options for the CPU, including lower power RISC machines and JavaChips. We would also be free to consider software other than Windows. Options might include a free Unix variant, a real-time operating system, or JavaOS. Taking this approach would probably benefit from some form of collaboration with USC-ISI.

4.2 Produce Prototype *AirJava* Adapters.

Once a design has been selected and verified, probably by implementing at least two working prototype ***AirJava*** adapters, at least another ten adapters must be produced to support the planned Smart Spaces demonstration. In addition, we need to provide a cookbook of instructions so that others can produce their own ***AirJava*** adapters.

4.3 Demonstrate a Smart Space.

To show that the ***AirJava*** adapters work as expected, and to illustrate the concepts underlying networking for Smart Spaces, I propose to demonstrate a small, Smart Spaces environment. Our demonstration will include one Jini Server (connected to the global Internet as well as to a wireless cell), two portable computers, and a range of other devices. The range of other devices under consideration include a video projector, a videocassette recorder (VCR), a digital camera, a large-screen display, audio-visual equipment (such as found in the MASH room produced for DARPA by UCB), cell phones, the Cross Pad, wearable computer systems, and personal digital assistants. We need to demonstrate that devices can be brought into a wireless cell, register with a lookup service, and be discovered by other devices and by individuals. We also need to show that appropriate devices can download interface code for registered devices and then use that code to communicate with and control the registered devices. A canonical demonstration might have a person bring a video projector and VCR into a room, plug them into electrical outlets, and then leave. No other wires would be required. The devices would discover the lookup service and upload interface code. Later, a second person can enter the room with a portable computer. Using only the portable computer, the person will discover the interface code for the projector and VCR, download that code, pop in a video tape, remotely configure the VCR output to input to the projector, and then remotely control the playing of the tape, all from the portable computer. This will be accomplished without connecting the computer, VCR, and projector with cables. While the tape is playing, a third person can enter the room with a portable computer. This portable computer can also discover and download the interface code for the VCR and projector. Now, the new person can share control of the VCR with the existing person. Either person can pause, fast forward, play, and rewind the tape. Again, no communication wires. Whatever the specific demonstration chosen, the existence of Smart Spaces test beds will free the research community to consider how Smart Spaces can be integrated with wired networks.

5. Smart Spaces Islands in the Internet Ocean

Given my view of Smart Spaces as islands of high-speed wireless connectivity in an ocean of wired Internet connectivity, interesting issues need to be resolved regarding the relationship between the islands and the ocean. What is the relationship between the local Jini lookup service and emerging global-scale lookup services for the Internet? How can multicast addresses be managed so that some have local scope, while others have global scope? What is an appropriate naming scheme that overarches the Internet and local wireless pico-cell networks? How should security services and protocols be designed to support roving wireless devices

that reconnect to the Internet for information access? The point of these questions is to illustrate that as the number of wireless devices increases, the urgency of research addressing the integration of wired and wireless networking will also increase. The research community cannot afford to wait for the problem to exist; otherwise, the chaos and confusion that will result from various commercial vendors adopting ad hoc solutions will inhibit market growth in wireless devices.

6. Conclusions

In this paper, I have advocated that researchers start today to address both the networking challenges and the human-information interaction issues of Smart Spaces. To permit work to begin soon, I have suggested one means to prototype Smart Spaces by integrating some emerging pico-cellular wireless and mobile code technology into the type of Smart Spaces interface that should be widely available in the next five years. Along the way, I have highlighted some early research that lights the path toward Smart Spaces.

7. References

1. http://www.pdapage.com/
2. http://www.portableconcepts.com/
3. http://www.crosspad.com/
4. http://www.symbol.com/data/std00055.htm
5. http://www.ics.agh.edu.pl/ABng/
6. http://www.ibutton.com/
7. http://www.daimler-benz.com/ind_gfnav_e.html?/research/text/70430_e.htm
8. http://www.bluetooth.com/default.asp
9. http://www.irda.org/index.asp
10. http://www.homerf.org/tech/
11. http://sun.com/jini/
12. http://www.caip.rutgers.edu/multimedia/groupware/darpa.html
13. http://vibes.cs.uiuc.edu/Project/VR/Virtue/VirtueOverview.htm
14. http://informedia.cs.cmu.edu/
15. http://mash.cs.berkeley.edu/
16. http://www.dstc.edu.au/wOrlds/
17. http://www.maya.com/visage/link/
18. http://ftp.python.org/
19. http://aglets.trl.ibm.co.jp/
20. http://snad.ncsl.nist.gov/antd-staff/mranga/agni/
21. http://www.sun.com/java/
22. http://daedalus.cs.berkeley.edu/
23. http://now.cs.berkeley.edu/
24. http://ninja.cs.berkeley.edu/
25. http://www.conde.com/cameras/index.html
26. http://www.videocamera.com/
27. http://www.tpico.com/
28. http://www.visioncs.com/eyet.htm
29. http://www.idsystems-dialoc.com/
30. http://www.penton.com/ed/Pages/magpages/july0698/ti/0706ti3.htm
31. http://athos.rutgers.edu/dataman/
32. http://www-video.eecs.berkeley.edu/~nlachang/MVR/
33. http://graphics.lcs.mit.edu/city/city.html
34. http://java.sun.com/docs/books/vmspec/
35. http://java.sun.com/products/javaos/
36. http://www.sun.com/microelectronics/picoJava/index.html
37. http://java.sun.com/products/jdk/rmi/
38. http://www.csd.toshiba.com/cgi-bin/WebObjects/Toshiba.woa/-/ProductFamily.wo?productTypeId=1&productFamilyId=3
39. http://www.microsoft.com/windowsce/default.asp
40. http://www.isi.edu/asd/gumps/