NETWORKING EMBEDDED AGENTS

Michael N. Huhns • ***University of South Carolina huhns@sc.edu***

Most of us will soon be managing an intranet in our homes, though we might not realize it. We might also be surprised at the devices that will be networked together.

Just about every electrical device now contains one or more microprocessors. Designers typically find this a cost-effective way to provide device functionality, even when much of a processor's power is unnecessary or unused. For example, my coffee maker contains a processor, even though the appliance needn't be very smart and wastes most of its CPU cycles. Nevertheless, it is cheaper to include a general-purpose microprocessor than to incorporate custom logic devices.

My kitchen, in fact, has at least six processors, in such appliances as the microwave, the dishwasher, and the toaster. These household devices are diverse and use their processors in quite different ways, but in the future they will share one important characteristic: each will contain an agent. The agent will provide an intelligent interface to the device and, most importantly, will communicate with other devices in my home.

At present, my devices are not very agent-like, and it is not useful to think, "My toaster knows when the toast is done" or "My coffee pot knows when the coffee is ready." However, once the devices are interconnected so that they can communicate, they can arrange to have my coffee and toast ready at approximately the same time. Then I may think of them in anthropomorphic terms. For example, when I shut off my alarm clock, I can imagine it telling my kitchen devices to prepare my breakfast. When devices talk to each other, they begin to seem more like agents.

At this point my house becomes more than just a collection of processors—it becomes a multiagent system communicating over an intranet.

Web vs. Distributed Architecture

There are two basic architectures by which devices can interact. One turns each device into a Web server using the familiar http protocol. From a browser you can then access and alter the state of any device. For example, your front door lock might have its own Web page. You could be on vacation and still check whether you locked the door.

The Web is primarily a client-server architecture, however, so it does not support devices working together very well. A distributed architecture supporting peer-to-peer interactions thus appears necessary.

Several major computer companies are taking this approach. Microsoft is developing a distributed operating system called Project Millennium. Lucent Technologies is developing Inferno, IBM has T Spaces (based on the Linda tuple-spaces developed by David Gelernter1), and Oak Ridge National Laboratory is working on its Parallel Virtual Machine project. The most notable, and agent-like, of the distributed systems, however, is Jini.

Jini Architecture

Jini is Sun Microsystems' new system architecture for distributed computing among interconnected devices. Although it is still in development, we have a good idea of how the architecture works. Every device has a power cord and a network cord. When a Jini device is plugged in, it announces itself to a special network service, a matchmaker. This is all it takes to enable the device to access and use other devices on the network and in turn to be usable by those devices.

A Jini announcement is a 512-byte packet that is broadcast to the network. The network replies with a description of itself so that the device can access its services. The device then sends a message registering its own capabilities with the network. Devices find each other through the matchmaker via a lookup process. Once they are matched, devices can interact directly.

Jini does not specify any agents, but it does provide the protocols and infrastructure to enable other devices to appear agent-like. Because there are not yet any standards in this area, Sun is hoping device manufacturers will adopt its protocols rather than those of its competitors.

OSEK/VDX

Processors and agents are being embedded in other things besides consumer electronics and household devices. A high-end automobile such as a Mercedes Benz now has 24 embedded processors. A group of manufacturers in the automotive industry has formed a joint project, OSEK/VDX, to specify a standard architecture for distributed control units in vehicles. A major emphasis is on communication among the units.

Although it has not yet been addressed, a further step is the incorporation of agent capabilities. One automotive agent might monitor radar, laser, and sonar sensors to keep track of surrounding traffic. Another might be responsible for navigation. With communication among the agents, the navigator might delay its recommendation to move into a freeway exit lane if the traffic-monitoring agent reports a vehicle in that lane.

ANTS

What if you could combine software agents written in Java with miniature sensors that can listen, sniff, see in the infrared spectrum, and feel seismic vibrations? The purpose of the Autonomous Networked Tactical Sentries project at Raytheon TI Systems and the University of South Carolina is to develop such intelligent sensors and investigate their capabilities and applications.

The ANTS project is developing software for multisensing devices that could be deployed on a battlefield. An array of sensors would be able to detect the position and movement of enemy forces and materiel with minimal involvement or risk for friendly forces. Detection results would be communicated to the appropriate personnel, who would make strategic and tactical battlefield decisions.

Each sensor device contains a digital signal processor with software to operate its respective sensor, process perceived signals, and communicate with other sensors and a supervisor processor. Results from the sensors and overall processing decisions are controlled by array software supervisors responsible for the next actions to be taken. These actions include command and control for array tasking, for long- and short-range communications, and for Global Positioning System communications.

The project uses Java as the high-level development language for sensor applications. The major software components under construction are

- a real-time operating system that can respond to the data-gathering needs of sensors and can implement Java threads,
- a Java bytecode interpreter for the microprocessor running on the sensor device, and
- Java threads for agent behaviors.

The result is high-level agent technology for signal processing, command, and control.

Simulation of Household Agents

Most of us will be more concerned with agents in our homes than on the battlefield. Intelligent Home Simulation,[2,3] a research project at the University of Massachusetts, is developing systems of household agents and then evaluating those systems via a simulator that displays the agents graphically in a plan view of a house. The agents must share resources such as hot water and electricity, and a noise allowance. The following example, adapted from the Intelligent Home Simulation Web site with permission, is a typical scenario being investigated.

It is 4 p.m. and a party is scheduled for 6 p.m. Before the party

- the dishes must be cleaned,
- the dining-room tablecloth must be washed,
- the house needs to be cleaned, and
- the room temperature must be set so that the house is comfortable when guests arrive.

Because of resource limitations, all the appliances cannot run at the same time; nevertheless, time pressures require some appliances to be used concurrently. An additional constraint is that the noise level should be low between 5:00 and 5:30, since someone wants to watch TV at that time.

The home's joint resources are

- dish washer (noise, electricity, hot water),
- washer (noise, electricity, hot water),
- vacuum (noise, electricity),
- television and weather radio (electricity),
- air conditioner (electricity), and
- heater (electricity).

In addition to choosing when to schedule their activities, agents may choose behaviors. For example, the dishwasher agent may choose

- Hot, warm, or cold cycle (affects quality of wash and amount of hot water consumed).
- Duration of cycle (affects quality of wash, amount of hot water consumed, and timing and duration of noise).
- Large, medium, or small load (affects electrical efficiency).

Obviously, several trade-offs could result in a compromise solution. For example, to schedule noisy tasks, a priority scheme is necessary to determine which appliance is allowed to produce noise at specific times. It certainly makes sense to suspend the

Where to find them ...

You can find the embedded agents systems cited in this column at the URLs listed below.

IBM T Spaces • www.almaden.ibm.com/cs/TSpaces
Intelligent Home Simulation • dis.cs.umass.edu/research/mass/cs691v.html
Jini • java.sun.com/products/jini/whitepapers/architectureoverview.pdf
Lucent Technologies Inferno • www.lucent-inferno.com/Pages/Developers
Microsoft Project Millennium • research.microsoft.com/sn/Millennium
Oak Ridge National Laboratory Parallel Virtual Machine • www.epm.ornl.gov/pvm/
OSEK/VDX • www-iiit.etec.uni-karlsruhe.de/~osek/main.html

dishwasher when the weather radio "wakes up" to deliver important news. Low dishwasher quality might be balanced by the user's being able to understand the news and still get the dishes cleaned. On the other hand, full noise level could be assigned to the dishwasher if no one is using the TV or radio. The washing machine could decide to merely wash the tablecloth, skip the spin dry, and signal the dryer to do the rest.

The Intelligent Home Simulation Web site contains a collection of embeddable household agents (TV, coffee maker, vacuum cleaner, dishwasher, shower, and so on) and a simulation system whereby the agents can operate and interact. Check it out!

A World of Embedded Agents

We can look forward to greater safety and efficiency in our automobiles, a sensor-enabled environment that can detect our presence and adapt to our needs, and friendlier household devices. Realizing these benefits requires continued progress in agent technology,[4] digital signal processors, and miniaturized sensors, but it is all within our reach. Programming our houses promises to be an interesting activity. ■

REFERENCES

1. D. Gelernter, "Generative Communication in Linda," *ACM Trans. Programming Languages and Systems*, Vol. 7, No. 1, 1985, pp. 80-112.
2. R. Vincent et al., "Survivability Simulator for Multi-Agent Coordination Learning," *Proc. Int'l Conf. Web-Based Modeling and Simulation*, Vol. 30, No. 1, ACM Press, New York, 1998, pp. 114-119.
3. V. Lesser et al., "A Multi-Agent System for Intelligent Environment Control," *Proc. Third Int'l Conf. Autonomous Agents*, to be published May 1999. Also available as Tech. Report TR-98-40, Univ. of Massachusetts, 1998.
4. M.N. Huhns and M.P. Singh, *Readings in Agents*, Morgan Kaufmann, San Francisco, Calif., 1998.