



Embedded Internet Systems Come Home

Robert E. Filman
 RIACS
 NASA Ames Research Center

In May 1998, *Internet Computing* ran a special issue on embedded Internet technologies. In this issue, we revisit that topic, with a particular emphasis on the Internet in the home.

An embedded computer is one whose primary purpose is to control and monitor some other device. The designers of general-purpose computers like workstations and mainframes anticipate that their users will run many different programs on them for a variety of purposes.

On the other hand, embedded computers are, by and large, engineered knowing in what device they will be used and to what purposes that device will be put. While general-purpose computers usually come with fast processors and a wealth of primary and secondary storage, embedded computers often have to be small, inexpensive, and even low power. Embedded computers have been critical for applications such as security systems, environmental controls, office equipment, and telemedicine.

Thinking Machines

Early machines were, shall we say, "stupid."

With a few exceptions, they did the one thing they were built to do, with little ability to sense their environment and respond to changes. Archetypically, control was instinctually "hard-coded" by physical devices like cams and gears that sensed by one-bit or linear mechanisms. Intelligence was supplied by a human operator.

Before microprocessors, machines were, well, mechanical. Embedded computers changed all that. By allowing fairly complex decision making in devices, heating systems could prepare buildings for weekday occupancy, medical instruments could recognize life-threatening situations and sound alarms, automobiles could dynamically tune ignition cycles for greater efficiency, elevators could organize for faster service, airplanes could autopilot themselves to their destinations, clothes dryers could sense the relative moisture in their loads and adjust the heating cycle, and so forth. In the same way neurons enabled primitive organisms to evolve into complexly behaving animals, embedded computers allowed the progression from simple machines to "thinking" and "sensing" machines.

Communicating Machines

The Internet gives machines a way to communicate. Neurons let animals hunt for food, evade predators, use tools to obtain new sources of nutrition, and find shelter from storms. Communication allowed a leap in animal behavior, enabling wolves to howl a coordinated hunt, bees to dance their comrades to the best flowers, and people to ... well, we won't go into all the ways humans communicate, or all the trouble that's caused.

Communication produced a revolution in animal behavior, at least for some species. Communicating, intelligent machines will be able to coordinate their behaviors to achieve greater efficiencies (the blinds opening and closing in synchrony with the air conditioning system, for example) and new services (remote medical monitor paging your doctor when your vital signs get critical; your alarm clock coordinating with your calendar device). Communicating intelligent machines may prove as revolutionary as communicating animals.

Communication requires two key ingredients: a channel over which to communicate and a language for expressing the communications. Bee dances are opaque to people; Swahili is incomprehensible to the speaker of Farsi. And even if we could communicate with bees, it's likely we'd have little to say to each other—the human appreciation of flowers being independent of their pollen density. Correspondingly, there are three key technical issues in engineering a world of communicating machines:

- the underlying communication channel
- the linguistic (programmatic) system for enabling communication, and
- what to communicate about.

For the embedded Internet, communication channels are seen in technologies such as Bluetooth, CEBus, UPnP, and TCP/IP; linguistic mechanisms in notions such as Java RMI, HTTP, and SNMP; and what to communicate about by different notions of services and service discovery, and XML-like ontologies. Engineering such systems requires balancing the costs of fatter communication channels and grander processor environments against the additional functionality such richer technology can enable.

Home: The Next Frontier

The two theme papers in this issue address the last two of those concerns in the context of home automation systems. Home automation demands

Embedded Resources

There are a variety of different embedded technologies competing for the emerging home networking market. The following are a few of the major initiatives.

- Bluetooth SIG • <http://www.bluetooth.com/>
- CEBus Industry Council Home Plug and Play • <http://www.cebus.org/>
- Embedded Java Application Environment • <http://java.sun.com/products/embeddedjava/>
- Home Audio Video Interoperability (HAVi) • <http://www.havi.org/>
- Open Services Gateway initiative (OSGi) • <http://www.osgi.org/>
- Universal Plug and Play (UPnP) Forum • <http://www.upnp.org/>
- Windows NT Embedded Web Site • <http://www.microsoft.com/windows/embedded/nt/>

More information on embedded technologies is available from:

- *Embedded Systems Programming* magazine • <http://www.embedded.com/mag.htm>
- *Embedded Technology* • <http://www.embeddedtechnology.com/>
- Embedded Systems conference series • <http://www.esconline.com/>

trivial “plug and play” for its devices: it can't require a system administrator to integrate your toaster into the network. On the other hand, home automation needs transparent security: I shouldn't be able to tell your toaster that your English muffin is done.

In “Internet Access to the Home Area Network,” Umar Saif, Daniel Gordon, and David Greaves expand on their work on AutoHan, a self-configuring software architecture for home device management. They discuss a flexible, XML-based architecture for describing the properties and capabilities of home devices, and present a naming and addressing scheme large enough to provide an address for every doorknob in your house over IP4.

In “A Software Architecture for Open Service Gateways,” Li Gong of Sun Microsystems argues that home devices will flourish best if they present a uniform API to application programmers, and that the ideal such interface is Java. He describes the Open Service Gateway initiative, a Java-based system that combines services with event and security mechanisms. □

Robert E. Filman is a rocket computer scientist at the Research Institute for Advanced Computer Science (RIACS), NASA Ames Research Center. He is *IC*'s associate editor in chief.

Filman can be reached at rfilman@mail.arc.nasa.gov.