
TOM WHITE

Smart Sensors to Network the World

An emerging class of pillbox-size computers, outfitted with sensors and linked together by radios, can form perceptive networks able to monitor a factory, a store—even an ecosystem. Such devices will more intimately connect the cyberworld to the real world

By David E. Culler and Hans Mulder

Today we coddle our computers. They are fragile and expensive, so each typically belongs to an owner who looks after it. When we need to connect many of them into a single system, we hire experts and set aside large amounts of time and money for the job. The sheltered cyberworld of computers still hardly intersects with the real world of birds and trees, ships and bridges.

Where the two worlds do connect, it is often because people have carefully altered objects and methods of work to be computer-friendly. Stores stick bar codes on everything they sell or ship. Warehouse clerks attach radio-frequency identification (RFID) tags to pallets. Tagged goods must then funnel through a few scanners so that the computers can do their accounting.

A new class of microelectronic devices frees us to mix computers much more freely with the objects and places of everyday experience. Our research groups at the University of California at Berkeley and Intel, as well as at start-up firms and other universities, have joined simple computers to radio transceivers and sensors to form small autonomous nodes that we call "motes." Running an operating system known as TinyOS, each mote links up with its neighbors from the moment it is turned on. Although these smart sensors have limited power and processing capabilities, an assembly of hundreds of them can spontaneously organize into a perceptive network that is spread throughout the physical world, able to perform tasks no ordinary computer system could.

These wireless gadgets are affordable and sensitive enough, for example, that dozens have been strapped to redwood limbs to form a new kind of scientific instrument—we might dub it a “macroscope”—that records the microclimate around an entire tree in each of several parts of a forest. The battery-powered motes are small enough that this past summer biologists placed 150 of them within and outside the nests of seabirds to help ecologists learn why they brood their eggs where they do. In addition to collecting and processing data, wireless nodes deduce how to route information through their neighbors so that it efficiently reaches an Internet-connected base station. That capability allows Intel to envision placing thousands of such sensor nodes in its manufacturing plants to monitor critical machinery and prevent costly outages.

It is easy to imagine that as the price of motes fall and their capabilities rise along with the rest of semiconductor technology, this novel class of machines will be put to myriad uses: boosting productivity, opening fresh avenues for scientific research, and enabling creative ways to prevent and respond to emergencies, environmental troubles and military engagements. But we do not underestimate the difficult engineering required to realize this potential. A mote is not a miniaturized PC; every aspect of the system, from the way it runs programs to the way it communicates data, must be optimized to conserve power, space and cost. A rule of thumb in designing motes and their networking protocols for long-lived applications is that each device should sleep 99 percent of the time and do its energy-consuming work in the remaining 1 percent.


CHARLES O'REAR

“MACROSCOPE” consisting of a network of several dozen smart sensors, or “motes,” was assembled throughout redwoods by researchers in April.

Computing in the Wild

THE NATURAL WORLD is not computer-friendly. To function outdoors and in industrial settings, computers must be “hardened” with enclosures to protect the electronics from weather, soil, wild animals, and jolts. But sensors must be exposed to the environmental conditions they monitor. Motes have small, inexpensive shells and use redundancy to increase their reliability.

They are designed to be inexpensive enough for deployment in large numbers to gather very detailed information about the environment. Networks of them are dense enough that it is acceptable if some fraction die and smart enough that the overall system can adapt to the loss and keep working. Designing for loss and the uncertainty of the physical world presents new challenges but allows perceptive networks to be economical, portable and unobtrusive.

While designing successive generations of motes and their networking capability, we have conducted pilot projects to help identify how the technology needs to evolve to be most useful for various applications. Several years ago, for example, we began working with biologists on studies of flocks of about 18,000 petrels that live at sea but fly inland every summer to lay eggs and rear their chicks on Great Duck Island, a small, uninhabited isle off the coast of Maine. The birds nest in underground burrows, which cluster around particular places on the island. Understanding why they choose the brooding spots they do may improve coastal wildlife conservation strategies.

As with many aspects of biology and ecology, what matter are local environmental conditions. A petrel does not dig a burrow where it does because of the average temperature or wind speed on the island but because of how warm or windy it is at that particular spot. Other variables are probably important, too, so biologists would also like to measure humidity levels and the amount of light—both inside each burrow and just outside of it. And investigators want to observe these factors over the nesting season to learn how they correlate with the presence of eggs and the habits of parent birds.

Since 2002 we have been using motes to study the petrels' nesting behavior. The biologists are asking a lot of the technology: to work well for this application (and many others like it), each mote must carry a suite of sensors. In this case, tempera-

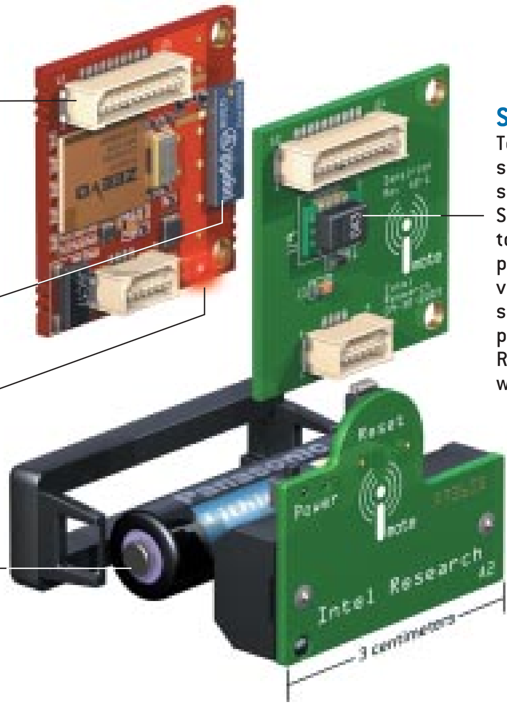
Overview/Perceptive Networks

- Thumb-size computers called motes combine microprocessors and memory with radio transceivers, onboard power supplies and a variety of sensors.
- Motes are inexpensive enough to deploy by the thousands in factories, farms or wildernesses. Each mote can collect and analyze sensor readings independently but can also link up with neighboring motes in a meshlike perceptive network.
- Motes are already being manufactured by Crossbow, Intel and others. Early prototype systems have helped biologists study seabird nests and redwood groves. Perceptive networks are also being developed to monitor vibrations of manufacturing equipment, strain on bridges, and people in retirement homes.

ANATOMY OF A MOTE

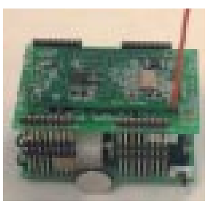
Smart nodes combine processing and memory capabilities with sensors, wireless communications and a self-contained power supply. A prototype produced by Intel Researc is drawn below.

Motes are typically designed in stackable layers so that a processing layer can be connected to a wide variety of sensors and power sources to suit a range of applications.

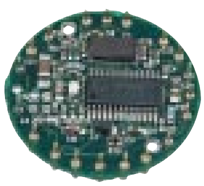


OTHER MOTE PLATFORMS

Mica mote is being used in some 500 research projects, including a "Robomote" made at the University of Southern California. Using motes that control actuators, perceptive netowrks can operate machinery, regulate indoor environments, and change the position of the sensors in the system.



Mica2Dot mote made by Crossbow incorporates a 900-megahertz radio transceiver, 640 kilobytes of memory and a coin-size three-volt battery. Sensor layers connect to the processing board using pins on the circumference of the device. These motes formed the redwood- and seabird-monitoring networks.



Smart Dust prototype, developed at Berkeley, performs many TinyOS functions in hardware rather than in software. Thanks to its ultraefficient radio and analog-to-digital converter, the five-square-millimeter device would be able to run on energy harvested from ambient light or vibration.

ture, atmospheric pressure and humidity sensors record microenvironmental conditions, while passive infrared sensors detect the presence of warm birds and eggs. Yet the device must be no more than a few centimeters in size so that it does not disturb the bird and its chicks. Clearly, it must be wireless, because it is not feasible to string power and network cables over acres of nesting grounds. So the device must carry its own energy, enough to power the electronics for the annual nesting season. And it must keep running and communicating its information through other nodes in the network without any human contact.

Many of the system's design constraints boil down to power. A single bulb on a strand of Christmas tree lights consumes about half a watt. Whether they use batteries, solar cells or gadgets that harvest energy from vibrations, as self-winding watches do, motes must operate on $\frac{1}{10,000}$ of this power on average.

A solar cell that is one square centimeter in size generates about 10 milliwatts (thousandths of a watt) in full sunlight; solar cells work poorly indoors and not at all inside burrows. A typical coin-size battery stores about three watt-hours of electrical energy. Microcontrollers generally burn about 10 milliwatts of power; low-power radios burn about 20 milliwatts. Many useful sensors consume similar amounts of power. Even running at a mere 30 milliwatts, however, such a battery will last for less than five days.

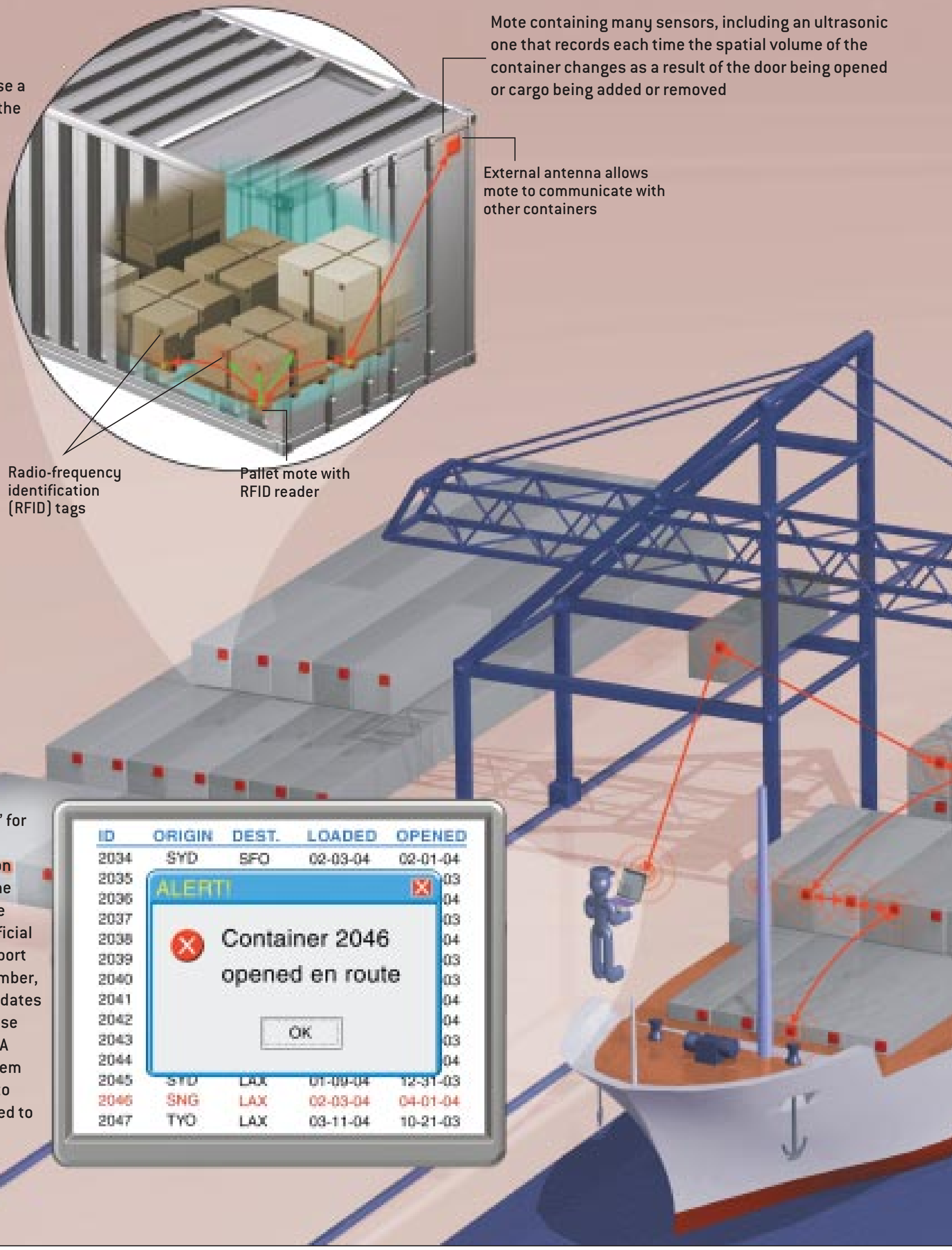
That is why motes spend about 99 percent of their lives "sleeping" in a standby mode that drops the power consumption to a few millionths of a watt. Several times each second, the device flicks on its radio to check for incoming messages,

GRIFF WASON (*illustration*); COURTESY OF GAURAV S. SUKHATME *U.S.C. Embedded Systems Laboratory* (*Robomote*); DAVID CULLER (*Mica2Dot*); TKTKTKTK (*Smart Dust*

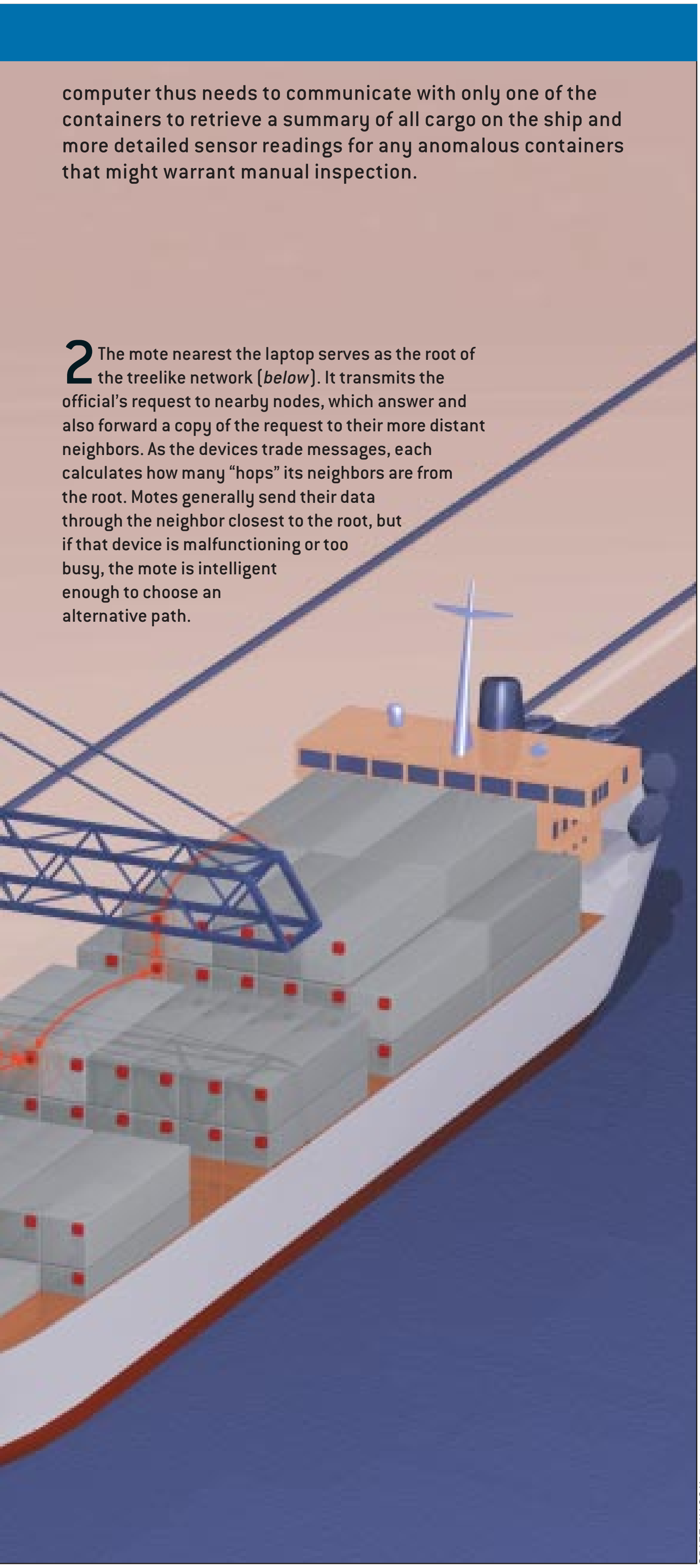
A SELF-ORGANIZING SMART SENSOR NETWORK

A perceptive network of smart, wireless sensors called motes could help customs officials prevent weapons or contraband from being smuggled in through ports. Each cargo container might hold numerous motes able to self-organize into wireless networks. Those on pallets inside each container could link up with a node on the container wall, and that device could in turn share data with motes on all the other containers on the ship in an efficient, treelike network. The port official's laptop

1 A mote on each pallet could use a built-in RFID reader to record the identity and origin of each box it carried (*colorTK arrows*). A node mounted on the wall of the container could aggregate data from the pallets inside and use its own sensors to note whether the container became too hot, cold or humid; whether it was dropped or bumped; or whether material was, suspiciously, added or removed during the voyage.



3 A program called TinyDB ("DB" for database), created by Intel Berkeley and U. C. Berkeley, runs on each mote and effectively hides the complexity of the network from the user. For example, the customs official might request that the network report each container's identification number, its origin and destination, and the dates it was loaded and last opened. Those opened en route could be flagged. A shipping agent might use the system differently, querying the network to identify any cargo that was exposed to high temperatures or humidity, potentially damaging the goods.



computer thus needs to communicate with only one of the containers to retrieve a summary of all cargo on the ship and more detailed sensor readings for any anomalous containers that might warrant manual inspection.

2 The mote nearest the laptop serves as the root of the treelike network (*below*). It transmits the official's request to nearby nodes, which answer and also forward a copy of the request to their more distant neighbors. As the devices trade messages, each calculates how many "hops" its neighbors are from the root. Motes generally send their data through the neighbor closest to the root, but if that device is malfunctioning or too busy, the mote is intelligent enough to choose an alternative path.

GRIFF WASON

but if there are none, the radio is shut off within milliseconds. Similarly, the sensors usually take their readings of the temperature, light level and so on only once every few minutes.

Most techniques for saving energy exploit the intelligence within the device to perform local processing and to turn off unneeded resources. We often use a simple, low-power sensor to turn on others in response to a preprogrammed stimulus. When a bird enters a nest, for instance, the temperature rises quickly. A heat-sensitive circuit could take readings once a minute and trigger a camera or other power-hungry sensors on the mote to start recording whenever the burrow warms rapidly.

The onboard processor offers other ways to save power. Communicating one bit of data through the radio transceiver costs as much energy as executing roughly 1,000 processor instructions. The mote can conserve power by storing and aggregating sensor readings, rather than sending them out immediately. The processor can also compress information before it is sent and can summarize the sensor logs with an average or the high and low values if the details are not crucial. Nodes may swap sensor data with one another, identify important observations and then send simplified descriptions out to the user. There is no way around certain network-protocol conversations between nodes, but these messages can be held until there are sensor measurements to transmit and then stuffed into the same "envelopes" as those packets of data.

The project on Great Duck Island successfully tested these and other ideas for making the most of wireless sensor networks on this scale. And in the 2002 breeding season alone, the macroscope there was able to take more than a million measurements, adding far more detail to biologists' picture of a key scene in the life cycle of petrels. Just as important, the technology allowed scientists to observe the birds without alarming them by a human presence.

Networking out on a Limb

COMPARED WITH a handheld PDA, an individual mote is a computational weakling [*see box on page 87*]. Each mote has a microcontroller that can handle four million to 10 million instructions a second, whereas a palmtop can whip through about 400 million a second. But unlike PDAs, motes can join forces in ad hoc networks to form a system that has greater computational power than its parts.

In April we assembled such a system by strapping 120 plastic-encased motes to the trunk and limbs of redwoods at a grove near Sonoma in northern California. The goal is to build a de-

THE AUTHORS

DAVID E. CULLER and *HANS MULDER* have collaborated for many years on wireless sensor node research. Culler is professor of computer science at the University of California, Berkeley, and was the founding director of Intel Research Berkeley. For the past decade his research has focused on ways to combine large numbers of computers to work in a highly coordinated fashion. Hans Mulder is associate director of Intel Research and director of the Intel Research network of university labs. He initiates and drives research on ubiquitous computing and distributed systems.

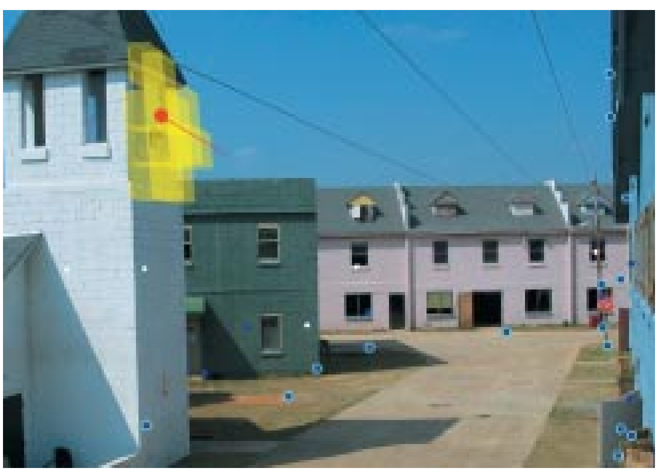
tailed picture of how the microclimate enveloping such trees changes and how the trees shape the local environment through their shade, respiration and water transport. For this project, cost determines the density of measurement, and power determines the lifetime. The network will run for several weeks on AA-size lithium batteries. The larger challenges in this case are collecting data from devices that are so high up that they are out of radio contact with the ground and reprogramming the motes as needed to test different hypotheses about the interaction between the forest and the environment.

The low-power, silicon microchip radios in the devices can transmit and receive data about as fast as a dial-up modem, but their range is limited to less than 30 meters—sometimes much less. In a forest, wet wood and needles attenuate the signals. A mote stuck to a tree trunk often cannot communicate directly with a neighbor on the other side of the trunk just a meter or two away. To cope with these limitations, a mote might beam its sensor readings to a mote on a higher limb. From that node, the data packets could travel to motes in the treetop and then continue hopping from one device to the next down the far side of the tree, over to other trees on the edge of the grove, and finally out for storage and analysis on a more powerful computer. The sensor-network macroscope in Sonoma is designed to relay its redwood measurements to a PDA-like cellular device on the ground and then through the Internet to a server in Berkeley, 70 kilometers away.

When a deployment involves hundreds of motes, it is not practical to set up such multihop networks by configuring each device individually, as is done in a typical office or cellular network. For many applications of perceptive networks—monitoring equipment, raw materials, and products in a factory or on a farm, for example—the arrangement of motes will be constantly changing. So the motes self-organize into networks. Special algorithms running in each sensor node determine how many hops it is from the server and which of its neighbors offers the most efficient path to that collection point at any given moment [*see box on preceding two pages*].

Mote-to-mote communications are coordinated by an operating system on each mote as well as by an application program that can run in pieces, with different pieces on different nodes in the network. Standard operating systems, such as Windows or Unix, are much too large and processor-intensive for these tiny devices. That is why Culler's group at Berkeley created TinyOS, an extremely compact, network-centric operating system that is now "open source" and maintained by a community of programmers using it in their own work.

TinyOS is stingy with power; it forces mote programs to shut down except when certain events occur that warrant action. The operating system is also highly modular. If a program needs only certain functions from TinyOS, the nonessential parts of the operating system are automatically removed from the mote. This modular approach ensures that the program code fills as little memory as possible, leaving more room for sensor data. Modules also enhance the robustness of the devices by limiting how the distinct parts of the software interact.



GUNSHOT LOCATION (*red*) was triangulated within seconds by a network of motes (*blue dots*) containing sound and shock-wave sensors.

Commanding a Computational Army

PERHAPS THE MOST challenging long-term question raised by perceptive networks is how we can most efficiently and reliably program the thousands of smart nodes that may coexist in a system. This scale is no idle conjecture: Intel has begun installing prototype nodes called iMotes on pumps and other machinery at its Jones Farm fabrication plant in Hillsboro, Ore. About 4,000 places in such a facility hold equipment that should be monitored for signs of wear and failure—so many locations that currently engineers can check only selected pieces every one to three months. That is not frequent enough. Not long ago a device failure occurred between two vibration inspections at an Intel plant, causing a costly interruption of operations. An entire system of 4,000 iMotes could be created now for well under $1 million that could provide hourly updates on the health of the plant's infrastructure, with no need for roving engineers. But we have had to think carefully about how to program and debug the network so that it remains manageable as it grows to include thousands of sensor nodes.

Because of the tight constraints on power use and processor speed, a perceptive network functions differently from the Internet and office LANs, where computers have individual names and addresses and most messages are sent from one machine to a specific recipient machine. In sensor networks, one node generally broadcasts messages to many, with the intended recipients identified by attributes such as their physical location or sensor value range.

Recently a team at Intel and Berkeley created software called TinyDB that makes a perceptive network system function much like a database. A user can "query" all the smart nodes at once with a request for, say, any vibrations between 40 and 120 hertz stronger than a certain level. The request enters the network at its "root" node, which forwards copies to its neighbors and so on until all sensors have received the command.

Motes that lack vibration sensors may ignore the message; others may turn on their sensors if they have been sleeping; still others may run a series of calculations on the data logged in

CREDIT

PROTOTYPES OF PERCEPTIVE NETWORKS

PURPOSE	SENSORS	NODES	ORGANIZATION
Observes weather and nesting behaviors of seabirds on Great Duck Island, Me.	Temperature, humidity, infrared	150	Intel, Berkeley
Analyzes activity of residents in elder care facilities in Portland, Ore., and Las Vegas	Motion, pressure, infrared	130	Intel
Antitank mines communicate and reposition themselves to close gaps in a mine field	Location, orientation, acceleration	96	DARPA
Collects readings on microclimates surrounding redwood trees	Temperature, humidity, light, atmospheric pressure	80	Intel, Berkeley
Monitors the performance of pump and scrubber motors in a microchip factory	Vibration and RPM	70	Intel, Berkeley
Maps growth conditions and susceptibility to fungal infections in a vineyard	Temperature	65	Intel
Listens for gunshots and then triangulates shooter position	Sound, shock wave, location	45	DARPA, Vanderbilt
Records microclimates within James San Jacinto Mountains Reserve, Calif.	Temperature, humidity, rainfall, light, wind	30	U.C.L.A.
Monitors movement of Golden Gate Bridge	Vibration, acceleration	Under design	Berkeley

their memories, extract readings that meet the requested criteria, and pass that information back to the root mote for collection. All the user sees is a spreadsheetlike list of the relevant measurements and locations. Software running on a high-powered server could then perform a wider analysis of the trends to determine which machines require maintenance.

In the redwoods, biologists are most interested in the dramatic temperature and humidity fronts that move up and down the tree every day, creating powerful gradients that may drive the flow of nutrients. To track these fronts, motes pool their data and search for spatial patterns. As scientists and engineers learn from their observations through the macroscope, they periodically change the tasks the network performs.

To replace the software on motes with updated versions, we have drawn on lessons from Internet viruses and worms. A new program is packaged in a special form and delivered to the root mote, which installs it and "infects" its neighbors with the package. The upgrade makes its way through the network like an epidemic, but it does so in a more controlled fashion that avoids redundant communications and adapts to the way that the motes are scattered in space.

This reprogramming model immediately suggests one of the harder problems in sensor network design: how to secure them against hackers, viruses and eavesdroppers. TinyOS has built-in algorithms that can authenticate the identity of motes. But for the system to work well, keys must be distributed to a large number of small nodes in reliable and uncomplicated ways. Malefactors can attack perceptive networks using strategies that are different from what is generally seen on the Internet. One promising way to defend the networks is to treat the effects of an attack as essentially another form of noisy sensor data, so the perceptive network as a whole will still function even if a small fraction of nodes has been compromised. But as with all forms of computer security, the protection of mote systems will be a constant battle of wits.

As we gain experience with this new kind of tool, we find that it fails in unfamiliar ways. A sensor network is unlikely to crash outright, but as some nodes die and others generate noisy or corrupt data, the measurements of the overall system may become biased or inconsistent. We and other computer scientists are working on techniques to judge the health of a perceptive network by perturbing the system in a controlled way and observing how the sensors respond.

Over the next decade or so, wireless sensor nodes and perceptive networks will probably evolve into a much less distinct and less visible form. Devices will gradually migrate out of their little boxes and will instead be incorporated directly into various materials and objects. Many will draw energy directly from the environment in which they operate. To the extent that these kinds of computers infiltrate homes, workplaces, farms, transportation terminals and shopping sites and are able to sense the presence, motion and even physiological states of individuals, they will raise substantial privacy concerns. Indeed, a discussion about such technology has already begun over the use of passive RFID tags [see "RFID: A Key to Automating Everything," by Roy Want; Scientific American, January]. Privacy issues are quite straightforward for many valuable applications—such as monitoring vibrations in pumps, fatigue in beams or microclimate in forests—but in other domains a careful balance must be struck to ensure that the technology properly empowers the individual.

With appropriate debate, these matters will surely be surmounted—mote technology is too useful to ignore. By connecting us to the physical world in ways not previously possible, it promises to advance scientific pursuits and the businesses of manufacturing, agriculture, construction and transportation. SA

MORE TO EXPLORE

Mica: A Wireless Platform for Deeply Embedded Networks. Jason Hill and David Culler in *IEEE Micro,* Vol. 22, No. 6, pages 12–24; November/December 2002.

Query Processing in Sensor Networks. Johannes Gehrke and Samuel Madden in *Pervasive Computing,* Vol. 3, No. 1, pages 46–55; January 2004.

The Emergence of Networking Abstractions and Techniques in TinyOS. David Culler et al. in *Proceedings of the First USENIX/ACM Symposium on Networked Systems Design and Implementation.* USENIX, 2004.

Great Duck Island monitoring network: **http://greatduckisland.net**

TinyOS: **www.tinyos.net**

U.C.L.A. Center for Embedded Networked Sensing: **http://cens.ucla.edu**