# Habitat Monitoring: Application Driver for Wireless Communications Technology*


Alberto Cerpa, Jeremy Elson, Deborah Estrin†, Lewis Girod
UCLA Computer Science Department
Los Angeles, California 90095-1596
{cerpa,jelson,destrin,girod}@lecs.cs.ucla.edu

Michael Hamilton
James San Jacinto Mountains Reserve
Box 1775
Idyllwild, California 92549
director@jamesreserve.edu

Jerry Zhao
USC Information Sciences Institute
4676 Admiralty Way
Marina del Rey, California 90292
zhaoy@isi.edu



## ABSTRACT

As new fabrication and integration technologies reduce the cost and size of micro-sensors and wireless interfaces, it becomes feasible to deploy densely distributed wireless networks of sensors and actuators. These systems promise to revolutionize biological, earth, and environmental monitoring applications, providing data at granularities unrealizable by other means. In addition to the challenges of miniaturization, new system architectures and new network algorithms must be developed to transform the vast quantity of raw sensor data into a manageable stream of high-level data. To address this, we propose a tiered system architecture in which data collected at numerous, inexpensive sensor nodes is filtered by local processing on its way through to larger, more capable and more expensive nodes.

We briefly describe Habitat monitoring as our motivating application and introduce initial system building blocks designed to support this application. The remainder of the paper presents details of our experimental platform.



**Keywords:** low-power wireless, sensor networks, testbeds, applications


## 1. INTRODUCTION

During the last decade, networking technologies have revolutionized the ways individuals and organizations exchange information and coordinate their activities. In this decade we will witness another revolution; this time one that involves observation and control of the physical world. The availability of micro-sensors and low-power wireless communications will enable the deployment of densely distributed sensor/actuator networks for a wide range of biological, earth and environmental monitoring applications in marine, soil, and atmospheric contexts. This technology has particular relevance in many Latin American countries because of its applicability to environmental monitoring of the diverse and unique ecosystems.

To achieve scalability, robustness, and long-lived operation, sensor nodes themselves will execute significant signal processing, correlation, and network self-configuration inside the network. In this way these systems will emerge as the largest distributed systems ever deployed. These requirements raise fascinating challenges for Information Technology and communication research, as well as for their application domains. One of the novel issues for network design is the shift from manipulation and presentation of symbolic and numeric data to the interaction with the dynamic physical world through sensors and actuators. This raises the need for good physical models, which requires extensive data analysis of monitored data. A second challenge arises from the greatly increased level of environmental dynamics. While all good distributed systems are designed with reliability in mind, these new target applications present a level of ongoing dynamics that far exceeds the norm. Perhaps the most pervasive technical challenge arises from the energy constraints imposed by unattended systems. These systems must be long-lived and vigilant and operate unattended. Unlike traditional Internet systems the energy constraints on un-tethered nodes present enormous design challenges. Finally, as with the Internet, there are scaling challenges. However, given the other characteristics of the problem space, the traditional techniques are not directly applicable, and alternative techniques must be developed.

This paper focuses on a particular application of embedded wireless sensing technology. The habitat sensing array for biocomplexity mapping emphasizes the need for continual automatic self-configuration of the network to adapt to environmental dynamics, and the use of coordinated actuation in the form of programmed triggering of sensing and actuation to enable identification, recording and analysis of interesting events.

We introduce the key architectural principle for constructing long-lived wireless sensor networks, adaptive self-configuration, and then describe its applicability to Habitat monitoring. In


*To appear in the Proceedings of the First ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean, 3-5 April, 2001, San Jose, Costa Rica. Also published as UCLA Computer Science Technical Report 200023, December 2000.



†Correspondence author.

the subsequent section we describe our tiered architecture, time synchronization techniques, and experimental platform developed to support this and other applications.

## 2. ADAPTIVE SELF-CONFIGURING SYSTEMS

The sheer number of distributed elements in these systems precludes dependence on manual configuration. Furthermore, the environmental dynamics to which these elements must adapt prevents design-time pre-configuration of these systems. Thus, realistic deployments of these unattended networks must self-reconfigure in response to node failure or incremental addition of nodes, and must adapt to changing environmental conditions. If we are to exploit the power of densely distributed sensing, these techniques for adaptation and self-configuration must scale to the anticipated sizes of these deployments. In recent years, some work has begun to allow networks of wireless nodes to discover their neighbors, acquire synchronism, and form efficient routes [Pottie-Kaiser00]. However, this nascent research has not yet addressed many fundamental issues in adaptively self-configuring the more complex sensing and actuation systems described here, particularly those arising from deploying embedded systems in real-world, environmentally-challenging contexts [Estrin-et.al.99]

Driven by our experimental domains, we are using this experimental platform to develop techniques for self-configuration:

- Integrated techniques for self-assembly and self-healing in these deeply distributed systems. These methods should enable self-configuration—both at the lower-level communication layers in addition to higher levels such as distributed name spaces.
- Simple localized algorithms that effect coordinated data collection and processing to achieve measurement aggregation or higher-level alert generation [Abelson99]. Preliminary research indicates that a particular paradigm for network organization, directed diffusion [Intanago-et.al.00], can efficiently achieve such coordination and resource allocation needs, but considerable experimentation and modeling work is still required.
- Protocol and system level techniques that enable energy-efficiency beyond what is feasible with low-power component design alone. Such techniques, designed for robust operation, can achieve system longevity without sacrificing vigilance.
- Techniques for time synchronization and localization in support of coordinated monitoring. At the target node scales, relying on global positioning systems alone may not be appropriate.
- In some contexts the ability of the node to move itself (or selected appendages), or to otherwise influence its location will be critical. Distributed robotics [Mataric95] in a constrained context will greatly extend the capabilities of these systems. Benefits of including self-mobilizing elements [Sukhatme99] are: self-configuring systems to adapt to realities of an inaccessible terrain, developing a robotic ecology for delivering energy sources to other system elements, and obtaining coverage of a larger area.

## 3. HABITAT SENSING ARRAY FOR BIOCOMPLEXITY MAPPING

The challenge of understanding biocomplexity in the environment requires sophisticated and creative approaches that integrate information across temporal and spatial scales, consider multiple levels of organization and cross-conceptual boundaries [Walker-Steffen97, Gell-Mann95]. Long-term data-collection for systematic and ecological field studies and continuous environmental monitoring are the domain of Biological Field Stations, and offer opportunities to establish cross-cutting and integrated investigations that facilitate studies of biocomplexity [Michener-et.al.98, Lohr-et.al.95]. Over the past two decades we have seen extraordinary developments in the field of remote sensing and automated data collection, resulting in dramatic increases in spatial, spectral and temporal resolution at a geometrically declining cost per unit area [Colwell98]. Multi-purpose data analysis and visualization software provides tools to study large and complex data sets. The Internet facilitates global data access, distributed data processing, collaborative studies, virtual proximity and tele-robotic operation.

Remote sensing from satellite and airborne sensors has proved to be a tremendous tool for studying "large" biodiversity (e.g. spatial complexity of dominant plant species). While many scientists and land managers attempt to study biodiversity using top down remote sensing tools, the fact is that the vast majority of the biodiversity, and resulting biocomplexity, within an ecosystem exists at very small scales, and is not readily observable with even the best airborne and satellite based sensors [Keitt-Milne97]. To get down to where the complexity is, so to speak, sensing and monitoring needs to become ground based [Hamilton92, Hamilton00]. Breakthroughs in VLSI digital signal processing, miniature sensors, low-power micro-controllers and wireless digital networks will make possible the development of cheap and nearly ubiquitous ground-based monitoring systems for outdoor field. Fresh opportunities afforded by these technologies allow us to rethink how Biological Field Stations can participate in the global effort to answer the big questions posed by biocomplexity.

Observation techniques involving cameras and microphones are in increasingly widespread use, however they involve small numbers of devices and require continuous human observation, greatly constraining their capabilities in natural environments. Unattended, heterogeneous sensors/actuators will enable a vast range of new habitat studies via continuous monitoring techniques. The data from such a network will need to be filtered and partially analyzed within the network e.g. seismic sensors could trigger data-intensive assets such as cameras. The proposed technology offers the chance for programmed observation, triggered response with specified patterns, and automatically recorded and reported responses. Such capabilities require the development of robust, adaptive techniques for coordinating across distributed and heterogeneous sensor/actuator nodes, many of which may be wireless and energy-limited.

Fundamental technological advances are needed to enable adaptive, programmable multi-modal networks to identify indicators of interest and use those to trigger analysis, correlation, and recording of events. Moreover, current techniques will not scale to very large numbers of wireless nodes and do not make effective use of multiple sensor modalities. To realize this goal we are developing and planning to deploy unique and innovative capabilities at the James Reserve in Southern California. Three, multi-node monitoring grids (25-100 nodes per grid) will be implemented for fixed view multimedia and environmental sensor data loggers (using wireless technologies and solar power, and ultimately capable of limited mobility, unique observation scales, proximity detection, and environmental ruggedness). We will develop and implement coordinated actuation to support experiments such as triggered emission and recording of acoustic signals from target species. Multiple perspective monitoring will be integrated through the addition of tower-based video cameras for coordinated hyper-stereoptical mapping (3D) of canopy topology and volume, monitoring of seasonal phenology of overstory tree species, and mid-ground level vegetation within the monitoring grids. Mobile nodes will also be integrated, such as an all-terrain robot for remote viewing, high resolution dimensional imaging, and "gap filling data collection" within each monitoring grid. In the long term we will incorporate tagged-animals into the system through the use of micro-RFID tags. All of these capabilities will require application of self-configuring and energy-conserving algorithms and protocols to achieve ad hoc, wireless system deployment and operation in uncontrollable environmental conditions.

This network will allow us to develop scalable techniques for non-destructive, multi-scale spatio-temporal sampling and biocomplexity data visualizations, thus enabling the rapid and low-cost mapping of new dynamic scales of species diversity, ecosystem structure, and environmental change. The facilities will provide on-site and Internet-based opportunities for graduate students and faculty to utilize these new tools, including training and research application consulting. This technology promises great opportunities in education and research alike.

In the following sections we describe two critical system building blocks needed to realize long-lived wireless sensor networks, and then present the details of our experimental platform.

# 4. TWO SYSTEM BUILDING BLOCKS FOR LOW DUTY-CYCLE OPERATION

While many aspects of deployed sensor networks will be specific to particular applications, we have begun to identify key building blocks that will enable a wide range of applications. This section discusses a "Frisbee" model for low-duty-cycle operation and techniques for time synchronization.

## 4.1 The "Frisbee" Model

One of the most important design goals of sensor network applications is minimizing the power used. Energy is a precious resource in sensor networks because there is a finite amount available; every use of it directly affects the lifetime of the network [Kaiser-Pottie00]. This makes energy-efficiency critical.

In networks where "interesting" events happen infrequently, an effective way of saving power is to turn nodes off when they are not needed. In an ideal but unattainable world, a sensor network is completely asleep during the long lulls in activity. When something does happen, only a limited zone of the network that is close to the event is kept in its fully active state. The active zone should be centered at the current location of a target phenomenon that is being tracked; and, of course, the zone should move through the network along with the target. Nodes that are not within sensing range of the event are outside of the zone, and therefore do not waste energy on data acquisition or housekeeping tasks such as maintaining time synchronization. Optimally, the zone should move such that a phenomenon of interest is always kept inside of the zone. The zone might be circular, with its radius proportional to the speed of the object. We call this model of a constantly moving circular zone the "Frisbee" model.

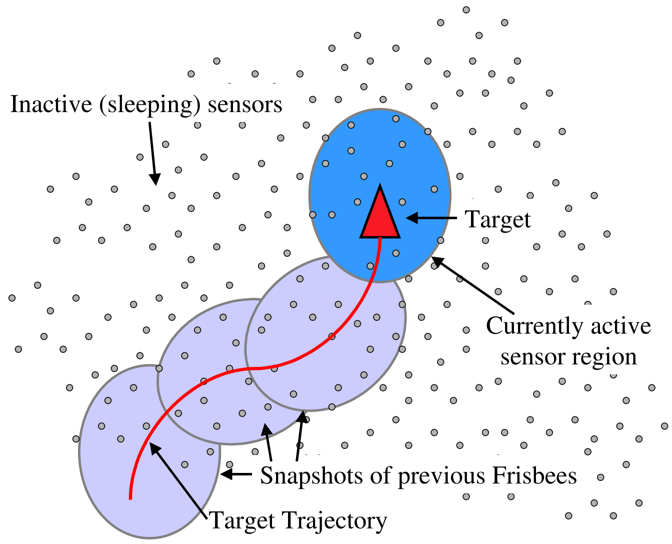


**Figure 1: The "Frisbee" model. The triangle represents the target being tracked by the network. The shaded regions are the sets of sensors awakened over time.**

To implement an approximation of the optimal Frisbee, two essential components required: (1) A low-power operating mode with wakeup: some sensors must always be vigilant; others must have a way of saving power such that they can be awakened by an external stimulus; and (2) Definition of the Frisbee boundary: sensors must use localized algorithms so that nodes can autonomously decide whether or not they are part of the Frisbee; i.e., entering or leaving it.

### 4.1.1 *Power Savings with Wakeup*

At the core of our idea is the fact that sensor nodes can be designed with a "power-saving" mode in which they are asleep. This sleep mode must require less power than a mode in which nodes are maintaining active vigilance. This mode is distinguished from simply being off because it must be possible for an external stimulus to "wake up" the nodes, bringing them from their low-power mode into a vigilant mode. The nodes should be able to generate this stimulus so that they can awaken their peers.

Most nodes should be asleep most of the time. Some nodes will remain awake all the time to serve as "sentries" – performing tasks such as accepting tasking instructions from users and looking for a potential event of interest. When a sentry senses a nearby event, it wakes up nodes in its vicinity, allowing the group to acquire data with finer granularity or a wider range of modalities than would be possible with a single (sentry) sensor.

As the target moves, nodes will send further wakeup signals to other nodes in the target's direction of motion. We envision a "wakeup wavefront" that precedes the phenomenon being tracked. As the target moves beyond the useful range of sensor nodes, those nodes go back to sleep.

An interesting observation is that by increasing the density of nodes we may be able to guarantee that enough nodes will be listening soon enough to achieve low latency wakeup, even with relatively low duty cycle wakeup at the level of individual nodes. On detection we might wake up every node that comes on cycle, so that over the course of an entire cycle, gradually all the nodes in the vicinity would be wakened. In a sense we are exploiting parallelism, by using larger numbers of low duty cycle (low power) nodes, instead of smaller numbers of higher power nodes.

### *4.1.2 Localized Algorithms for Defining the Frisbee Boundary*

A central theme pervading the design of all sensor networks is a fully distributed, decentralized design. This naturally extends to the algorithms that we plan to develop for defining the Frisbee boundary. That is to say, each node will autonomously decide whether or not it is "in the Frisbee" – there is no central controller that doles out instructions to nodes.

One example of a simple distributed algorithm for this is as follows. Nodes that have recently detected the target wake up all other nodes within a fixed radius. If possible, neighboring nodes that are already awake should be queried so that the target's speed and direction can be determined, in order to better shape and direct the "wakeup wavefront." After a certain amount of time without any detected activity, the nodes time out and return to their previous sleep state.

The design of this algorithm will also incorporate the idea of **adaptive fidelity**. In motion tracking, ever-denser distributions of sensors will usually lead to increasingly precise tracking results. However, if a particular application considers network lifetime to be more important than tracking precision, it is possible to adjust this by only waking up, say, one-half or one-quarter of the nodes that fall physically within the Frisbee. Another potential variation is that sensors closer to the center of the Frisbee are awakened with a higher density than nodes near the edges.

With a framework such as the above in place, each application will be able to tune its desired precision and commensurate energy expenditure. Applications can define what percentage of the sensor nodes within a Frisbee should be awakened every time the Frisbee moves into a new region. The decision of which exact nodes should actually be awakened is, again, a decentralized decision, likely to be made through randomization or some other technique.

## 4.2 Time Synchronization

Time synchronization is a building block critical to many aspects of a distributed sensor system. For example, it is critical to a common sensor network feature: data aggregation. Due to the high energy cost of communication compared to computation [Kaiser-Pottie00], local processing, summarization, and aggregation of data is often employed in order to minimize the size and frequency of transmissions. Suppression of duplicate notifications of the same event from a group of nearby sensors can result in significant energy savings [Intanago00]. To recognize duplicates, events must be timestamped with a precision on the same order as the event frequency. Correct time synchronization can also lead to energy savings in systems that use TDMA or other types of scheduled wake-ups; the size of the guard bands used to ensure rendezvous in the face of clock skew is inversely proportional to the synchronization precision.

For an application such as ours, which involves the detection of the speed and direction of phenomena such as tagged animals, time synchronization is critical — a set of distributed sensors must share a common time coordinate in order to integrate a series of proximity detections into a determination of speed and direction. Important variables that will vary with the application are the required precision of the time synchronization, and (closely related) the required frequency with which the sensors must be able to determine a "fix" on the tracked object. Both of these are informed by the expected nominal speed of tracked objects. Another factor affecting these parameters is the geographic density of sensors relative to both the tracked object's speed and the effective range of the sensors.

Maintaining synchronized time *energy-efficiently* among a large cluster of nodes that may often be off is a challenge not undertaken by conventional wired networks. The needs of a time coordinate and constraints on a synchronization system in a distributed sensor network vary along numerous axes: precision, scope, lifetime, cost, form factor, and so on. A variety of methods methods for establishing time synchronization are available across these same axes.

The simplest is the distinction between global and local time. A global timebase is one that is shared by every node in the network, or that exists completely independent of the sensor network. By some metrics, these are the best solutions because they eliminate accumulated error that invariably comes from distributing time hop-by-hop within a network. A single time signal broadcast to all nodes can go a long way towards reducing time-sync error. Existing time distribution infrastructures such as the WWVB time signal or GPS-time may be used for this purpose. We are investigating the use of various low-power devices that receive these time signals and can be integrated with our sensor node hardware.

While the idea of using universal time is attractive, it also does have a number of drawbacks. The hardware required to receive special time signals consumes energy and physical space, both of which are quite limited in tags. Adding a new piece of specialized hardware will always be more resource-intensive than a solution that uses existing communication capabilities of a sensor node. Moreover, universal time by definition depends on an existing time distribution infrastructure, which is often unavailable. WWVB works only in the continental US; no infrastructure is available underwater, inside of many structures, or on Mars.

For these reasons, it is also wise to consider a peer-to-peer time distribution strategy. One example is that employed by NTP, the Network Time Protocol [Mills96]. NTP can be used to establish a federation of synchronized nodes, even if the federation has no external time source. An NTP-like algorithm operates over the existing communications infrastructure of nodes; peers repeatedly poll each other for their current time in an effort to establish their relative bias and skew. Of course, if some NTP nodes have access to an external time source, a hybrid distribution algorithm is possible.

# 5. TIERED ARCHITECTURE

Although Moore's law predicts that hardware for sensor networks will inexorably become smaller, cheaper, and more powerful, technological advances will never prevent the need to make tradeoffs. Even as our notions of metrics such as "fast" and "small" evolve, there will always be compromises: nodes will need to be faster *or* more energy-efficient, smaller *or* more capable, cheaper *or* more durable.

Instead of choosing a single hardware platform that makes a particular set of compromises, we believe an effective design is one that uses a tiered platform consisting of a heterogeneous collection of hardware. Larger, faster, and more expensive hardware ("sensors") can be used more effectively by also using smaller, cheaper, and more limited nodes ("tags"). An analogy can be made to the memory hierarchy commonly found in desktop computer systems. CPUs typically have extremely expensive, fast on-chip cache, backed by slower but larger L2 cache, main memory, and ultimately on-disk swap space. This organization, combined with a tendency in computation for locality of reference, results in a memory system that appears to be as large and as cheap (per-byte) as the swap space, but as fast as the on-chip cache memory. In sensor networks, where localized algorithms are a primary design goal, similar benefits can be realized by creating the network from a spectrum of hardware ranging from small, cheap, and numerous, to large, expensive, and powerful.

The smaller "tag" devices will trade functionality and flexibility for smaller form factor and power. Alone, they would not be adequate to support our sensor network application. However, in conjunction with more endowed nodes, they significantly enhance the network's capabilities. There are many possible advantages to augmenting sensor nodes with small-form-factor tags, including: (1) Density: Tags, by definition, can be significantly lower cost and therefore can be deployed in larger numbers, more densely, than larger, higher capacity sensor nodes, (2) Longevity: Tags can be significantly lower power and therefore can be deployed for longer periods of time, or at higher duty cycles, than larger, higher capacity sensor nodes, particularly if we are able to exploit higher density, and (3) Form factor: Tags are smaller and therefore can be (a) more easily and unobtrusively attached to a wider variety of targets (e.g., for tracking, condition based maintenance, and other logging applications), and (b) deployed with high density.

# 6. EXPERIMENTAL PLATFORMS

Our initial testbed consisted of 5 Toshiba Libretto 50CT laptops running RedHat 6.0 with radiometrix transceivers [RPC]. After our initial experiences with those laptops, we gradually clarified our requirements for the testbed:

- Small in Dimensions.
- Low-Power (long lifetime).
- Moderate Computing Resources
- Flexible I/O capability.
- Well-supported Operating System.

At the same time we realized that one size did not fit all, and that we would need to work in an heterogeneous environment with nodes of different capabilities, as described in the previous section.

In this section we describe the hardware and software developed for our experimental platforms.

## 6.1 Hardware platform

### *6.1.1 PC104*

The PC104 nodes are our "high end" nodes of our tiered sensor architecture. We chose to use off-shelf PC-104-based products to replace the Librettos. PC-104 is a well supported standard [104] and compatible with desktop PCs. They can be designed for low power applications and equipped with processors ranging from 386 to Pentium II and from memory ranging from 0MB RAM to 64MB and more. There is a full spectrum of PC-104 peripheral devices including digital I/O, sensors, actuators, most of which are compatible. Before settling on the PC-104, we also considered the following alternatives:

- Laptops. Those real PCs are expensive, and have unnecessary devices for our testbed such as LCD display, IDE Hard drive, etc., each of which consume significant power.
- Palm [PAL] and other PDAs. Even though they have a compact form factor, they have inadequate computing power for our applications. Another drawback is that they usually use vendor-specific operating systems, applications and devices, and are thus difficult to extend with new hardware and software capabilities. They also have often-unnecessary and power-hungry displays.
- TINI [TIN] and ucsimm [UCS]. These small devices are promising platforms, small and cheap with support for open source OS (UClinux for ucsimm) or standardized virtual machine (Java for TINI). However, they are still in their preliminary stage and do not provide the full functionality of the Linux development environment.

The final factor that led us to PC-104 products is their ability to support almost all PC software. We chose Linux as our operating system because it is one of the most widely supported open source operating systems. We spent time evaluating different distributions or distribution building tools such as LEX, and CCLinux [CCL]. However, we decided to use our homebrew distribution based on a 2.2.12 Kernel and glibc2.1. It also contains some utility programs for management and configuration. It is binary-compatible to RedHat 6.1, thus we can develop and debug our applications on desktops/laptops, and later move to our PC-104 based testbed easily.

We are currently using an Advanced Digital Logic MSM486SN16 (Figure 2). Its features include: (1) AMD ElanSC400, 3.3V, 66MHz CPU, (2) 16MB RAM, 8kBytes (1st Level) Cache, (3) Hard Disk (E-IDE), (4) Floppy Disk Interface (3.5"), (5) COM1(optional SIR IrDA mode), COM2, COM3 and LPT1, LCD-Character Interface, Key Matrix Scanner, (6) RTC with Battery Back-up,

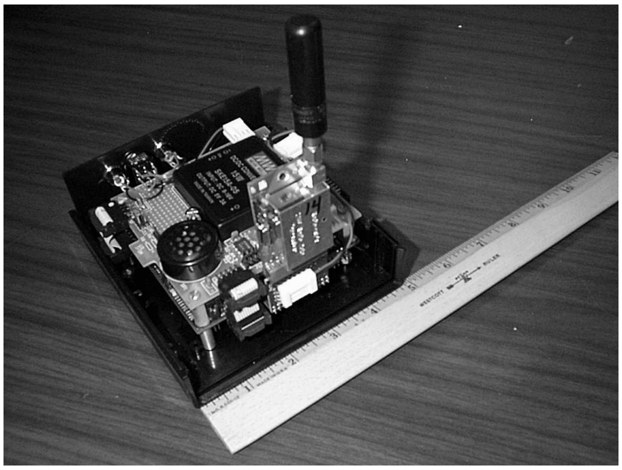

**Figure 2: Our prototype PC104-based sensor node.**

Watchdog with Power Failure Detection, Power Management (Suspend/Sleep), (7) Form Factor: PC/104 (3.6" x 3.8" x 0.6"), (8) BUS: PC/104 (ISA), (9) MSFLASH16 from ADL. 16MB IDE Flash Disk, and (10) Power: 5VDC only, 600mA/3W.

Most of PC104 devices require a strict 5V power supply (as little as 5.25V will cause damange to the MSM486SN16). We therefore designed a PCB power board for MSM486SN16, including: (1) Power Supply, 9-18V to 5V converter, (2) Infrared transceiver for IrDA or Optical Serial Asynchronous Communication, (3) Speaker Amplifier, and (4) Prototype Area. The size of the board and its standoff configuration are the same as PC104 boards for easy stacking.

### *6.1.2 Radio Subsystem*

We are currently using a Radio Packet Controller [RPC] at 418Mhz. We have a small adaptor designed to attach the RPC module to the PC104 system. We wrote a kernel mode driver to use the RPC via the parallel port (see software section). The next generation of this hardware will use an RF Monolithics radio instead and will provide variable power transmission and signal strength measurement.

### *6.1.3 Tags*

Tags are untethered devices that have a small enough form factor to be easily attached to objects that are of human scale. We term this form-factor "velcroable," meaning that the device is small enough and light enough to be attached to another object using Velcro brand fasteners.

The scale of tags puts them into an approximate one-to-one correspondence with human-scale objects, such as furniture, shipping packages, laptop computers, and specific regions in a room on a scale of humans, e.g. a 2 square meter patch of the upper left wall. The tags are untethered so that they can recede into the background, performing their tasks with minimal disruption of the environment and minimal infrastructure requirements. This untethered requirement bounds the capability of a tag: the energy available to it is limited by its form factor while its capabilities are bounded by the energy cost of its task.

In our system, tags are implemented with a modular architecture. This is desirable for development purposes, and it leaves room to improve efficiency through integration. The modules that compose a tag are a loosely coupled system connected by a bus. Each module has local computational capability, implemented by a microcontroller that is interfaced to local resources such as sensors, actuators, communications hardware, or memory. In many respects, each module can operate standalone, only waking up another module when the resources that module controls are needed for its task.

The requirements of the bus are relaxed to accommodate long duty cycles. In our architecture, a master module stays on all the time at very low clock rates and acts as a central point of coordination. Other modules may operate on fixed or variable duty cycles. For example, a sensor module might be operating in a low-power vigilant mode, and when a particular condition is detected, it wakes up the RF board and reports the event. The RF board may then test the channel to see if any other tags are reporting, before reporting an event itself.

We are currently developing several components of the tag platform:

- The master module. This module hosts a number of services that most tag systems will need, including a real time clock, an EEPROM memory, and three UARTs for communication to external devices such as a larger PC-104 based system or an external sensor module. The master module can operate at a very slow clock rate for power savings, or at a high rate when precise coordination is required. It also has a pair of interrupt lines connecting to each of the other modules in the system. During times of high power expenditure, these interrupt lines are used for precise timing and synchronization, while during times of minimal power expenditure, they are used to wake up a module so that it can prepare to receive a message on the serial data bus.
- Radio controller module. This module provides a radio network interface. An RFM baseband radio is incorporated into this module driven by a micro-controller. The micro-controller implements a MAC layer as well as other protocol components. A large (32K) SRAM is also included on the radio module, enabling the storage of state pertaining to higher layer protocols. The simplest applications envisioned, such as a simple beaconing or sensor reporting system, may be implemented directly in the radio controller, eliminating the need for other external components.
- Sensor interface module. This module has a variety of on-board sensors and several actuators, interfaced to a micro-controller. The micro-controller can perform simple analysis of the sensor data as it streams into the system, and can watch for certain characteristic events. It can also record time series, possibly sending them over the bus to the SRAM in the radio module or storing them into the master module EEPROM. The sensor module will also feature a remote low-power wakeup device that can wake up the processor when a certain frequency of sound is detected. This enables most of the devices to be completely off, waiting to be waked up by one of the few left on when an event occurs.

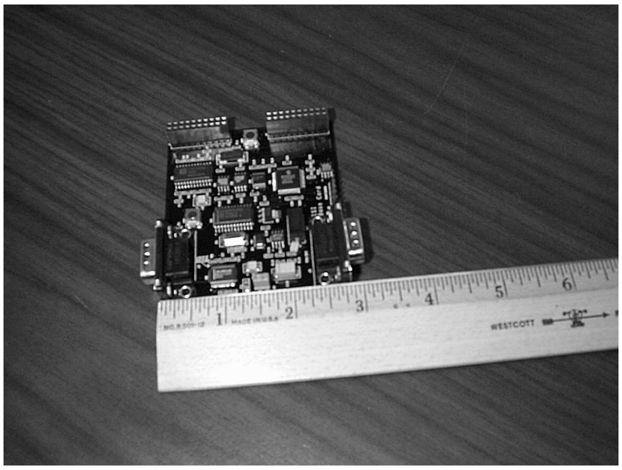

**Figure 3: Prototype radio controller module for use with our "tag" platform.**

- CPU or DSP module. Some applications may require more computational power and more memory when an event occurs. To handle these cases, a computational module can be developed. This module would be invoked when necessary to perform a complex calculation and then resume sleeping.

The tag implementation favors COTS, simplicity of design and flexibility, over optimization of the hardware design. Our tags require a modular architecture. Each module has an inexpensive processor and connects to other modules via a system bus. Modules may be powered down via software controlled switches on each board. None of the modules require complex design. However, modularity would allow for other types of sensors to be added as the outcome of other research projects now or in the future.

We envision that the following modules will be needed in order to get a useful experimental platform:

- Power supply/host interface module. This module provides a real time clock, regulated power to the bus and one or more serial interfaces for debugging/logging output, or for interfacing to a host processor such as a SENSIT node or PC.
- Radio controller module. This module provides a radio network interface to other modules in the system. An RFM baseband radio is incorporated into this module with a micro-controller to drive it. The simplest applications envisioned may be implemented directly in the radio controller, eliminating the need for other external components.
- Sensor sampling and storage module. The simplest form of this module samples from an A/D converter and stores the resulting time series in a serial memory. After sampling, the data can be loaded over the bus into another module to process it. A more powerful version with a faster CPU will be developed as part of the development of an acoustic ranging sensor based on ultrasound.

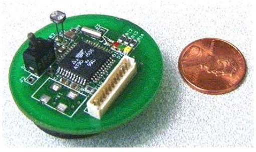

**Figure 4: Pister's "COTS Mote", developed at U.C. Berkeley [Pister99]**

- Remote wakeup module. This module provides a way for a node to be "awakened" from a very low-power sleep state. For such a module to be useful, the wakeup module must consume far less energy than the node it is awakening. A trivial example of a wakeup module is an "on/off" button. We plan to leverage other existing technology, such as a module that can wake nodes up in response to high-energy RF pulses.

#### *6.1.4 Motes*

Motes are the smallest components of our tiered sensor architecture. Where tags are on a scale comparable to human scale, motes are much smaller and much more numerous. Motes are envisioned to be small enough to float in the air [Pister99] or to be attached in large numbers to a surface [Abelson99]. The cost of motes is intended to be small enough that they are entirely disposable. The technology to produce motes is thought to come from the area of MEMS.

Motes communicate with macro-scale objects through low power RF and through visible light. Through extensive integration, low power RF can be included on the same silicon die as the mote itself, and the antenna may serve as a "tail" that enables it to float in an air current. Another possible communication technique uses steerable corner-cube reflectors to reflect laser light shined on a collection of motes. A third communication technique uses a steerable mirror to point a laser beam. All of these options rely on MEMS technology to provide this functionality in such a small package.

Although motes the size of dust motes are still a long way off, current research is progressing using very small and simple devices that integrate a microcontroller with sensors and a radio on a small board.

We are using the mote developed by Kris Pister at UC Berkeley [Pister99], shown in Figure 4.

### 6.2 Software

#### *6.2.1 Radiometrix Device Drivers*

This package [RDD] contains a Linux device driver for the RPC (Radio Packet Controller) model of radio manufactured by Radiometrix. The RPC is a fairly low power, self-contained, short-range, plug-on radio. It has been a critical part of our testbed infrastructure for implementation and validation of directed diffusion and other algorithms.

#### *6.2.2 Emlog*

Emlog [EML] is a Linux kernel module that makes it easy to access the most recent (and only the most recent) output from a process. It works just like "tail -f" on a log file, except that the storage required never grows. This is very important for our logging and debugging facilities in embedded systems where there isn't enough memory or disk space for keeping complete log files, but the most recent debugging messages are sometimes needed (e.g., after an error is observed).

#### *6.2.3 Parapin*

Parapin [PAR] makes it easy to write C code under Linux that controls individual pins on a PC parallel port. This kind of control is very useful for electronics projects that use the PC's parallel port as a generic digital I/O interface. Parapin goes to great lengths to insulate the programmer from the somewhat complex parallel port programming interface provided by the PC hardware, making it easy to use the parallel port for digital I/O. By the same token, this abstraction also makes Parapin less useful in applications that need to actually use the parallel port as a parallel port (e.g., for talking to a printer).

## 7. CONCLUSIONS

We have described some of the system building blocks we are developing for distributed sensor networks, presented details of our experimental testbeds for wireless sensor network development, and described the habitat monitoring application for which they are being developed. Many of our system building blocks are being developed in parallel with the goal of eventual integration into a deployable system.

## 8. ACKNOWLEDGEMENTS

This work was supported by DARPA under grant No. DABT63-99-1-0011 as part of the SCADDS project, NSF grant ANI-9979457 as part of the SCOWR project, and was also made possible in part due to support from Cisco Systems.

## References

[Abelson99] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman and R. Weiss, "Amorphous Computing", MIT AI Memo 1665, August 1999.

[Colwell98] Colwell, Rita, 1998. Testimony of Dr. Rita Colwell, Director, National Science Foundation, Before the Basic Research Subcommittee, House Science Committee, Hearing on Remote Sensing as a Research and Management Tool. September 28. 1998.
http://www.nsf.gov/od/lpa/congress/rc80928.htm

[Estrin-etal.99] Deborah Estrin, Ramesh Govindan, John Heidemann and Satish Kumar "Next Century Challenges: Scalable Coordination in Sensor Networks", ACM MobiCom 99, August 99, Seattle, WA.

[Gell-Mann95] Gell-Mann, Murray. 1995. What Is Complexity? Complexity, Vol. 1, no.1 John Wiley and Sons, Inc.

[Hamilton00] Hamilton, Michael P. 2000. Hummercams, Robots, and the Virtual Reserve. Directors Notebook, James San Jacinto Mountains Reserve web site. February 6, 2000.
http://www.jamesreserve.edu/news.html

[Hamilton-Flaxman92] Hamilton, M.P. and M. Flaxman. 1992. Scientific data visualization and biological diversity: new tools for spatializing multimedia observations of species and ecosystems. Landscape and Urban Planning. 21:285-297.

[Intanago-et.al.00] Intanagonwiwat, C., Govindan, R., Estrin, D. "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks", ACM MobiCom 2000, August 00, Boston, MA.

[Keitt-etal97] Keitt, T.H., D.L. Urban, and B.T. Milne. 1997. Detecting critical scales in fragmented landscapes. Conservation Ecology [online]1(1): 4. Available from the Internet.
http://www.consecol.org/vol1/iss1/art4

[Lohr-etal95] Lohr, S. A., P. G. Connors, J. A. Stanford and J. S. Clegg. 1995. A New Horizon for Biological Field Stations and Marine Laboratories. Miscellaneous Publication 3, Rocky Mountain Biological Laboratory, Crested Butte, CO, 36 pp.

[Mataric95] Maja J. Mataric, "Issues and Approaches in the Design of Collective Autonomous Agents", Robotics and Autonomous Systems, 16(2-4), Dec 1995, pp. 321-331

[Michener-etal98] Michener, W. K., J. H. Porter and S. G. Stafford. 1998. Data and Information Management in the Ecological Sciences: A Resource Guide. LTER Network Office, University of New Mexico, Albuquerque, NM. 138 pp. ("DIMES report")
http://www.lternet.edu/ecoinformatics/guide/frame.htm

[Mills94] David L. Mills. Internet Time Synchronization: The Network Time Protocol. In Zhonghua Yang and T. Anthony Marsland, editors, *Global States and Time in Distributed Systems.* IEEE Computer Society Press, 1994.

[Pister99] J. M. Kahn, R. H. Katz and K. S. J. Pister, "Mobile Networking for Smart Dust", ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom 99), Seattle, WA, August 17-19, 1999.

[Pottie-Kaiser00] G. Pottie and W. Kaiser, "Wireless Sensor Networks", Communications of the ACM, Vol. 43, No. 5, May 2000, pp. 51-58

[Sukhatme99] Gaurav S. Sukhatme, James F. Montgomery, and Maja J. Mataric, "Design and Implementation of a Mechanically Heterogeneous Robot Group", Proceedings of Mobile Robots XIV - SPIE 99, Boston, MA

[Sukhatme00] Gaurav S. Sukhatme and Maja J. Mataric, "Embedding Robots into the Internet", Communications of the ACM, May 2000, 43(5), pp. 67-73

[Walker-Steffen97] Walker, B., and W. Steffen. 1997. An overview of the implications of global change for natural and managed terrestrial ecosystems. Conservation Ecology [online]1(2)
http://www.consecol.org/vol1/iss2/art2

## Web Page References

[RDD] Radiometrix device drivers,
http://www.circlemud.org/jelson/software/radiometrix

[EML] Emlog Software Package,
http://www.circlemud.org/jelson/software/emlog

[PAR] Parapin PC Pin Control Library
http://www.circlemud.org/jelson/software/parapin

[RPC] Radio Packet Controller, http://www.radiometrix.com

[104] PC104 Consortium, http://www.pc104.org

[PAL] 3COM Palm Pilot Devices, http://www.palm.com

[TIN] TINI: Tiny InterNet Interface, Ibutton,
http://www.ibutton.com/TINI/index.html

[UCS] Embedded Linux/Microcontroller Project,
http://www.uclinux.com/

[CCL] CCLinux: The Minimal Linux Distribution,
http://www.cosmicchaos.com/CClinux/index.shtml