Turn Off Your Radios!

Environmental Monitoring Using Power-Constrained Sensor Agents

Daniel D. Corkill
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01002
corkill@cs.umass.edu

Douglas Holzhauer[*] & Walter Koziarz
Air Force Research Laboratory
Rome, NY 13441-4505
douglas.holzhauer@sunyit.edu
Walter.Koziarz@rl.af.mil

ABSTRACT

CNAS[1] (Collaborative Network for Atmospheric Sensing) is an agent-based, power-aware sensor network for ground-level atmospheric monitoring. In many multi-agent applications, reducing message transmission is a primary objective. In CNAS, however, it's not the cost of sending messages, but *when* messages can be sent that is the driving communication constraint. CNAS agents must have their radios turned off most of the time, as even **listening** consumes significant power. Working in such *collaborative isolation* changes the character of agent interaction, as agents must have their radios turned on when others are sending messages to them. CNAS requires agent policies that can intelligently meet operational requirements while communicating only during intermittent, mutually established, communication windows.

In this paper, we describe the CNAS agents and their hardware and blackboard-system software architectures. We also relate experiences and lessons learned from a field deployment of CNAS at the 2006 PATRIOT Exercise held last July at Fort McCoy, Wisconsin, and we discuss the upcoming CNAS deployment in conjunction with the Talisman Saber Combined Exercise to be conducted May–July 2007 in Australia. We conclude with an overview of current CNAS research that is exploring the addition of a rollable solar panel to each sensor agent that allows its battery reserves to grow (up to full capacity) when sunlight is available. Replenishable power reserves can support unlimited operational lifetimes, but activity decisions become more complex as each agent now must consider how much additional power may become available and when.

1. ATMOSPHERIC MONITORING

The U.S. Air Force is interested in sensor networks for ground-level environmental monitoring, as detailed knowledge of local atmospheric conditions increases air drop precision and all-weather landing safety. Such networks also have application to detecting forest fires, monitoring their changing status, and informing firefighters of changing conditions that affect their strategy and safety. Similarly, detailed knowledge of local atmospheric conditions is important in managing responses to airborne hazardous materials (hazmat) incidents and in determining prudent evacuation areas and routes.

Low-level atmospheric phenomena are characteristically complex with changing spatial gradients. Because of this complexity, mathematical models based on a small number of observations can not accurately quantify important local environmental variations. At present, weather-based mission decisions use predictions made by the Air Force Weather Agency using complex large-scale models such as Mesoscale Model 5.1 (MM5).[2] Using the new Weather Research & Forecasting (WRF) model,[3] which incorporates individual observations into MM5, can increase the prediction accuracy. Currently most observations fed into MM5/WRF are acquired by satellites or by land based radar. Naturally, the closer the direct observations used as inputs to MM5/WRF are to the region of interest, the more accurate the predictions for that region will be. Even when used in combination, these large-area sensors can exhibit serious limitations when the area of interest is located in a remote isolated region. Cloud cover can mask the lower elevation weather parameters, and the curvature of the earth quickly restricts ground radars from observing lower portions of the troposphere. In the case of mountainous terrain, large geographical changes over small distances can prevent even the best models from accurately determining local weather conditions [3].

Large, battery-powered, ad hoc sensor networks can provide the high-accuracy environmental data needed in these application settings. Work by both DARPA and AFRL's Sensors Directorate is leading to sensor nodes that are sufficiently rugged that they can be air dropped into regions of interest[4] and that are able to selectively control their battery-power expenditures to provide monitoring services over extended time periods. Self-organizing, air-dropped sensor networks will enable the collection of detailed envi-

The UMass portion of this work is supported by the AFRL "Advanced Computing Architecture" program, under contract FA8750-05-1-0039. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views contained in this paper are the authors.'

[*]Doug Holzhauer is now retired and teaching at SUNYIT.
[1]Pronounced "see-nas."

[2]http://www.mmm.ucar.edu/mm5/
[3]http://www.wrf-model.org/
[4]The current tactical weather station used by the Air Force, the AN-TMQ-53, cannot be air dropped because of its cost and packaging.

ronmental data from regions that were previously closed to ground-level monitoring.

Air-dropping atmospheric monitoring nodes introduces additional issues. Normally when a weather station is positioned, meteorologists use their understanding of geography and meteorology to optimize the location for weather measurements. Precise placement is not possible, when sensor nodes are air dropped. (However, research and development into maneuverable air-drop delivery systems are underway.) For the time being, though, even a marginal location for an air-dropped weather station can not be assured. To compensate, additional sensors may be deployed and their observations weighted as to quality.

Environmental monitoring networks may also include many different types of sensors, and individual sensor capabilities may need to be dynamically adjusted (in terms of what aspects of the environment are sensed, the precision, power, and usage frequency of sensing, and the amount of local processing done by each sensor node before transmitting information). Information processing in the network may require the integration/fusing of information coming from heterogeneous and geographically distant sensors. Additionally, sensor usage and parameters may need to be adjusted in real-time as the network tracks phenomena moving through the environment and as the power and communication resources available to the sensor nodes change. Battery-powered sensor nodes need to spend their limited power wisely in collectively performing their best in achieving overall sensor-network goals.

In addition to this real-time, operational agility, the design of the sensor network should allow the software approaches and algorithms of nodes to be changed, improved, and extended throughout the operational lifetime of the network. We should expect from the outset that new and improved components and software techniques will be developed over time and added to the system. The underlying design of the sensor network should be able to adapt to such new capabilities and be able to manage their use effectively.

Sensor Agents

A central challenge in building effective environmental-monitoring sensor networks is coordinating the use of critical resources (including sensors, processing, communication, and power) to best achieve conflicting mission, organizational, and sensing goals [2]. In resource-constrained settings typified by sensor networks, the activity decisions of “who,” “what,” “when,” “where,” and “with whom” must involve an overall awareness of organizational and operational capabilities and goals, the state of activities and resources in the network, and the time-critical nature of activities and sensor data. This requires that every sensor node understand that it is part of a larger organization and that it may need to satisfy more global goals at the expense of its own local goals.

The need for autonomous and self-aware sensor nodes is a natural fit for employing multi-agent system (MAS) technology. Agent-based sensor networks are part of an important class of MAS applications in which issues of organizational structuring, coordination, collaboration, and distributed, real-time resource allocation are critical for success. Simply put: sensor agents must do more than react to their local situation—they must collaboratively determine what they should be doing, when they are doing them, and why.

2. CNAS

CNAS (Collaborative Network for Atmospheric Sensing) is an experimental, agent-based, power-aware sensor network for ground-level atmospheric monitoring. It is intended as a research and demonstration tool for conducting realistic explorations of the advantages and limitations of agent-based environmental monitoring using hardware capabilities that are likely to become cost-effective for production deployments in the next few years.

CNAS has the following major hardware and operational characteristics:

- Sensor agents have on-board GPS capabilities for obtaining location and time data.
- The distance separating sensor agents is near the limit of their wireless communication range. Alternate message-route options are relatively sparse, and much communication is multi-hop.
- The WiFi adapter used at each agent is the component with the largest power-expenditure rate, by far. So, although the power to the central processor and peripheral modules at each sensor agent can be controlled independently, such power savings are secondary to the savings achieved by having the WiFi turned off as much as possible.
- Sensor agents obtain and process local-environment readings once every second (even when their WiFi is powered off).
- Summaries of the sensor-agent readings taken over geographic region of interests are aggregated and maintained by sensor agents performing a regional “cluster head” service.
- Mobile “console nodes” may enter and leave the CNAS monitoring area, obtaining regional summaries and individual sensor-agent data and changing the tasking and policies of CNAS.
- The processing, memory, and storage capabilities available at each sensor agent are relatively powerful.

We use a combination of blackboard-system and MAS techniques in our CNAS sensor agents. Blackboard systems [4, 5] are proficient in supporting indirect and anonymous collaboration among software entities and in exploiting temporal decoupling of entity interactions in order to obtain maximum flexibility in coordinating activities. MAS researchers, on the other hand, have developed effective techniques for operating in highly distributed, dynamic settings and for coordinating local, autonomous activity decisions. These capabilities are all highly valuable assets in designing an effective architecture for agile, resource-aware sensor network agents. Nevertheless, applying blackboard and MAS approaches in concert to the sensor-net domain is a novel aspect of the CNAS effort.

The agent-level design of CNAS was driven by the hardware and support software that had been designated for this effort. Therefore, from a MAS perspective, the hardware and support-software characteristics and capabilities

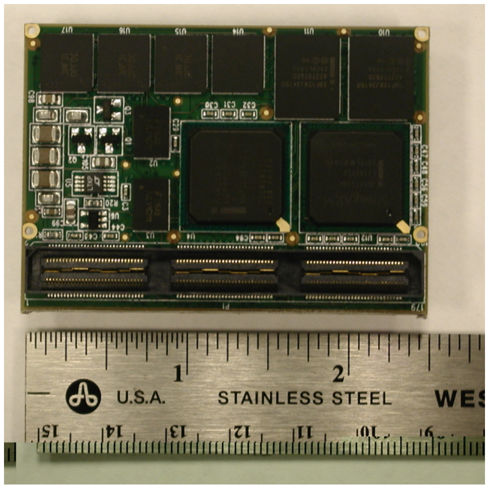

Figure 1: PASTA PXA255 CPU Module

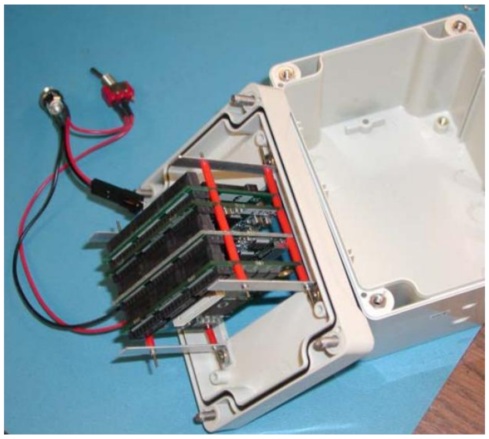

Figure 2: PASTA/Crossbow Sensor Agent

are pre-established and unchangeable. Our challenge was to develop an effective agent-based monitoring network using the specified hardware and operating-system software.

Sensor-agent hardware

Each sensor agent is built around ISI's PASTA (Power-Aware Sensing, Tracking, and Analysis) microsensor platform [8][5] and its Intel PXA255-based CPU (Figure 1). Unlike traditional hub-and-spoke sensor-node architectures that have peripherals clustered around a central processor, the PASTA uses a distributed-peer model that can decouple processing from peripheral operation. In the hub-and-spoke architecture, the central processor must be continually active to broker peripheral operations, and this power consumption represents the lowest possible rate of total system-power expenditure.

In the decoupled, distributed model used in the PASTA, the central processor and peripheral modules operate autonomously, and each can be powered independently. Higher-performance processing can be made available when needed, but low average-system-power expenditure can be achieved by operating in extremely low-power modes with only essential modules active whenever possible. The PASTA's central processor is running a customized 2.4.19 Linux kernel, even though this places Linux in the unconventional role of a peer module rather than controlling a central processor.

[5]http://pasta.east.isi.edu/

Figure 3: Crossbow MTS420CA

In addition to the PASTA, each CNAS sensor node (Figure 2) is equipped with a Crossbow MTS420CA sensor board (Figure 3) providing:

- Intersema MS5534AM barometric pressure sensor
- TAOS TSL2550D ambient light sensor
- Sensirion SHT11 relative humidity/temperature sensor
- Leadtek GPS-9546 GPS module (SiRFstar IIe/LP chipset)
- Analog Devices ADXL202JE dual-axis accelerometer[6]

A Netgear MA111 Wireless adapter, connected to the PASTA's USB interface, provides standard IEEE 802.11b wireless communication, which achieves the 1–2km range required by CNAS.

Power expenditure & communication

The 12-volt battery used at each sensor agent provides approximately 12,000mA-hours of power. The IEEE 802.11b USB adapter is the component with the largest power-expenditure rate, by far, on the sensor agent. The adapter draws at a rate of 250mA when powered on, which would consume all battery power, operating alone, in 48 hours. Since we were constrained to use 802.11b in CNAS, the only solution was to have each agent turn off its WiFi adapter most of the time. These radio-power decisions cannot be made unilaterally, as other nodes need to know when their transmissions will be receivable, and nodes may be acting as forwarders in multi-hop transmissions that don't pertain to them.

Communication policies and routing protocols have been designed especially for energy saving in wireless sensor networks. (Akkaya and Younis provide a recent survey [1].) Most of these policies assume sensors are stationary, as is the case with CNAS. Some assume mobile sinks, like CNAS's console nodes, and periodic reporting requirements. Unlike CNAS, where listening is as expensive as sending and agents are at the limit of their direct communication range, much of the energy-efficient routing work focuses on limiting transmission quantity and distance while assuming full-time listeners. Even in protocols such as Geographic Adaptive Fidelity (GAF) [9], where nodes are switched on and off to reduce communication-energy expenditures, a percentage of the nodes in each geographic region are always on.

[6]Not used in CNAS.

In CNAS most, if not all, sensor agents must have their radios activated at the same time—if only to provide a multi-hop route for other agents. We address this in the obvious way, by using a set of compatible time-based radio-power policies that allow nodes that may not be aware of the current policy to eventually synchronize their policy with others. Each policy consists of fixed-length communication windows that occur at regular intervals, where the windows of each policy align with one another whenever possible. The policies we are using are:[7]

hourly: A communication window occurs at the top of each hour

half-hourly: A communication window occurs every 30 minutes, starting at the top of each hour

quarter-hourly: A communication window occurs every 15 minutes, starting at the top of each hour

hourly-overnight-sleep The hourly policy, but without communication after the 6PM window until the 6AM window the next morning (local time)

half-hourly-overnight-sleep The half-hourly policy, but without communication after the 6PM window until the 6AM window the next morning (local time)

quarter-hourly-overnight-sleep The quarter-hourly policy, but without communication after the 6PM window until the 6AM window the next morning (local time)

The current policy can be switched at the next communication window, based on current weather trends and mission objectives. A new or rebooted node that is not aware of the current policy can be assured of communicating with others during the next daytime top-of-the-hour window, no matter which policy is in effect. Alternatively, the node can be more aggressive and try connecting at the next quarter-hour window, and at subsequent fallback windows.

Inter-node communication in CNAS uses standard TCP/IP operating over an OLSR (Optimized Link State Routing)[8] multi-hop protocol. OLSR is intended for dynamic routing under changing connectivity and propagation conditions, and where a relatively small proportion of nodes are likely to come and go at the same time. Due to the long periods of no radio power, CNAS forces OLSR to essentially reinitialize at the start of each communication window. Given this stabilization requirement, each CNAS communication window was structured as the sequence of activities shown in Figure 4. These staged activity intervals are very conservative, and provide substantial slack time for OLSR re-initialization and for coping with highly degraded communication. They also can be changed on the fly, so shorter, more aggressive, communication windows can be attempted. During a communication window, each agent uses an application-level message retransmission strategy whenever the TCP/IP-layer reports delivery failure. A prioritized store-and-retry message delivery strategy holds outgoing messages that cannot be delivered due to outage during a communication window as well as messages generated when the agent's radio is turned off.

0–60 sec	WiFi power on, OLSR stabilization
60–120 sec	node assessment, status exchange, high-priority message delivery, cluster head determination
120–140 sec	node observation transmission (to cluster head)
140–240 sec	cluster head processing, low-priority message delivery
240–300 sec	cluster observation transmission (to console & regional nodes)
300 sec	WiFi shutdown

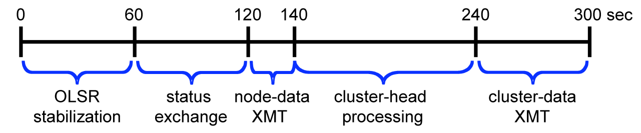

Figure 4: CNAS Communication Window

[7]An alternate set of policies was considered in which the half-hourly policies were replaced with windows occurring every 20 minutes and the quarter-hourly policies with windows occurring every 10 minutes. With significantly shorter communication windows, this alternate set, perhaps even augmented by an every-5-minute policy, might be preferable.

[8]http://olsr.org/

Software

One objective of the CNAS effort was demonstrating the feasibility of hosting a high-level language and an AI blackboard-system framework on the PASTA. After a preliminary assessment, we felt that it was indeed possible to support both Common Lisp and the GBBopen open-source blackboard-system framework[9] on the PASTA, and we began a porting effort to the PASTA for CNAS. GBBopen is written in Common Lisp and uses CLOS (the Common Lisp Object System) [6] and the Common Lisp Object System Metaobject Protocol (MOP) [7] to provide blackboard-specific object capabilities. The blending of GBBopen with Common Lisp transfers all the advantages of a rich, dynamic, reflective, and extensible programming language to blackboard-application developers. Thus, GBBopen's "programming language" includes all of Common Lisp in addition to the blackboard-system extensions provided by GBBopen.

We initially considered using a partially completed ARM port of SBCL (Steel Bank Common Lisp)[10] as the Common Lisp implementation for the PASTA. SBCL is an open-source Common Lisp implementation that supports an excellent optimizing native-code compiler. Although SBCL's compiler technology supports both RISC and Intel-class processors, the combination of a RISC instruction set with a relatively limited number of registers (more Intel like) on the ARM processor did not match any of the existing compilation models in the SBCL compiler. The need to implement a new "hybrid" compilation model for the ARM processor had delayed volunteer work on the finishing the ARM port of SBCL indefinitely.

We did not have the time or resources to invest in completing the ARM port, and were forced to abandon an SBCL strategy. Fortunately, at about this same time, sufficient MOP support for GBBopen was completed for another open-source Common Lisp implementation, CLISP.[11] CLISP was already ported to ARM processors, so with the added MOP support in CLISP 2.34, we had a viable Common Lisp implementation for hosting GBBopen on the PASTA.

[9]http://GBBopen.org/

[10]http://sbcl.sourceforge.net/

[11]http://clisp.cons.org/

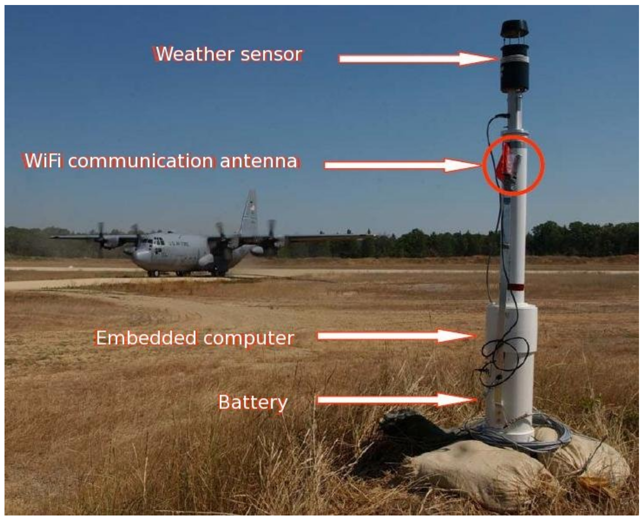

Figure 5: A TACMET-Augmented Sensor (at the PATRIOT 2006 exercise)

An important advantage of CLISP is that it is structured as a small C-based kernel that operates in conjunction with a platform-independent bytecode compiler and virtual-machine executer. A major disadvantage of CLISP, however, is that it does not support Lisp multiprocessing (process threads), which complicates real-time event processing. However, the small and portable C-based kernel and compact bytecode executer were well suited to the memory space available on the PASTA.

We were able to make use of the Debian ARM packaging of CLISP 2.34 (performed by Will Newton). The Debian package allowed us to bypass cross-compiling the CLISP kernel using an ARM cross-compilation toolchain running on an Intel x86 host and bootstrapping the rest of the build directly on the PASTA (which would have been a very painful process). Using the Debian package did introduce some problems, however. Differences between Debian and the PASTA's TinyOS Linux distribution required forced bypassing of package dependencies and introduced incompatibilities in several shared libraries. The latter resulted in some degradation in CLISP's memory management and garbage-collection performance on the PASTA.

The Pasta running basic Linux system processes and an `ssh` remote-login session has approximately 34.3MB of free memory space (out of a total of 64MB). CLISP consumes slightly more than 2MB, and GBBopen uses another 2MB. This brings the free memory down to about 30MB that is available for blackboard objects, knowledge sources (KSs), and sensor data—a reasonable amount for performing sensor-agent processing.

Node types & roles

A CNAS network can contain four different "types" of agent-based nodes:

Sensor Agents: A basic CNAS sensor agent consists of: the PASTA stack and Crossbow, a USB WiFi adapter, and a 12V battery, all packaged in a PVC housing that positions the wireless antenna and sensors 4.25' above ground level. The node packaging is intentionally large to allow easy access during testing and evaluation.

TACMET-Augmented Sensor Agents: A TACMET-augmented sensor agent is a basic CNAS sensor agent (as above) that also includes a Climatronics TACMET II 102254 weather sensor (see Figure 5). The TACMET II provides a temperature sensor, a fast-response, capacitive relative humidity sensor, a barometric pressure sensor, a flux gate compass, and a folded-path, low-power sonic anemometer. Wind speed, wind direction (resolved to magnetic North with the flux gate compass), temperature, and relative humidity readings are provided to the PASTA over an RS-232C serial connection once every second. Power to the TACMET cannot be controlled by software, and the TACMET is powered by a separate (unmanaged) 12V battery. TACMET II provides wind speed and direction readings that are not available from the Crossbow sensors, as well as corroborating measurements for temperature, humidity, and pressure. Unlike the Crossbow, TACMET II measurements have been certified by the Air Force, and having duplicate measurements allows calibration of the Crossbow readings.

Console Nodes: A console node is a laptop or handheld computer that, upon entry into the CNAS-network area, can obtain observation data from the network and display that data graphically. Unlike sensor and TACMET nodes, console nodes do not turn their wireless on and off. An authorized user at a console node can change network objectives and policies, transition the network into continuous-communication ("debugging") mode, and perform detailed inspection of the data and activities at individual sensor agents. Of course, unless the network is in continuous-communication mode, these activities can only be performed during communication windows when sensor agents have their radios turned on.

Regional Node: The regional node is a console node that is also connected to an external network, such as the Internet. When a regional node is available in the CNAS network, observational data and summaries can be made available outside the CNAS monitoring region, and authorized remote users can re-task CNAS objectives, perform detailed inspection of the data and activities at an individual sensor agent, and even update sensor-agent software.

Each CNAS sensor agent performs basic sensing activities, obtaining atmospheric readings once each second. Following Air Force meteorological practice, the following summary readings are computed from the 1-second readings and saved every five minutes:

- temperature (5-minute average)
- dew point (5-minute average)
- pressure (last reading)
- altimeter (last reading)
- wind-u-component (2-minute average, TACMET agents)
- wind-v-component (2-minute average, TACMET agents)

All unsent 5-minute summary observations are transmitted to the node acting as the *cluster head* (discussed shortly) for the agent during the next communication window.

In addition to the 5-minute summary observations, each sensor node performs an interval-based compression of the raw sensor observations. These compressed 1-second readings and the 5-minute summary observations are held by the agent for a user-specified period (typically for many days).

Each sensor agent performs saturation-vapor-pressure, humidity-to-dew-point, pressure-to-altimeter, millibars-to-inches, and wind-meter-to-knot computations as needed. The PASTA does not include floating-point hardware, so these computations are performed by software emulation.

Cluster heads

In addition to its sensor duties, a regular sensor or TACMET agent can also assume the role of cluster head. A *cluster* in CNAS is a grouping of sensor nodes located in a geographic region of interest. Non-overlapping cluster regions are user-defined and are communicated to all nodes from a console or regional node. Each node determines its cluster membership at start up, based upon its location and the user-specified cluster regions it receives when it first makes contact with another node in the network. Through an information-spreading process, the identity and locations of all nodes in a node's cluster becomes known. Given this cluster-membership information and globally established criteria, each agent computes a total preference ordering over all the agents in its cluster for assuming the cluster-head role.

During the initial phase of every communication window, each sensor agent determines the agent that is the most preferred cluster-head among all the agents that are alive and communicating in its cluster. If the agent is not assuming the cluster-head role, it transmits its 5-minute observations to the cluster head. If it is the cluster head, it accumulates the observations received from the other agents in its cluster, creating 5-minute cluster summary observations.[12] As the end of the communication window draws near (at the 4-minute mark in our conservative policy), the cluster head transmits the cluster summaries to all console and regional nodes that are active.

The cluster-head determination policy takes advantage of the OLSR-layer routing information to determine what nodes can receive messages from the agent. Should a cluster become bifurcated, separate cluster heads will be selected for each cluster fragment. When connectivity is re-established, these cluster heads provide summaries for the same cluster to console and regional nodes, where they can be combined into a single cluster summary. Furthermore, the most preferred sensor agent will again become the sole head of the reunited cluster.

3. CNAS DEPLOYMENT: PATRIOT 2006

CNAS was field tested at the 2006 PATRIOT Exercise held last July at Fort McCoy, Wisconsin. Over 1,600 Army and Air National Guardsmen, U.S. Air Force and Army active-duty and Reserve personnel, and soldiers and airmen from Canada, the Netherlands and the United Kingdom participated in the Exercise. Nine sensor agents and 8 TACMET-augmented sensor agents were manually positioned (not air dropped) in the area around Young Field and the Badger Drop Zone (see map, Figure 6). A telephone line at the south-eastern edge of the monitoring area was also reserved to connect a laptop-based regional node to the Internet via a dial-up modem connection.[13]

[12]Cluster summaries include the list of the individual nodes that contributed to them.
[13]The regional node was removed each night.

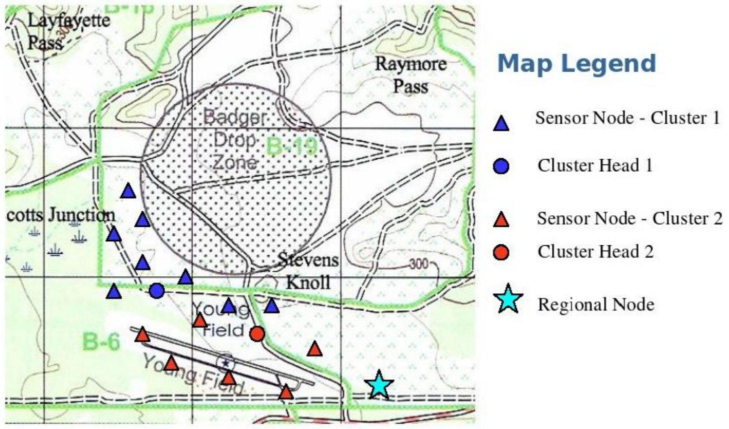

Figure 6: PATRIOT Sensor Agent Locations

Figure 7: Node 3 at the PATRIOT Exercise

Heat, humidity, line noise, and bugs

Our original plan was to demonstrate full CNAS capability at the PATRIOT Exercise. As the date of the exercise approached, firmware issues involving the Crossbow interface made its reliable operation uncertain, and we decided to deploy CNAS with the Crossbow sensors disabled. Without the Crossbow, GPS positioning and system-clock setting was lost. To compensate, a handheld GPS unit was used to determine the location of each sensor agent as it was placed, and this location and the node name (IP address) was entered into a console-node laptop. Then, as each sensor agent came on line, it obtained its location from the console node.

The PASTA does not have a hardware clock; the system clock has to be set every time the PASTA is booted. Without a GPS-obtained time, our fallback was to use the regional node as a CNAS time server and synchronize all agent clocks to it. This meant that when a sensor agent is booted, it does not have the correct time for synchronizing with CNAS network communication windows. We could have implemented a strategy of cycling the rebooted node's radio on and off every few minutes until a communication window was observed, but we elected to have the node keep its radio active until it detected the presence of another node and then obtain the regional-node-based time from it.

The PATRIOT deployment began on the wrong foot, as it soon became clear that the provisioned Internet connection for the regional node was unusable due to high noise levels on the telephone line. This meant that one of our objectives, providing cluster-level METAR reports to external weather centers, would not be possible. We also had intended to allow authorized remote users to perform the same console-node CNAS commands as would be available if they were present in the monitoring area. Another objective lost to land-line quality.

We also had initial problems using the regional node as a network time server, which made synchronized communication windows difficult to achieve. Even using `ntpdate` with a 30-second timeout and a single sample, we had problems obtaining the time from the regional node. After some frustration, we discovered that the OLSR parameters at a number of the sensor agents had been set incorrectly, and that very few nodes were communicating beyond direct hops.[14] Once the parameter settings were corrected, the PASTA clocks became synchronized (at least within a few seconds of one another), and CNAS communication and cluster-head selection began to operate as intended.

With their Crossbow sensors deactivated, only TACMET agents could sense the environment. However, the other agents could still serve as cluster heads, and they still contributed to network connectivity (and therefore, still needed to participate in communication window activities). Originally only four TACMET II agents were planned, but without the Crossbow sensing, an additional four TACMET sensors were procured for the Exercise. Each sensor agent automatically detects if it has an operational TACMET sensor attached and, if it does, the agent assumes the TACMET-augmented role. However, there was another surprise in store for us. All four newly arrived TACMET sensors were producing garbled output. We initially feared that the new sensors had been damaged in transit, but we soon discovered that the serial output format of the new TACMET sensors was different from the original TACMETs—even though they were the same TACMET II model and part number as the originals.[15]

Fortunately, we had developed a live-updating facility for CNAS sensor agents. This facility allowed new or updated software to be distributed from a regional or console node to all CNAS agents during the next communication window. Taking advantage of the dynamic nature of Common Lisp, these updates are compiled and integrated directly in each running agent. The original plan was to test any such updates on a two sensor-agent CNAS network located in Amherst, Massachusetts. Tested updates would then be transmitted to the regional node at the PATRIOT Exercise via the telephone-line connection. However, with the land line unusable, we had to resort to transporting the regional node to the hotel (a one-hour trip), downloading the tested updates onto the regional node, and then returning the regional node to the Exercise site (another one-hour trip). As an additional complication, there were no TACMETs on the mini-network in Amherst—they were all at the Exercise! Nevertheless, co-authors Doug Holzhauer and Walt Koziarz used remote debugging mechanisms that had been put in place in CNAS to obtain the detailed serial device output from one of the remote sensor agents that was equipped with a new TACMET sensor. This information was later relayed verbally from the hotel by phone to Amherst. Once the updates supporting the new TACMETs were developed, transferred to the regional node, and then distributed to all CNAS nodes, all TACMET-augmented agents were sensing their surroundings.

During the exercise, transient hardware failures occurred in nearly one-half (6 of 17) of the sensor nodes. These failures occurred over several days when unseasonable air temperatures reached the upper 90s, and high humidity levels produced heat indexes approaching 110°F. These six sensor nodes returned to full functionality when the temperature dropped. Two other nodes failed permanently during the PATRIOT deployment. These failure rates were not unexpected, as many of the components used in the CNAS sensor nodes and the construction methods employed and were not intended to be used in such harsh surroundings.

In addition to hardware failures and software bugs, the PATRIOT deployment involved coping with *real* bugs. The worst of these were swarms of "ravenous" grasshoppers inhabiting the Fort McCoy area. They ate all of the surveyor flags that had been affixed to the sensor-node enclosures for visibility improvement. The grasshoppers also enjoyed wire insulation, and some even took up residence within the PASTA computer box, fancying the space between the boards comprising the processor and modules stack.

Even with these many issues, the PATRIOT deployment was a success. Sensor agents adapted to communication and node failures, reassigned cluster-head roles as needed, and provided local atmospheric data as designed. One afternoon, a regional tornado watch forced the cancellation of all activities and the withdrawal of troops and personnel (and the regional node!) from the area. CNAS sensor agents remained on duty, and when the regional node was activated the next morning, atmospheric data of the strong front's passage was provided.

4. NEXT DEPLOYMENT: AUSTRALIA

Based on the PATRIOT performance, CNAS has been selected for demonstration at the Talisman Saber Combined Exercise to be conducted May–July 2007 in Australia. Talisman Saber is a biennial series of joint exercises aimed at further developing and enhancing the defense relationships between the United States and Australia. With over 15,000 U.S. and 12,000 Australian personnel scheduled to participate, it is the largest and highest priority Tier II exercise in

[14]This highlights the difficulty of fully testing a sensor network like CNAS prior to deployment. The characteristics of widely scattered nodes cannot be duplicated accurately—even with sensor nodes distributed around research laboratory buildings.

[15]We learned later that the new TACMETs include a battery status report in their output.

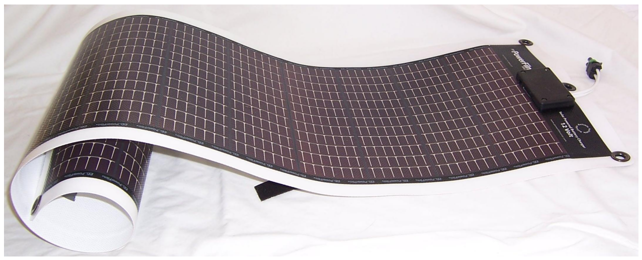

Figure 8: Rollable Solar Panel

the Pacific Theater.

One change from the PATRIOT deployment will be the full use of Crossbow sensing. We will use the MTS420CA's GPS unit to obtain the node location and set the clock. The GPS unit will be turned off most of the time, and only reactivated occasionally to correct clock drift and check that no one has moved the sensor node. We also plan to experiment with shorter communication windows than we used at PATRIOT. Since it will be winter, heat failures should not be a problem. (We do wonder what unexpected animal or insect life will complicate matters "down under.")

5. REPLENISHABLE POWER RESERVES

Sensor agents whose power reserves can be expected to be replenished from time to time adds new challenges in power management. We are currently exploring the addition of a rollable (thin film on plastic) solar panel (see Figure 8) to each sensor agent that allows its battery reserves to grow (up to full battery capacity) if the agent is in an unshaded location and the sun is shining. The activity decisions that are made by agents with replenishable resources now must consider how much additional power may become available and when. We assume that the CNAS sensor network is provided with the sunshine forecast (from sources outside the network), but each agent needs to learn the implications of the forecast on its own power reserves. A sensor agent that recognizes that it is shaded by a hill in the afternoon should realize that it cannot expect to obtain much power replacement if the forecast is for cloudy weather until noon. On the other hand, an unshaded agent could anticipate being able to perform additional power-intensive activities, based on its expectation of power replenishment.

6. SUMMARY

Developing CNAS has been exciting and challenging. Sensor agents operating near the limit of radio-communication range with their radios turned off most of the time lowers power expenditure at the cost of network responsiveness. Although CNAS can shift its communication policies in response to observations or planned objectives, it can be frustrating to have to wait until the next communication window in order to obtain CNAS data or transition all agents into continuous-communication mode. As developers, however, such frustrations are somewhat mitigated by the level of software capability that we have been able to achieve at each agent. PASTA processors running Linux, Common Lisp, and GBBopen provide a level of programming expressivity and on-the-fly modification that greatly facilitates the implementation of advanced behaviors and opportunistic-control decision making. We are fortunate in CNAS to have sufficient processing power and memory available to make shoe-horning complex behaviors and reasoning into tightly constrained C (and assembly) code a distant memory. Our emphasis to date has been on demonstrating basic CNAS capabilities and reliability, and we are eager to begin incorporating more complex activities and adaptive reasoning into CNAS in the coming months. We are also working to shorten the stabilization portion of the communication window by providing initial routing estimates at WiFi on time that are based on greater information sharing between the network routing and application layers.

The computational and power-management capabilities of the next generation of microsensor platforms will be even greater.[16] Perhaps someday soon, in a new generation of CNAS-like networks with advanced, low-power, long-distance communication hardware, we'll be able to say: "Leave your radios on!"

Acknowledgments

Kevin Bartlett, Wilmar Sifre, Wenchian Lee, and Paul Yaworsky of AFRL also contributed to CNAS. ISI's Joe Czarnaski and Brian Schott provided PASTA microsensor advice and assistance.

7. REFERENCES

[1] K. Akkaya and M. Younis. A survey on routing protocols for wireless sensor networks. *Elsevier Ad Hoc Network Journal*, 3(3):325–349, 2005.

[2] C.-Y. Chong and S. P. Kumar. Sensor networks: Evolution, opportunities, and challenges. *Proceedings of the IEEE*, 91(8):1247–1256, Aug. 2003.

[3] D. A. Clark and M. P. Matthews. An integrated sensor testbed for support of theater operations in areas of complex terrain. In *Proceedings of the Battlespace Atmospheric and Cloud Impacts on Military Operations Conference (BACIMO 2000)*, pages 25–27, Fort Collins, Colorado, Apr. 2000.

[4] D. D. Corkill. Blackboard systems. *AI Expert*, 6(9):40–47, Sept. 1991.

[5] R. S. Engelmore and A. Morgan, editors. *Blackboard Systems*. Addison-Wesley, 1988.

[6] S. E. Keene. *Object-Oriented Programming in Common Lisp*. Addison-Wesley, 1989.

[7] G. Kiczales, J. des Rivieres, and D. G. Bobrow. *The Art of the Metaobject Protocol*. MIT Press, 1991.

[8] B. Schott, M. Bajura, J. Czarnaski, J. Flidr, T. Tho, and L. Wang. A modular power-aware microsensor with >1000X dynamic power range. In *Information Processing in Sensor Networks (ISPN 05), special track on Platform Tools and Design Methods for Network Embedded Sensors*, Los Angeles, California, Apr. 2005.

[9] Y. Xu, J. Heidemann, and D. Estrin. Geography-informed energy conservation for ad hoc routing. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'01)*, pages 70–84, Rome, Italy, July 2001.

[16] PXA270-based modules are currently in Alpha testing.