Reporting Down Under

A CNAS (Collaborative Network for Atmospheric Sensing) Update

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

ABSTRACT

CNAS (Collaborative Network for Atmospheric Sensing) is an agent-based, power-aware sensor network for ground-level atmospheric monitoring. To conserve battery power, CNAS sensor agents must have their WiFi radios turned off most of the time, as even *listening* consumes significant power. This complicates agent interaction and system responsiveness, because an agent cannot simply turn on its radio when it needs to send a message. CNAS agents also must have their radios turned on when others are sending messages to them and to support multi-hop message forwarding.

In this paper, we briefly review the sensor-agent hardware and blackboard-system software used in CNAS, as well as how CNAS agents collaborate with only periodic radio availability. We also relate experiences and lessons learned from field deployments of CNAS at the Talisman-Saber Combined Exercise held in Queensland, Australia. We conclude with an overview of CNAS research performed since Talisman-Saber that focuses on: 1) improving CNAS performance and responsiveness with limited radio availability, 2) power-aware reasoning associated with solar harvesting obtained from a rollable solar panel at each sensor agent, and 3) potential next-generation CNAS hardware.

1. CNAS

CNAS[1] (Collaborative Network for Atmospheric Sensing) is an experimental, agent-based, power-aware sensor network for ground-level atmospheric monitoring [3]. CNAS is a research and demonstration tool developed jointly by AFRL (Rome, NY) and UMass for conducting realistic explorations of the advantages and limitations of agent-based environmental monitoring. The CNAS effort is investigating the use of hardware capabilities that are likely to become cost-effective for production deployments in the next few years.

A combination of blackboard and multi-agent system (MAS) techniques are used in CNAS sensor agents. Black-

board systems [2, 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 developing an effective software architecture for agile, resource-aware sensor network agents.

Each CNAS sensor agent is running Linux, Common Lisp, and the GBBopen blackboard-system framework.[2] Developing CNAS agent software using Common Lisp and GBBopen provides a level of programming expressiveness and on-the-fly modification that greatly facilitates the implementation of advanced behaviors and opportunistic-control decision making.

Agent types

A CNAS network contains four different "types" of agents:

Sensor Agents: Each sensor agent is built around ISI's PASTA (Power-Aware Sensing, Tracking, and Analysis) microsensor platform [8][3] and its Intel PXA255-based CPU. In addition to the PASTA, each sensor agent is equipped with a Crossbow MTS420CA sensor board containing:
- 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[4]

A Netgear MA111 Wireless adapter, connected to the PASTA's USB interface, provides standard IEEE 802.11b wireless communication. This hardware, and a 12V battery, is packaged in a PVC housing that positions the wireless antenna and sensors 4.25' above ground level. The agent packaging is intentionally large to allow easy access during testing and evaluation.
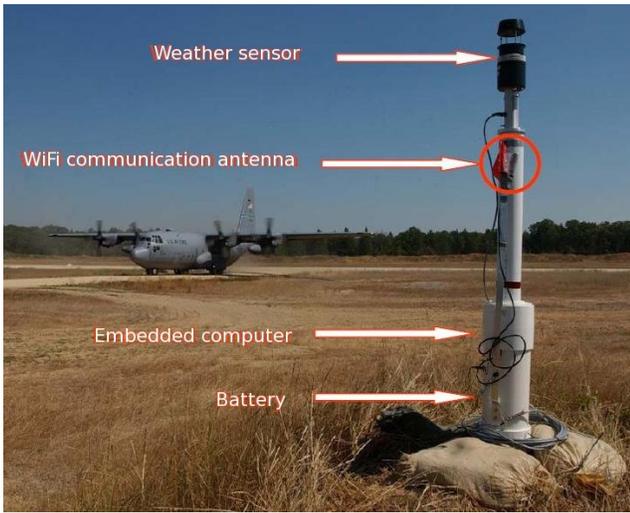
Each CNAS sensor agent performs basic sensing activities, obtaining atmospheric readings once each second. Following Air Force meteorological practice, the following summary readings are computed from the 1-second readings and saved every five minutes:
- temperature (5-minute average)
- dew point (5-minute average)

---

[1]Pronounced "see-nas." The original CNAS acronym was short for Cognitive Network for Atmospheric Sensing, but "Cognitive" has since evolved into "Collaborative," emphasizing the importance of the agent-based interactions supporting the *cognitive* reasoning activities in CNAS.

---

[2]http://GBBopen.org/
[3]http://pasta.east.isi.edu/
[4]Not used in CNAS.

Figure 1: A TACMET-Augmented Sensor (at the PATRIOT 2006 exercise)

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

Sensor agents perform saturation-vapor-pressure, humidity-to-dew-point, pressure-to-altimeter, millibars-to-inches, and wind-meter-to-knot computations as needed.[5] All unsent 5-minute summary observations are transmitted during the next communication window to the sensor agent that is acting as the *cluster head* (discussed shortly).

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).

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 1). 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 using the flux gate compass), temperature, and relative humidity readings are provided to the PASTA over an RS-232C serial connection once every second. Unlike the Crossbow, TACMET II measurements have been certified by the Air Force.

Console Nodes: A console node is a laptop or hand-held computer that, upon entry into the CNAS-network area, can obtain observation data from the network and display that data graphically. Unlike basic and TACMET-augmented sensor agents, 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

---

[5]The PASTA does not include floating-point hardware, so these computations are performed by software emulation.

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 special 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 the CNAS software at individual sensor agents.

Communication policies

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 on at any point in time.

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 agents 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 used in the CNAS deployments are:

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 6PM until 6AM (local time)

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

quarter-hourly-overnight-sleep The quarter-hourly policy, but without communication after 6PM until 6AM (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 agent 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 agent can be more aggressive and try connecting at the next quarter-hour window, and at subsequent fallback windows.

Inter-agent communication in CNAS uses standard TCP/IP operating over an OLSR (Optimized Link State
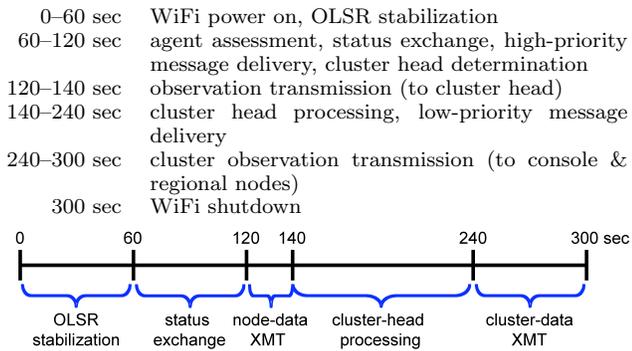
0–60 sec	WiFi power on, OLSR stabilization
60–120 sec	agent assessment, status exchange, high-priority message delivery, cluster head determination
120–140 sec	observation transmission (to cluster head)
140–240 sec	cluster head processing, low-priority message delivery
240–300 sec	cluster observation transmission (to console & regional nodes)
300 sec	WiFi shutdown

Figure 2: CNAS Communication Window

Routing)[6] multi-hop protocol. (Use of OLSR was an externally imposed requirement for the CNAS effort.) 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. Because the radio has been powered off, OLSR reinitializes at the start of each communication window. Given this initialization requirement, each CNAS communication window was structured as the sequence of activities shown in Figure 2. The durations of these staged activity intervals are very conservative and provide substantial slack time for OLSR re-initialization and for coping with highly degraded communication. 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 locally when the agent's radio is turned off.

Cluster heads

In addition to its sensor duties, a regular or TACMET-augmented sensor agent can also assume the role of cluster head. A *cluster* in CNAS is a grouping of sensor agents located in a geographic region of interest. Non-overlapping cluster regions are user-defined, and each agent determines its cluster membership at start up, based upon its location and the specified cluster regions. Through an information-spreading process, the identity and locations of all agents in a cluster become mutually 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 appear to be alive in its cluster. If a sensor 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.[7] 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

---

[6]http://olsr.org/

[7]Cluster summaries include the list of the individual agents that contributed to them.
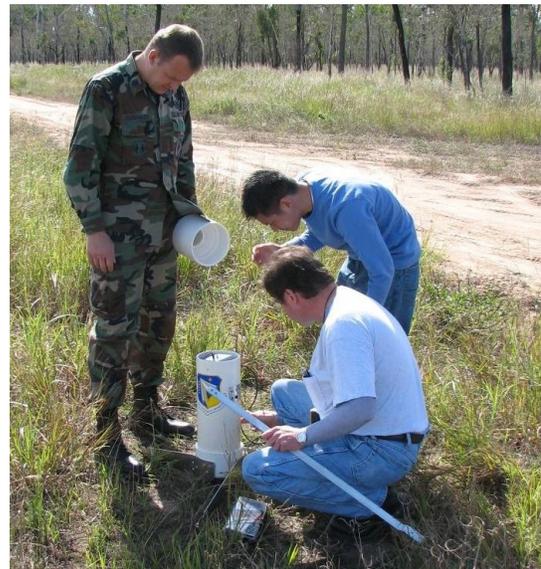

Figure 3: Assembling a Sensor Agent

nodes that are active.

The cluster-head decision strategy takes advantage of the OLSR-layer routing information to determine what agents 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.

2. TALISMAN-SABER DEPLOYMENTS

In July 2006, CNAS was field tested at the 2006 PATRIOT Exercise held at Fort McCoy, Wisconsin. Based on the PATRIOT performance [3], CNAS was invited to participate at the 2007 Talisman-Saber Combined Exercise in Queensland, Australia. Talisman-Saber is a biennial Australia/United States bilateral exercise and the primary training venue for Commander Seventh Fleet Combined Task Force (CTF) operations. Over 15,000 Australian and US forces participated in Talisman-Saber 2007.

At the 2006 PATRIOT Exercise, problems with the PASTA firmware interface to the Crossbow forced us to deploy CNAS at PATRIOT with the Crossbow sensors disabled. An important goal for the Talisman-Saber deployment was to have the Crossbow fully operational. Unfortunately, ISI was unable to resolve the PASTA firmware issue (see Section 5), and we were forced, once again, to operate without Crossbow sensors.

Without the Crossbow, GPS positioning and system-clock setting[8] is also 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 agent name were entered into a console-node laptop and then transferred to a configuration file on the sensor agent. In addition, since agents could not obtain the time using GPS, we used the re-

---

[8]The PASTA does not have a hardware clock.

Figure 4: TACMET-Augmented Sensor Agent at the Talisman-Saber Drop Zone (Note the "high-tech" broom-handle antenna mast to the left of the sensor.)

Figure 5: CNAS Sensor Agents in Transport

gional node as a CNAS time server. 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 agent's radio on and off every few minutes until a communication window was observed, but we elected to have the agent keep its radio active until it detected the presence of another agent and then obtain the regional-node-based time from it.

Drop-zone deployment

The CNAS network was initially deployed at Talisman-Saber at a remote[9] aerial drop zone (Figure 3). Conditions at the drop zone were austere: no electrical power, no telephone communication, and the shelter for the regional node and staff consisted of a floor-less tent erected in a field of brush adjacent to the drop zone. The CNAS objective was to gather and report a 72-hour window of historic low-level atmospheric data to aid in airdrop operations scheduled for June 19, 2007. Unlike the PATRIOT 2006 deployment which had two clusters, only a single cluster was defined in each of the Talisman-Saber deployments.

CNAS sensor agents were deployed in a linear line at the drop zone, with TACMET-augmented sensor agents interleaved with basic sensor agents (with disabled Crossbows). The distance between agents was almost 300m (at the limit for 802.11b). Position decisions were made in the field by walking with a laptop from the last positioned agent until its WiFi signal was lost, then walking backward a few meters and placing the sensor agent.

One technical issue that arose was the taller than anticipated native vegetation interfering with the 802.11b wireless communication used by CNAS agents. This issue was resolved by moving the WiFi adapters from the existing antenna masts to broom handles (obtained from a not so nearby hardware store) which raised the adapters high enough to diminish the native-vegetation attenuation (Figure 4).

Unlike the previous CNAS deployment at the 2006 PATRIOT Exercise—where the hardware in two sensor agents failed permanently and 6 of the remaining 17 sensor agents experienced transient hardware failures due to the unseasonably high air temperatures and humidity levels during the

_____
[9] A 90-minute drive over muddy "roads" from nightly sleeping facilities.

deployment—all CNAS agents operated flawlessly throughout all Talisman-Saber deployments. We attribute this to the elimination of faulty PASTAs during the 2006 PATRIOT "burn in" (literally) experience, improved agent-construction methods, and the more moderate _winter_ weather in July in Queensland. The swarms of grasshoppers that took up residence within the PASTA computer boxes at PATRIOT 2006 (fancying the space between the processor and module boards) and that ate wire insulation within sensor agents were also missing from the Queensland deployments. Thankfully, no native creatures caused problems at Talisman-Saber.

Problems did arise with the regional-node laptop, however. The sound chip in the laptop failed and began spitting out sporadic interrupts, which interfered with the timing in the laptop's internal WiFi adapter. Because the bursts of spurious interrupts were unpredictable, connections might work for a while and then suddenly die (after broadcasting a number of packets with incorrect time values, confusing the OLSR collision-avoidance scheme at other agents). Once diagnosed, a backup laptop was used as the regional node. However, the backup laptop did not have a car charger, so without line power at the drop zone, the replacement regional node had to operate on its own limited battery power. Eventually, a connection was rigged that allowed the battery in an uninterruptable power supply (UPS) unit to be charged from a car. The laptop's AC power supply could then draw AC power from the UPS unit to power the laptop and recharge the laptop battery. The failure of the original regional node laptop also killed our goal of uploading real-time CNAS weather information to the Air Force Weather Agency (AFWA) server from the drop zone using the Iridium satellite communication system.

Drop-zone redeployment

The original Talisman-Saber Exercise demonstration plans called for CNAS to be removed from the drop zone prior to airdrop operations (after collecting the 72 hours of observations). So, on June 18th, the drop-zone deployment was dismantled and the sensor agents transported to the Urban Operations Training Facility (UOTF) for deployment there (Figure 5). The UOTF deployment was nearly complete when, early in the morning of the 19th, CNAS was unexpectedly summoned back to the drop zone to provide real-time wind data during the airdrops. The UOTF deployment was dismantled and the sensor agents transported back to the drop zone. The CNAS team demonstrated a rapid re-

Figure 6: TACMET-Augmented Sensor Agent at UOTF (The device to the right of the sensor is not part of CNAS.)

sponse and set-up capability by having CNAS on-line and reporting observations within 15 minutes of arrival at the drop zone.[10]

UOTF deployment

After the drop-zone redeployment, CNAS was moved back to the UOTF. The UOTF is an urban environment that was constructed at the Shoalwater Bay Training Area. UOTF includes a number of buildings (commercial, retail, residential, shanty, rubble) built using both standard building materials and reconfigurable container-based structures. CNAS deployment at UOTF differed from the drop zone because the sensor agents were located much closer to each other and some of them were positioned on buildings that had different heights. In terms of WiFi communication, vegetation attenuation was replaced with urban reflection and interference. Fortunately, AC power for the replacement regional-node laptop was available at UOTF, eliminating that headache.

Several additional technical issues surfaced at the UOTF. Water from heavy rains entered the WIFI cable connectors and disrupted network operation. This was resolved on-site with application of petrolatum as sealant for the connections. A second issue resulted from periodic electromagnetic interference from a nearby demonstration system that was disrupting the 802.11b frequency range. This interference issue required no additional intervention, as the CNAS-agent software was sufficiently robust to recover gracefully during interference-free periods with no loss of data. CNAS pushed hourly weather observations, in properly-formatted structure, to the AFWA server as well as to the Australian Bureau of Meteorology (BOM) (Table 1).

A return to Australia in 2009?

The CNAS deployments at Talisman-Saber met all the technical goals and objectives established for the Exercise. These included: 1) automatic dissemination and posting of weather observations to AFWA and BOM weather servers, 2) support of the AFRL "COUNTER" small UAV demonstration, 3) support of airdrop operations, and 4) adapting to chang-

---

[10]The sensor agents were positioned at the same locations as the original drop-zone deployment.

ing user requirements, observation needs, and mission objectives. Even in its experimental form, CNAS was deployed and providing information within 15 minutes of arrival at the drop-zone site—a highly visible achievement. As a result, CNAS has been invited to participate in the next Talisman-Saber Exercise in 2009.

3. SILENCE WITH RESPONSIVENESS

Collecting sensor readings, aggregating them together at the cluster heads, and communicating them to console or regional nodes is well suited to the periodic communication windows used in CNAS. However, when more dynamic activities need to be performed (such as inspecting the local state of an agent, retasking an agent's activities or changing network policies, or having agents modify the world around them), having to wait until the next communication window can be an issue (as well as frustrating). Techniques are needed that improve network responsiveness without increasing the total amount of time that agents' radios need to be turned on.

Initially, CNAS agents were constrained to communicate using a "stock" OLSR routing protocol. Following the Talisman-Saber Exercise, this program-objective restriction was relaxed, allowing exploration of improvements to standard OLSR.

Persistent routing tables

One obvious improvement is to eliminate OLSR reinitialization at the start of each communication window. By having OLSR assume that no change has occurred while the wireless radio has been off, it can proceed when the radio is switched on using the state that existed at the end of the previous communication window. Intuitively, it is equivalent to having all changes happen at the moment the radio was switched back on. To the degree that the old routing information is reasonable, application-level communication is possible immediately and, hopefully, the cost of any adaptation is less than the loss of time required for complete reinitialization.

AFRL's Zenon Pryk explored maintaining OLSR routing information across communication cycles using a small CNAS agent network deployed indoors at AFRL. Experiments were run where a small utility which exercised all possible source & destination pairs. In this noisy indoor setting, OLSR stabilized in less than 15 seconds into the communication window versus the 40–50 seconds required by a from-scratch reinitialization. In a sparse, outdoor CNAS setting we expect the routing changes to be less dynamic due to the small number of paths available, and that OLSR with persistent routing tables will stabilize even faster.

Using organization knowledge in routing

MASs benefit greatly from an organization design [4, 6] that guides agents in determining when to communicate, how often, with whom, with what priority, and so on. However, this same organization knowledge is not utilized by general-purpose wireless network routing algorithms normally used to support agent communication. By making the routing algorithm aware of how agents interact in the CNAS organization, it can direct path exploration where it is most beneficial, find optimal paths faster, and conserve significant bandwidth for application use. General-purpose routing algorithms also treat all message as having the same priority. They spend the same amount of energy exploring paths

```
METAR KQRS 240755Z AUTO 11001KT 17/14 A3005 RMK PK WND 08003/52 SLPNO ESTMD ALSTG P1013;
METAR KQRS 240855Z AUTO 28000KT 15/14 A3007 RMK PK WND 29002/54 SLPNO ESTMD ALSTG P1014;
METAR KQRS 240955Z AUTO 24001KT 15/14 A3007 RMK PK WND 25003/54 SLPNO ESTMD ALSTG P1014;
METAR KQRS 241055Z AUTO 03000KT 14/13 A3008 RMK PK WND 35002/53 SLPNO ESTMD ALSTG P1014;
METAR KQRS 241155Z AUTO 18001KT 15/14 A3009 RMK PK WND 21004/50 SLPNO ESTMD ALSTG P1014;
METAR KQRS 241255Z AUTO 16000KT 15/14 A3007 RMK PK WND 22003/52 SLPNO ESTMD ALSTG P1014;
METAR KQRS 241355Z AUTO 19001KT 14/13 A3004 RMK PK WND 24003/54 SLPNO ESTMD ALSTG P1013;
METAR KQRS 241455Z AUTO 20001KT 14/13 A3006 RMK PK WND 22003/55 SLPNO ESTMD ALSTG P1014;
METAR KQRS 241555Z AUTO 19001KT 14/14 A3004 RMK PK WND 23003/50 SLPNO ESTMD ALSTG P1013;
METAR KQRS 241655Z AUTO 08001KT 15/14 A3001 RMK PK WND 10003/50 SLPNO ESTMD ALSTG P1012;
METAR KQRS 241755Z AUTO 22001KT 14/14 A3002 RMK PK WND 22005/52 SLPNO ESTMD ALSTG P1012;
```

Table 1: A portion of the METAR statements produced by CNAS at UOTF

for application messages where it is extremely important to find the best possible path, as they do exploring paths for messages that are not as important, and can be delayed. We wanted to make use of the organizational importance (priority) of various agent communications when controlling routing.

We modified a proactive routing protocol (CQRouting [7]) to incorporate knowledge of the CNAS organization (which agents communicate with which other agents given their roles and cluster membership, and the priority and frequency of the messages being sent). Using a detailed network-level simulation of CNAS, we observed a large reduction in the number of exploration messages relative to traditional routing protocols (such as OLSR). This resulted in a significant (43%) increase in the communication bandwidth available to the CNAS application. We also obtained an increased global utility with fewer exploration messages by having the eCQRouting routing protocol bias path exploration based on message priority characteristics of the CNAS organization. This work is detailed in Zafar et. al. [11].

Rapid radio cycling

Maintaining routing information across CNAS communication cycles and providing organizational communication information to the routing algorithms allow application-level communication to be performed using shorter communication windows. Shorter windows allow more of them (a higher communication-cycle frequency) to be supported with the same power expenditure. At the extreme, we would like to have each agent turn on its radio and, as quickly as possible, determine if it needs to be available to support any message communication during that window. If no communication support is needed (either to send a message, receive a message, or perform multi-hop forwarding), then the agent can turn off its radio straight away. The shorter these "turn on and determine" windows can be made, the more frequently they can be cycled while maintaining the same level of energy expenditure. Frequent cycles mean more CNAS responsiveness.

Intuitively, we could estimate the time needed for any agent to get an "I need to communicate" broadcast message to all other agents in the network. Then, when each agent begins a cycle, it keeps its radio turned on for that long to hear if any agent (including console and regional nodes) wants to communicate. If no such message is received during that time, it turns off its radio.

For a small network, this broadcast time will be short, but the time increases as the number of hops grows. However, for very brief "no message" communication windows, the time until the next window is also short, so the need to communicate a message across the network in one cycle is lessened. If the message does not make it all the way to its destina-

Figure 7: Rollable Solar Panel

tion in the current window, it will be stored and forwarded in the next. Thus, what we really seek is having the agents along a segment of the message path determine that they are needed in the current cycle. The goal is obtaining "waves" of communication-forwarding windows appearing where and when needed in the network. As with the organizationally based routing work described previously, CNAS can benefit by having the network-routing layer be aware of the application-level communication window and cycles. Making the "no message" decision at a lower communication layer can be very efficient.

We are beginning development and evaluation of such a rapid-cycle, communication-window-aware routing protocol for CNAS-like networks. What differentiates this work from recent power-aware routing research is that we assume all agents' radios remain off (except for the frequent "any communication?" windows) until packets along a path need to be propagated. Rapid cycling is ideal for conserving communication energy in low-bandwidth, highly responsive settings.

4. SOLAR HARVESTING

Sensor agents whose power reserves can be expected to be replenished from time to time add new challenges in power management. We are currently exploring the addition of a rollable (thin film on plastic) solar panel (see Figure 7) 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.

A requirement of CNAS is that the positioning of sensor agents cannot be optimized for either data collection or solar harvesting. Eventually sensor agents may be airdropped (rather than hastily deployed by hand). In either case, the exact placement of sensor cannot be regulated, which requires the shade and tilt attenuation to be determined once the agents are situated. Thus, agents in CNAS, like agents deployed in other real-world applications, need to determine aspects of their local environments in order to make appropriate decisions. For predicting solar-harvesting amounts, each agent must develop a model of the efficiency of its solar panel, attenuation due to positioning of the panel relative to the sun, attenuation due to shading (by trees, hills, buildings, etc.), and attenuation due to cloud cover. With such a model, an agent can translate a temporal sunshine forecast into a realistic estimate for the solar energy to be harvested at its location. This model development must be done quickly, as power-management decisions will be inaccurate until each agent becomes aware of its surroundings.

We have developed a strategy of factoring solar-harvesting model development (which would typically be done using multi-agent learning) into two phases: pre-deployment (site independent, individual agent) learning and post-deployment (site dependent, multi-agent) model completion. By performing as much site-independent learning as possible prior to deployment, we simplify what needs to be determined on-site by each sensor agent. Furthermore, we use collective sharing of local-observation information, in conjunction with temporal and spatial constraints in relating this information, to reduce the number of observations needed to perform each agent's model-completion activities. In all but the most unlikely of environmental conditions, our two-phased strategy allows individual-agent harvesting models to be completed using only the first and second day's observations. Thus each agent can make fully informed solar-harvesting predictions given cloud forecasts starting on the third day. This work is detailed in Zafar and Corkill [10].

5. NEXT-GENERATION HARDWARE

The PASTA microsensor platform used in CNAS is already showing its age. New PASTA "Ziti" processors are no longer available and, as discussed above, problems interfacing the Crossbow to the PASTA remain unresolved. Similarly, the Netgear MA111 (IEEE 802.11b) USB Wireless adapter used in CNAS is outdated. After participating in the Talisman-Saber Exercise, we began looking into new hardware possibilities for CNAS agents.

One promising line of replacement processors are the PXA255-based Connex and the PXA270-based Verdex motherboards from Gumstix, Inc. These have become very popular, and they can be interfaced with compatible sensor, WiFi, and GPS hardware. The Gumstix have the advantage of being commodity devices readily available at commodity prices. However, they have the disadvantage of not being designed or targeted for low-energy consumption. The PASTA has a distributed-peer architecture that can decouple processing from peripheral operation, allowing even the central processor to be powered down to lower total system-power expenditure [8]. For use as a CNAS agent, however, this limitation is not a serious liability. Managing the high-expenditure WiFi and GPS devices has been the prime focus of our CNAS effort, and having the Gumstix running continually is acceptable.

Porting CLISP to the Gumstix

GBBopen and its Common Lisp substrate provide many advantages for CNAS. One important advantage is the ability to compile source code directly in the Common Lisp image, without the need for a traditional development environment on the CNAS agent. We used the open-source Common Lisp implementation, CLISP,[11] in the PASTA-based sensor agents, so also providing CLISP on the Gumstix processors was crucial. An important advantage of CLISP is that it is structured as a small C-based kernel that operates in conjunction with a platform-independent byte-code 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 byte-code executer are well suited to the memory space available on the PASTA and Gumstix processors.

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. This was not the case for the Gumstix, as the kernel and library code there are tightly coupled with the buildroot cross-compilation toolchain. CLISP has a fairly complicated build process that has evolved over the years to support deployment on a wide range of hardware and operating system platforms. At one point, cross compilation of CLISP was supported, but that capability has long passed into disrepair. We did not have the space or desire to use an ARM-based GNU development environment on the Gumstix itself, so we had no alternative but to explore CLISP cross compilation.

Building CLISP involves the following eight steps:

1. `make init` — prepares all symbolic links and utilities
2. `make allc` — makes all `*.c` files
3. `make lisp.run` — makes the C-kernel executable
4. `make interpreted.mem` — creates a memory image with everything uncompiled
5. `make halfcompiled.mem` — creates a memory image with `compiler.fas` and the rest uncompiled
6. `make lispinit.mem` — makes all `*.fas` files and creates a memory image with everything compiled
7. `make modular` — makes the base module set
8. `make install` — completes the CLISP installation

We developed a procedure for performing the first three steps (with manual interventions in the automated build scripts), to create a `lisp.run` executable for the Gumstix using the Linux/x86 buildroot cross-compilation toolkit. Compilation of the Common Lisp source files for the CLISP compiler and the rest of the CLISP Common Lisp environment was performed using a normal CLISP image running on the Linux/X86 development platform with appropriate compilation settings for the ARM-based deployment.[12] Then the compiled output files (`*.fas` files) and the `lisp.run` executable are copied to the Gumstix and the `lispinit.mem` memory image is built there. Again, this requires manual intervention, however the result is the latest CLISP release running smoothly on the Gumstix.

---

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

[12]With the exception of calls to some low-level routines, CLISP's byte-code compilation is platform independent. The compiler settings address these low-level issues.

Given the suitability of CLISP for small, embeddable devices such as the Gumstix, restoring cross-compilation capability to the automated build scripts would eliminate the labor-intensive manual build process that we were forced to use. As an open-source project with a large user base, nontrivial changes to the current build process will require care and perseverance in order to be accepted. So, at least for now, recompiling CLISP for new system-software updates (such as the new Gumstix OpenEmbedded (OE) build process) will remain labor intensive.

We are only beginning to consider sensor hardware for the Gumstix platforms. Given the growing success of this marketplace, however, the CNAS goal of exploring hardware capabilities that are likely to become cost-effective for production deployments in the next few years is quickly becoming a reality.

6. SUMMARY

CNAS has been successful as both a deployed agent-based sensor network and as a research environment. Working on CNAS has also been a lot of fun! Sensor agents operating near the limit of radio-communication range with their radios turned off most of the time present different challenges than traditional MAS settings. Our emphasis to date has been on achieving basic CNAS capabilities and reliability, and we are only starting to expand our use of blackboard-system capabilities in incorporating more complex activities and adaptive reasoning into CNAS.

CNAS has become acceptably reliable for use in the field (literally), and there is interest in demonstrating CNAS on board ships at sea. The latest hardware directions, and accompanying low cost, will also enable permanent deployment of CNAS agents in areas where the economic cost of losing a sensor agent due to damage or theft becomes tolerable. The future for agent-based sensor networks like CNAS is growing ever brighter. We are already looking forward to Talisman-Saber 2009!

Acknowledgments

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

[3] D. D. Corkill, D. Holzhauer, and W. Koziarz. Turn off your radios! Environmental monitoring using power-constrained sensor agents. In *Proceedings of the First International Workshop on Agent Technology for Sensor Networks (ATSN-07)*, pages 31–38, Honolulu, Hawaii, May 2007.

[4] D. D. Corkill and V. R. Lesser. The use of meta-level control for coordination in a distributed problem-solving network. In *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, pages 748–756, Karlsruhe, Federal Republic of Germany, Aug. 1983. (Also published in *Computer Architectures for Artificial Intelligence Applications*, Benjamin W. Wah and G.-J. Li, editors, IEEE Computer Society Press, pages 507–515, 1986.).

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

[6] B. Horling and V. Lesser. A survey of multi-agent organizational paradigms. *Knowledge Engineering Review*, 2005.

[7] S. Kumar. Confidence based dual reinforcement Q-routing: An on-line adaptive network routing algorithm. Master's thesis, University of Texas at Austin, Austin, Texas, 1998.

[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.

[10] H. Zafar and D. Corkill. Simplifying solar-harvesting model development in situated agents using pre-deployment learning and information sharing. In *Proceedings of the Second International Workshop on Agent Technology for Sensor Networks (ATSN-08)*, Estoril, Portugal, May 2008. To appear.

[11] H. Zafar, V. Lesser, D. Corkill, and D. Ganesan. Using organization knowledge to improve routing performance in wireless multi-agent networks. In *Proceedings of the Seventh International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2008)*, Estoril, Portugal, May 2008. To appear.

---

[13]Professors Victor Lesser and Deepak Ganesan advised Huzaifa on the organization-based routing research.