Real deployments of sensor networks

Mike Yu Cheng
DCSL
Fall 2005

Outline

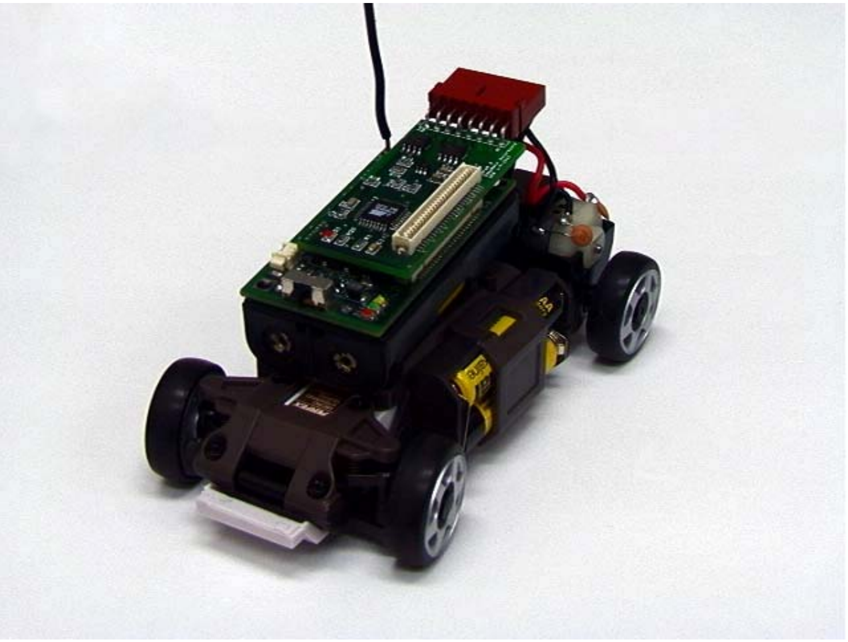
- Common hardware
- Common software
- Look at several real deployments
  - Firebugs
  - Calamari
  - Exscal

Questions for each deployment

- What is the goal of the deployment?
- Where is the deployment?
- What is the system architecture?
- What is the hardware and the software in the system?
- What is the high level algorithm
- Experiment Results

MICA mote

- From UC Berkeley
- 433, 868/916, or 310 MHz Multi-Channel Radio Transceiver
- >1yr Battery Life on AA Batteries (Using Sleep Modes)
- Light, Temperature, Barometric Pressure, Acceleration/Seismic, Acoustic, Magnetic, GPS, and other Sensors available
- $125, $2000 for a kit



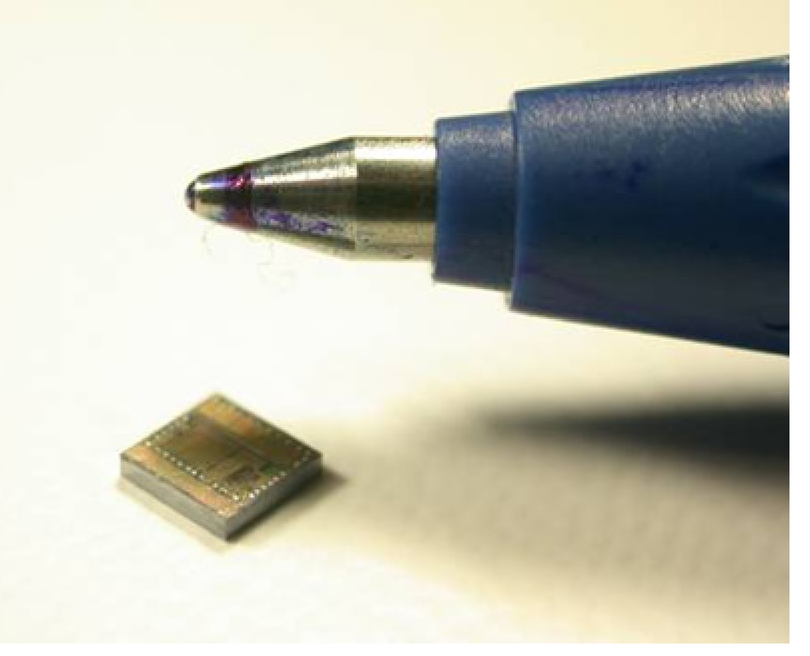
http://www.xbow.com/

MICA mote CPU

- Atmel ATmega 128L processor running at 4 megahertz.
- The 128L is an 8-bit microcontroller
- 128 kilobytes of onboard flash memory to store the mote's program.
- ATmega consumes only 8 milliamps when it is running, and only 15 microamps in sleep mode
- Software on MICA motes is built on TinyOS
- 120 hours running time with 2 AA

COTS dust

- Contains:
  - CPU
  - memory
  - A/D converter for reading sensor data
  - radio transmitter.
- Needs:
  - sensors
  - battery
  - antenna.
- Cost
  - less than a dollar when it is mass produced



http://computer.howstuffworks.com/mote5.htm

COTS dust vs. MICA Mote

- Sense:
    - temperature
    - humidity
    - barometric pressure
    - light intensity
    - tilt and vibration
    - magnetic field sensors

- 1 cubic inch including
    - bi-directional radio
    - 20 meter communication range
    - microprocessor controller
    - battery
- one week lifetime in continuous operation

http://computer.howstuffworks.com/mote5.htm

TinyOS

- License: BSD License
- Operating System: All 32-bit MS Windows (95/98/NT/2000/XP), All POSIX (Linux/BSD/UNIX-like OSes)
- OS Independent
- Programming Language: C, Java, Perl
- NesC language, C-like syntax, compiler

Sample nesC code

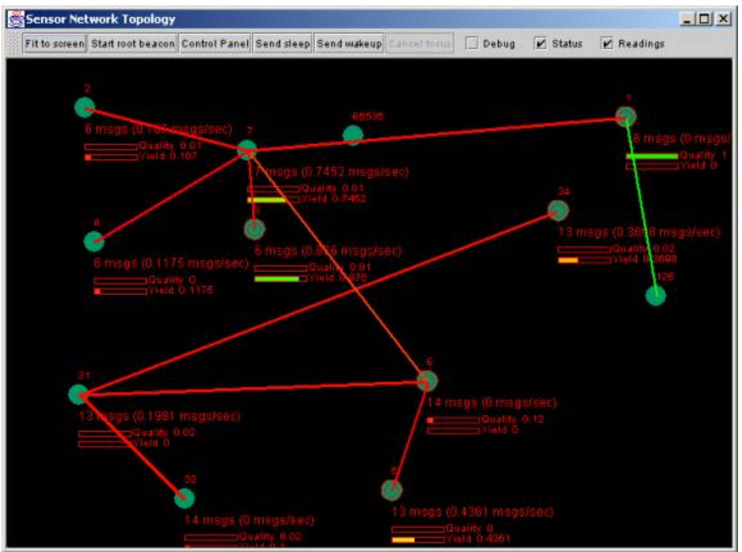
- implementation {

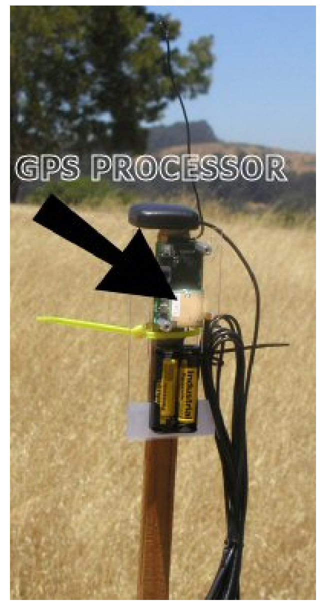
```
  command result_t StdControl.init() {
    call Leds.init();
    return SUCCESS;
  }

  command result_t StdControl.start() {
    return call Timer.start(TIMER_REPEAT, 1000) ;
  }

  command result_t StdControl.stop() {
    return call Timer.stop();
  }

  event result_t Timer.fired()
  {
    call Leds.redToggle();
    return SUCCESS;
  }
}
```

http://www.tinyos.net/tinyos-1.x/doc/tutorial/lesson1.html

FireBug

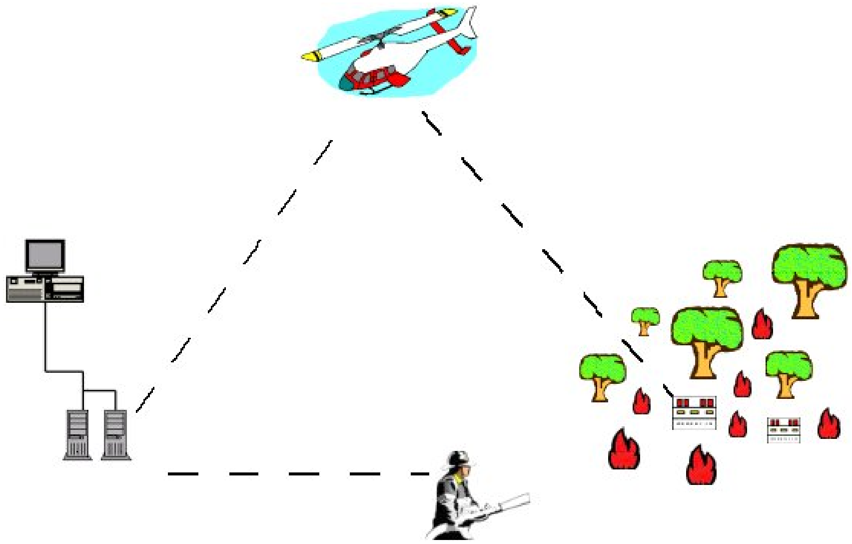
- Goal?
  – Collecting real time data from wildfires





FireBug

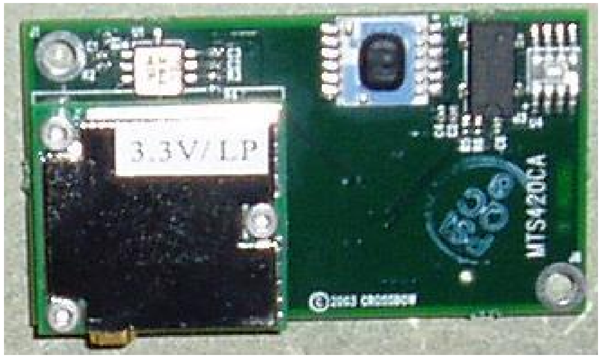
- Where?
  - Berkeley
- System Architecture?
  - a network of GPS-enabled, wireless thermal sensors
  - a control layer for processing sensor data
  - a command center for communicating with the sensor network.

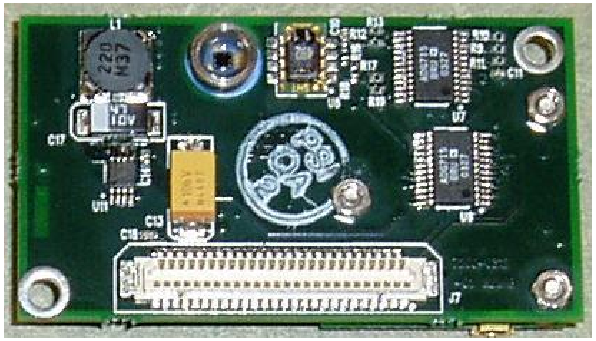


http://firebug.sourceforge.net/

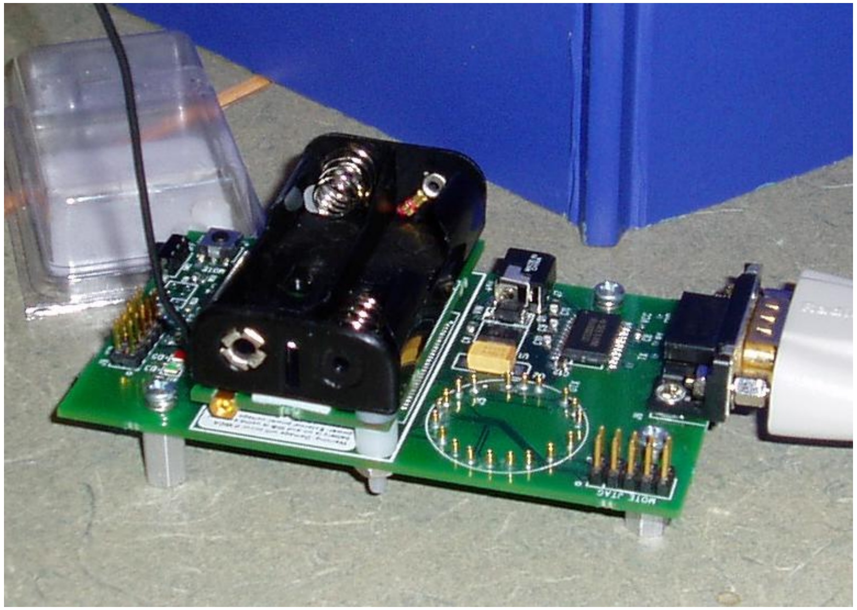
FireBug

- Hardware?
  - MICA mote
  - Sensirion SHT11 temperature and humidity sensor
  - Intersema barometric pressure sensor.
  - LeadTek 9546 GPS unit.
  - ADXL 202AE accelerometer.
  - Taos TLS257 light intensity sensor.



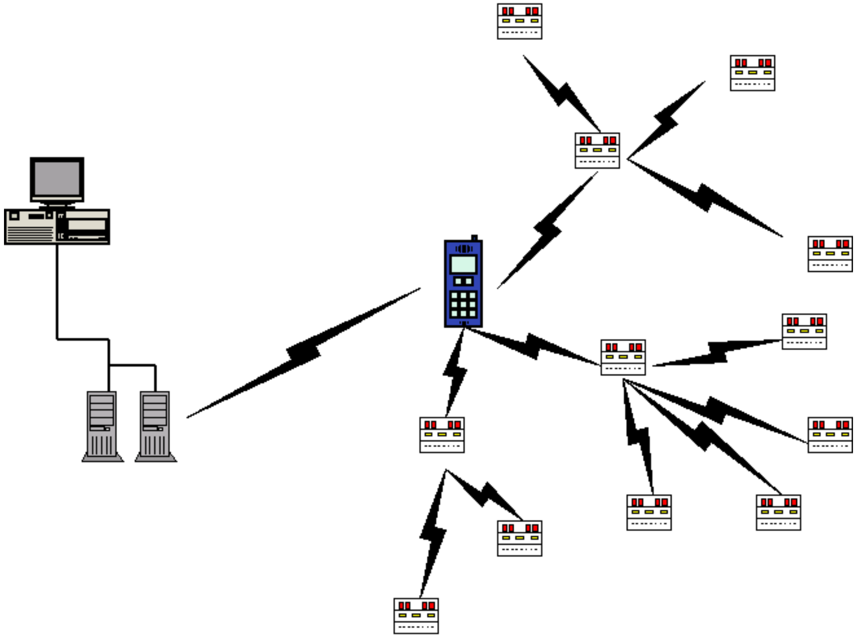


FireBug Base Station

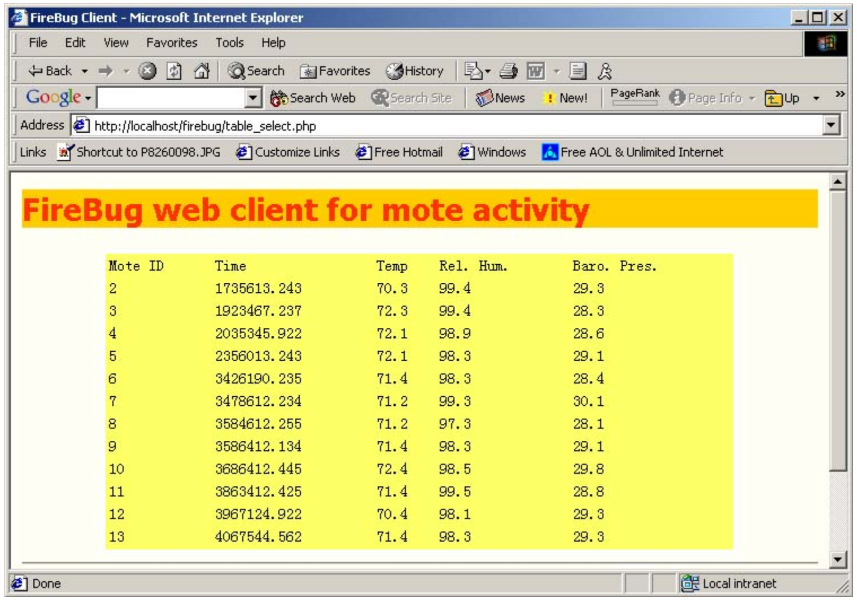


FireBug

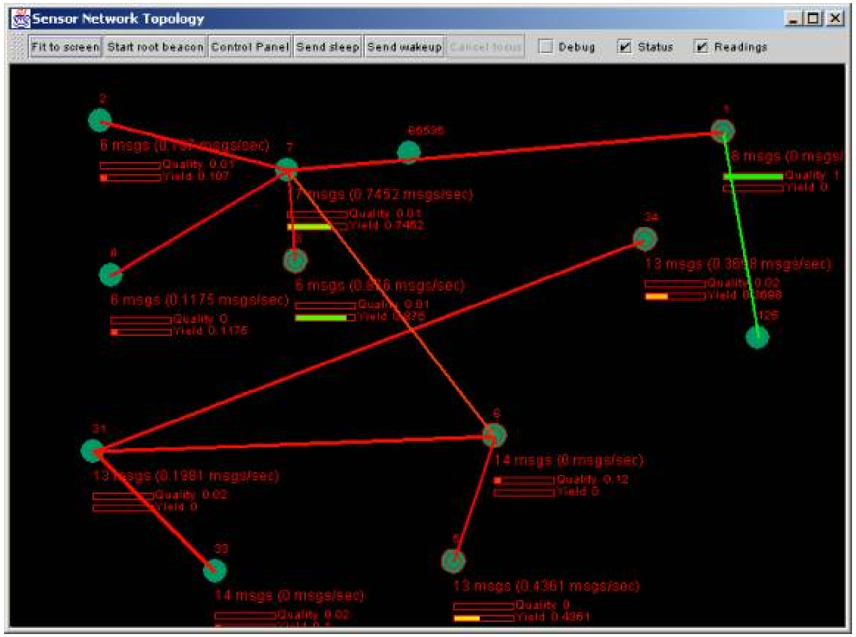
- Software?
  - TinyOS
  - Application set provided by Crossbow
  - Apache web server interfaced with MySQL using PHP



Software application

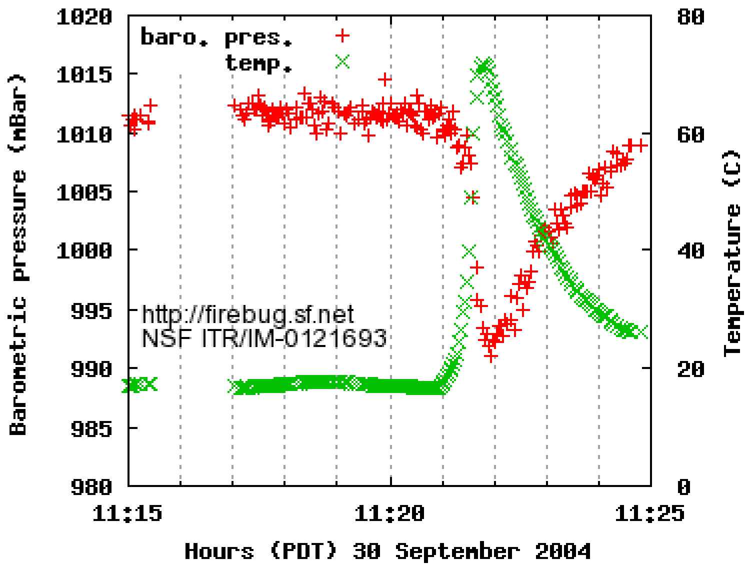


Mote ID	Time	Temp	Rel. Hum.	Baro. Pres.
2	1735613.243	70.3	99.4	29.3
3	1923467.237	72.3	99.4	28.3
4	2035345.922	72.1	98.9	28.6
5	2356013.243	72.1	98.3	29.1
6	3426190.235	71.4	98.3	28.4
7	3478612.234	71.2	99.3	30.1
8	3584612.255	71.2	97.3	28.1
9	3586412.134	71.4	98.3	29.1
10	3686412.445	72.4	98.5	29.8
11	3863412.425	71.4	99.5	28.8
12	3967124.922	70.4	98.1	29.3
13	4067544.562	71.4	98.3	29.3

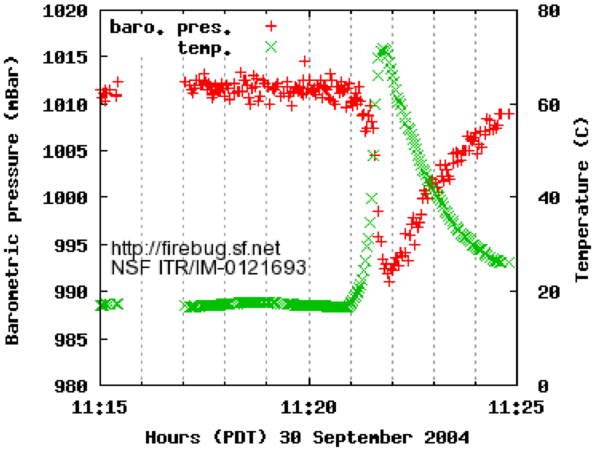


FireBug results

- 10 motes were used in testing
- 3 were down before testing
- 2 were melted during testing
- Fuel and dry grass shortened the transmitting strength



FireBug Results



Calamari

- Goal?
  - a system for providing location information of each node in a sensor field
  - Localized nodes in ad-hoc networks using
    - ultrasound ranging
    - radio signal strength (RSS)
- Where?
  - Where else… Berkeley

http://www.cs.berkeley.edu/~kamin/calamari/

Calamari

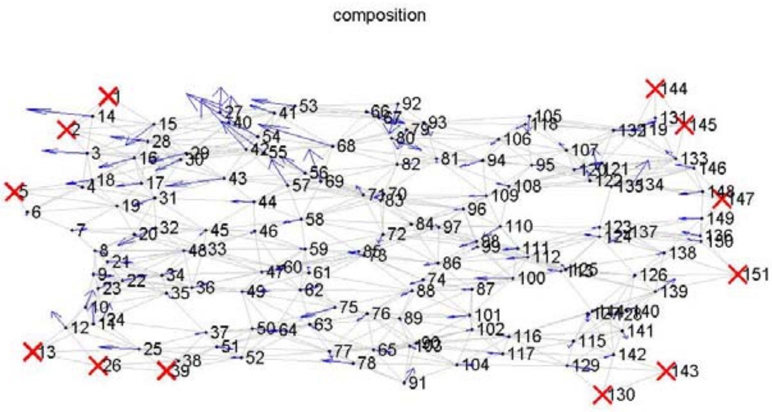
http://www.cs.berkeley.edu/~kamin/calamari/

Calamari

- Hardware?
  - MICA 2
  - Piezoelectric ultrasonic transducers
  - Higher frequencies ->higher accuracy ->shorter range.
  - Just above audible frequency at 25KHz for maximum range while minimizing human irritation
  - 5m using a cone, 12 meter when pointing direct at each other
- Software?
  - TinyOS
- Challenge?
  - Nodes must localize themselves based on information from neighboring nodes which also do not know their positions

Calamari

- High level algorithm ?
  - Ad hoc Positioning System (APS) DV-distance algorithm
  - Unlike GPS, which landmarks are connected directly. It uses hop-by-hop to propagate the distance measurement
  - Randomized grid topology



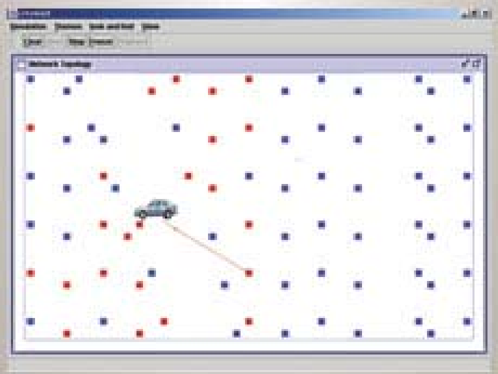
Calamari

- Problem that they have faced
  - Raw ultrasound pulses are not differentiable and collisions between them cannot be detected.
    - Put the ultrasound pulses in a radio message
- Result?
  - Localized nodes in ad-hoc networks using
    - ultrasound ranging
      - 49 node network deployed over a 144 square meter (1600 square foot) area was localized with a median error of .53m (1.7 feet).
    - radio signal strength (RSS)
      - Localized 49 nodes spread over half of a football field with a median error of 4.1 meters.

Exscal

- Where?
  - OSU
- Goal?
  - “A Line in the Sand”

In a demonstration held on the lawn of Chemical Abstracts Service in Columbus, a car is sent through a field of computational devices with the Ohio State middleware software.



Having detected and classified the “intruder,” the path of the vehicle is tracked by the devices and displayed on the screen of a laptop computer. Computational devices that have detected the car are depicted as red squares; other devices on the field are represented by the blue squares.

Where to use it

Exscal

- Detect and track instructor
- XSM, XSS (Stargate), command center
- XSS monitors 20-50 XSM
- 10000 XSM and 300 XSS were built
- 1000 XSM nodes
- 200 ad hoc nodes for data transmission (XSS)

XSM

- Atmel ATmega128L microcontroller
  - a Chipcon CC1000 radio operating at 433MHz
  - a 4Mbit serial flash memory
  - quad infrared
  - dual-axis magnetic
  - acoustic sensors
- Weatherproof packaging
- Asynchronous processor wakeup circuitry.

XSM detection

Sensor	Intruder	Sensing Range
Magnetometer	SUV	7 m
PIR	SUV	30 m
	Person	12 m
Acoustics	ATV	50 m

- 4 Passive Infrared: ~30m for SUV
- Sounder: ~10m
- Microphone: ~5om for ATV
- Magnetometer: ~7-8m for SUV

XSS

- Linux-based Stargate computer
- a GPS unit
- XSSs communicate with each other via high power IEEE 802.11b card
  - connected to a 9dBi antenna
  - 1.82m long
- Over 700m reliable communications in the field at full power.
- XSSs communicate with nearby XSMs via the Chipcon CC1000 radio in a Mica2 that is connected to Stargate

Software

- During deployment:
  - basic confirmation that each node is awake and functioning
    - exercises the sounder each time it is booted, and sends out multiple radio messages containing the node’s unique identifier and network address.
  - power-saving sleep system is immediately enabled after the startup confirmation

Software feature

- This system uses low-power listening, possible to receive radio messages while asleep.
- It is able to dissemination of new programs and uses SNMS for sending commands and collection of health/status

Transmission of data

- XSS network is configured in IEEE 802.11 peer-to-peer ad hoc mode
- XSM uses the default distance-vector routing and queue management protocols in TinyOS
  - only 33.7% of packets from XSMs are delivered to the XSSs on average
  - unreliable wireless links
  - high degree of channel contention where a huge burst of data packets is transported simultaneously
  - Use Logic Grid Rounting

Logical Grid Routing

- Spanning tree bases on BS (XSS)
- BS send “connected message” periodically down the tree
  - Every 20 seconds
  - Size of 2-4 bytes
  - Each node stores no more than 10-15 bytes of routing information
- A node join the one that sends the message
- Switch parents every time a new connected message is received
- After (3*T), decide no longer connected to the tree, stop sending “connected message”
  - load balancing and fast fault recover
    - message go through different path
- Even if 50% fail-stop, 84% still route data

Challenge, that were not solved

- Hardware failure rates (50%) on manufactured XSM
- Noise on sensor chains due to acoustic operation
  - reduce reliable detection by 5 m
- Incorrect operation of XSM on reboot

Reference

- Kamin Whitehouse. **"The Design of Calamari: an Ad-hoc Localization System for Sensor Networks."** Master's Thesis, University of California at Berkeley, 2002.
- http://www.xbow.com/
- http://www.cs.berkeley.edu/~kamin/calamari/
- http://firebug.sourceforge.net/
- http://ceti.cse.ohio-state.edu/exscal/
- http://computer.howstuffworks.com/mote5.htm