Software Engineering for Wireless Sensor Networks
A Case Study
Rachel Cardell-Oliver
Joint work with
Mark Kranz, Keith Smettem, Anna Parsons,
David Glance (UWA), Kevin Mayer (ANU)

Project Goal

• To deliver robust, reactive sensor networks for dynamic environmental monitoring

• Building a sensor network application provides an opportunity to use and evaluate best practice software engineering methods
The Customers’ Problem

The customers want robust tools for monitoring soil moisture dynamically in surface soils.

Stakeholders

- WA Water Corporation (customer-funder)
- Centre for Water Research, UWA (developer-client)
- Computer Science & SE, UWA (developer)
- Agriculture, Government, Urban water users (future customers)
Solutions?

• Current
  – wired monitoring
  – field studies
  – monolithic weather stations
• Future
  – wireless sensor networks for dynamic, untethered environmental monitoring

Wired Monitoring (sap flow)

Problems include: Cost, Power supply, potential wire damage, service requirements
Field Studies

- Expensive in time and people e.g. $20K for one experiment
- Monitored results are only available after field collection

Wireless Weather Stations

- Expensive ($10K to $20K per station)
- Not for fine grained monitoring
- Fixed monitoring regime (e.g. averages taken every 15 minutes)
Wireless Sensor Networks

• Cost effective: $1K per node today but prices to fall
• Fine grained, real-time observations
• Programmable (flexible)
• Results relayed to the lab. immediately

Example: WSN for micro-climate monitoring

Todd Dawson UC Berkeley & Steve Burgess UWA
Requirements

Functional Requirements (V1 2003)
1. Field personnel deploy nodes and base station in the field study area.
3. Nodes will be time-triggered to take measurements.
4. Sensing instruments on nodes collect precise environmental data.
5. The data collected will be the moisture content (or temperature) of the soil in which the node is deployed.
6. At a minimum, the sensor reading, the location of the node, and the time & date are to be recorded.
7. Nodes propagate data to base station.
8. Base station transmits data to persistent storage.
9. Field personnel can add, remove or relocate nodes from within the network.
10. Field personnel can change the power supply on the nodes.
11. System administrator can alter the frequency and timing of measurements taken by each node.
12. System administrator can enquire of the node network information on the health and status of the network.
A requirements problem

- WSNs will be used by plant and soil scientists to gather data that has *not previously been observable*
- So we won’t know the data gathering requirements until we have successfully deployed a network to gather that data
- The requirements set is volatile, and will change as we gain more knowledge of the problem
- Solution: goal based requirements

Stakeholder Goals

1. **Water Corp**: Gnangara soil moisture assessment for Perth Groundwater Project
2. **CWR**: general purpose tools for monitoring water dynamics
3. **CWR**: new results from data on previously unobservable phenomena
4. **CSSE**: discover, understand and test new distributed algorithms for WSNs
Some more goals

5. **Prototype** WSN for same price as a full field experiment (apx $20K)
6. **Reliability** important, but some noise acceptable (current systems are very noisy)
7. Aim for general, reusable solutions – there are many applications for this technology
8. **Delivery time flexible** but ideally ready for the 2004 wet season (from May)
WSN Protocol Design Challenge

- WSN protocols have millions of possible behaviours
- WSN nodes have very limited resources (memory, radio, battery power)
- So, design decisions can have unexpected consequences
- Solution: formally specify and verify protocol designs

Verifying the Design

A rain message being delivered to the SM leaf nodes

Novel development of a reactive sensor system: the system ‘sleeps’ until it rains and wakes up with a signal from a rain gauge
Implementation of the WSN

Technology Choices

• Decision: use existing HW and SW wherever possible
  – risk reduction
  – focus limited development time on our protocols
• Selected Technology
  – UC Berkeley Motes
  – UCLA MDA300 sensor board
  – Echo 20 Soil Moisture probes
  – Superlite Gateway GSM modem and SW (CSIRO)
  – TinyOS software
Connecting a sensor network to the Internet via the GSM network. A network of motes is connected to a Superlite. The reason for using the GSM network is that many sensor network deployments are likely to be 'away from the office'. Connecting the network to a mobile phone network means we have instant Internet connectivity - no infrastructure to worry about!

Kevin Mayer, ANU and CSIRO ICT

TinyOS is an *open-source operating system* designed for *wireless embedded sensor networks*. It features a *component-based architecture* which enables rapid innovation and implementation while minimizing code size as required by the severe memory constraints inherent in sensor networks.

http://webs.cs.berkeley.edu/tos/

---

**Open Source Pros and Cons**  
Spinellis & Szyperski, IEEE Software Jan/Feb 2004

- free, best-of-breed source code  
- the right to make derivatives  
- more than required functionality  
- OS code undergoes continuous improvement  
- code may become isolated from its developers  
- undesired coupling between components  
- widely varying SW quality  
- limited information on testing, processes, release criteria, etc.
A Changing Code Base

- Package: 1.45 MB
tinyos.tinyos-1.x.contrib.s-mac
- Total revisions: 147
- Total lines added: 3,278
- Total lines removed: 1,540
- Average lines added: 22.3
- Average lines removed: 10.48
- Average time between revisions: 6 weeks

- Package: 5.11 MB
tinyos.tinyos-1.x.tos
- Total revisions: 431
- Total lines added: 6,001
- Total lines removed: 5,578
- Average lines added: 13.92
- Average lines removed: 12.94
- Average time between revisions: 4 weeks

http://sourceforge.net/projects/tinyos/
avg time between revisions: min 8 days, max 4 months
currently 26 open bugs reported
first release 2001 (CVS Oct 03)
Verification & Validation

Why is testing WSNs difficult?

• can’t control WSN non-determinism
• can’t control WSN fine timing of events
• code and messages for debugging create probe effects
• only 3 LEDs for mote debugging
• radio RF link properties vary dramatically in different environments
Some Tentative Solutions

- Perform defect testing on the design and try to ensure that the implementation matches it
- Reliability testing
  - good, but only covers a small number of possible scenarios
  - test on site as well as in the lab (RF link changes)
- Regression test suites
  - to re-test code after any changes
  - for both new and for open source SW in the application

SE as Knowledge Acquisition
Armour on Software

• “Software is not a product, but rather a medium for the storage of knowledge.”
• “Systems development can be viewed as the reduction or elimination of ignorance.”
• Problem: “process only allows us to do things we already know how to do”

Phillip Armour, CACM, 2000-2004

Knowledge Acquisition in this WSN project

Sources:
over 100 scientific papers, TinyOS help, UCLA researchers email, UWA researchers (CWR, Viticulture, Salinity CRC), Motorola, CSIRO ICT, UWA students: 3 hons, 3 PhD, 1 practicum, 3 PC307 projects, Crossbow mote users course, Australian mote users workshop, Crossbow, Altronics, Dick Smith, CWR calibration labs, ICT (sensors), Master Instruments (batteries),
**Observations on KA**

- The project is succeeding because of the generous sharing of knowledge from many sources
- Some correct answers were wrong for our context (but we didn’t know that)
- Some answers were just wrong

**Lessons Learnt**

- Requirements: identify stakeholder goals and constraints and plan for reuse
- Design: verify the design to understand the consequences of design decisions
- Implementation: be aware your OS code base will change
- V&V: test new code and reused code both in lab and in field – biggest variable is radio properties
- Other SE Techniques we used: incremental delivery, risk management, lightweight process
Future Work (some for $)

- Support for testing and debugging
- Increase fault tolerance of the WSN
- Increase adaptiveness of the WSN
- Improved web support for monitoring and network management
- SW drivers for sap flow sensors
- Field trial support
- Solar battery trickle recharging
- Mote antenna and radio studies