Reconfigurable Sensor Networks with SOS

Chih-Chieh Han, Ram Kumar Rengaswamy and Roy Shea
UCLA Networked and Embedded Systems Laboratory

Introduction: Remote Reconfiguration in Wireless Sensor Networks

Remote and Inaccessible Deployments

- **Deploy and Leave**
  - Sensors networks observe the otherwise unobservable.
  - Deployments locations are often *hazardous, remote, or fragile.*

- **Quotation From the Field**
  - “Sitting here in the middle of nowhere and getting motes to work... I have to walk for 5 minutes on a mine field (they swear they have removed all the mines but who knows?) just because we forgot to initialize some variable!”

- **Remote Operation**
  - *Operation from afar is essential* to the continued success of sensor networks.

Reconfiguration of a Deployment

- **Post Deployment Updates**
  - Users need to: *add new functionality, remove bugs, and evolve the system* in an actively deployed network.

- **Limits of Physical Reprogramming**
  - Reprogramming a network of 100 or 1000 motes by physically accessing each node is a waste of time and resources.

- **Related Proposals**
  - XNP and *differential updates* load new system images but result in an interrupted sensor operation.
  - *Virtual machines* provide flexibility at the cost of interpreted languages.
  - *Dynamically loadable kernel modules* target a different class of devices.

Proposed Solution: Sensor Network System Support for Loadable Modules

Need a Minimal System Kernel

- **System Core Implements Basic Services**
  - *Message passing scheduler* for messages between parts of the system.
  - Low layer *hardware abstractions* for a given system platform.
  - Fixed-partition *dynamic memory* mechanism to provide constant time dynamic memory allocation to the SOS kernel and modules.

Need Loadable Binary Modules

- **Modules Implement Most Functionality**
  - Everything from sensor drivers to user programs are implemented as modules.
  - Modules can be added, modified, and removed at run time. This creates a *dynamic system* that can change over time to meet new challenges.
  - Modifications to modules does not interrupt system operation.

Solution Analysis: Examining the SOS Operating System

**Inter-Module Communications**

- Priority-based asynchronous messaging for inter-node module communication.
- Type-safe synchronous function call for intra-node module communication.

**System Communications**

- Static kernel interface for access system resources such as timers, memory, sensors, and actuators.

Sample Application

- Testing a sensing application written in SOS. Uses *One Phase Pull* to gather sensor data from a network.
- Application is composed of five modules that can be customized *individually* post deployment.

Competitive Performance of SOS

- **SOS Combines:**
  - *Performance* of traditional wireless sensor network operating systems.
  - *Flexibility* of virtual machines.
  - The combination creates a system that supports larger sensor networks that are able to evolve after they have been deployed.