Experimental performance evaluation of sensor-based networking for energy efficiency in smart buildings

Prof. Sotiris Nikoletseas

University of Patras and CTI, Greece

FIRE Workshop on Measurement and Measurement Tools,

FI week, Aalborg, Denmark, May 2012

HOBNET

Overview

STREP Research Project HOBNET

(FP7-ICT-257466, 2010-2013)

HOlistic Platform Design for Smart Buildings of the Future InterNET

(www.hobnet-project.eu)

(FIRE - Future Internet Research & Experimentation)

Participants

1. Computer Technology Institute (Coordinator), Greece
   Leader: Sotiris Nikoletseas
2. Ericsson Serbia, Serbia
   Leader: Srdjan Krco
3. Mandat International, Switzerland
   Leader: Sebastien Ziegler
4. Sensinode, Finland
   Leader: Zach Shelby
5. University College Dublin, Ireland
   Leader: Antonio Ruzzelli
6. University of Edinburgh, Scotland
   Leader: DK Arvind
7. University of Geneva, Switzerland
   Leader: Jose Rolim

Main Objectives/Expected Results

a. an **all IPv6/6LoWPAN infrastructure of buildings** and how IPv6 can integrate heterogeneous technology (sensors, actuators, mobile devices etc)

b. **6lowApp standardization** towards a new embedded application protocol for building automation

c. novel **algorithmic models and scalable solutions** for energy efficiency and radiation-awareness, data dissemination, localization and mobility

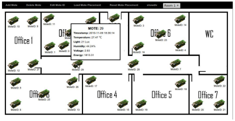
d. rapid development and integration of **building management applications**, and their deployment and monitoring on **FIRE test beds**

Methodological Approach

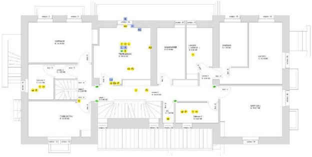
We take a **holistic approach addressing critical aspects at different layers** (networks, algorithms, applications/tools) in an integrated way, including the following hierarchy:

- At the low level, **network protocols and architectures, mainly based on IPv6**, are studied, with an emphasis on heterogeneity and interoperability.
- At a second layer, we provide **algorithmic models and solutions** for smart buildings, with a special care for scalability.
- An interface layer for the rapid development and the evaluation of **building management applications** is provided at a third level.
- Finally, proposed research solutions and key innovations are organically evaluated in the context of the **platform integration**.

The three main test-beds



CTI - Patras test-bed



University of Geneva test-bed

Mandat International test-bed

- High diversity of deployed devices (sensors, actuators, etc.)
- Different thematic emphasis (energy, tracking, visualization, etc.)

List of consolidated scenarios we implement

- Local adaptation to presence
- Emergency management
- Electric device monitoring
- CO2 monitoring
- Maintenance control
- Customization
- Building 3D visualization & monitoring
- Mobile phone ID
- User awareness
- Oil tank monitoring
- Garden watering
- Resources tracking and monitoring

The energy balance property

Guarantees that:

- the average energy spent per sensor *is the same for all sensors* in the network at any time during the network operation
- *prolongs the network lifetime* by avoiding early energy depletion of sensors and the non-utilization of available energy on sensors

Protocols implemented and measured

We implement and experimentally evaluate in our SenseWall test-bed *two energy balancing protocols*:

- P_i Protocol
- E_i Protocol

and compare them to *two pure data propagation schemes*:

- Multi-hop routing protocol
- Direct transmissions protocol

The Distance-based P_i Protocol (designed in HOBNET)

Let i the distance (in hops) to the sink. In each step the algorithm *decides probabilistically and locally* whether to propagate data:

- *one-hop closer to the Sink* with probability P_i
- or send it *directly to the Sink* with probability 1 - P_i

The right probability P_i is rigorously computed to:

$$P_i = 1 - \frac{1}{(i+1)(i-1)}$$

Intuition:

- The closer to the Sink data is, the more probable it is to send data directly to the *Sink* via fast jumps, by bypassing the bottleneck region.
- The larger is the distance to the *Sink*, the more probable multihop data propagation is.

The Energy-based E_i Protocol (developed in HOBNET)

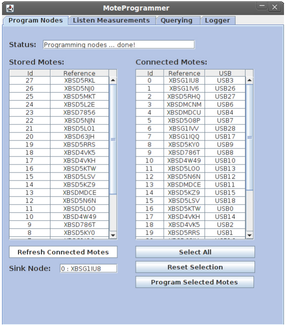
Let *n* be the current sensor and *m* be the sensor in the next hop towards the *Sink* with the lowest energy spent. When *n* holds data it makes the following decision:

- If node *n* has spent more energy than *m*, then *n* sends the message to *m* (spending one energy unit).
- Otherwise, *n* sends the message directly to the Sink spending d^2 energy units, where *d* is the distance from *n* to the *Sink*.

The SenseWall experimental test-bed

- a set of 28 TelosB motes
- a control Base Station PC
- USB cables and hubs
  - + easy control (mass flushing, reseting, etc.)
  - + receive packet-statistics through the wired USB backbone
  - + leave the wireless medium free for the routing algorithms

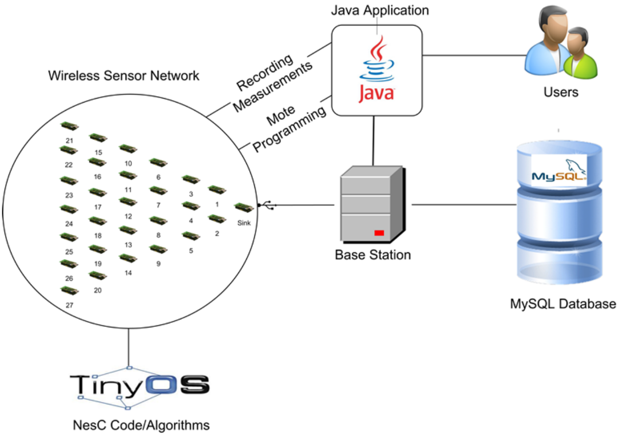
SenseWall: hardware and networking



- We deployed the motes in a sector-shaped topology as shown in the above figure to approximate the theoretical model.
- The *Sink* sends the received messages to the PC using the serial UART interface and then the Java application stores them in a MySQL database.

SenseWall software architecture

The general overview of the architecture is depicted in the following figure:



The MeasurementsLogger Component

Further to implementing in TinyOS 2.1.0 the four routing protocols, we also implemented a MeasurementsLogger Component.

- The **MeasurementsLogger Component** is a binding interface between the desktop software control application and the routing protocols we evaluate in SenseWALL
- The role of this component is:
  1. to setup the parameters of the experiments (i.e. event generation rate, energy sampling rate, experiment duration, etc.) and
  2. to monitor the evolution of the routing protocol by enabling the logging of the performance measurements

Software to control SenseWall

The Desktop application:

- automatically detects the motes that are connected to the desktop PC
- allows the administrator of the network to interact with the motes (event generation rate, reset, etc.)
- provides an interface for computing the following performance evaluation metrics:
    - *data delivery latency*
    - *average energy consumption*
    - *success ratio*

Experimental Setup and Metrics

- Each node generates a total of 1.000 events/messages with a rate of 0.2 events/second.
- Every 50 events (or 250 seconds) a node sends its current energy to the *Sink*.

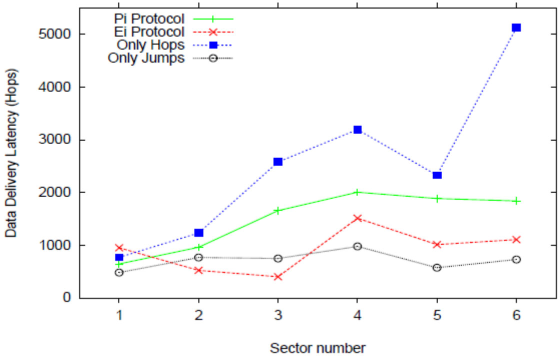
Performance measures:

a) **delivery latency** is the average number of hops needed to reach the *Sink*.

b) **average energy consumption** per node during the network operation.

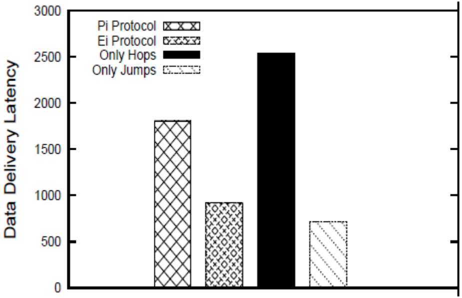
c) **success ratio**, of the total number of packets received by the sink to the number of packets generated.

Experimental Results - Data Delivery Latency



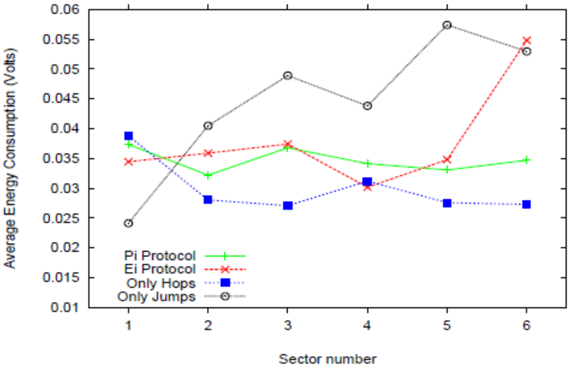
- The multi-hop propagation scheme is *severely affected* by the distance, in terms of latency
- The performance of the E_i and P_i protocols *lies between the two extremes* of multi-hop propagation and direct transmissions

Experimental Results - Data Delivery Latency



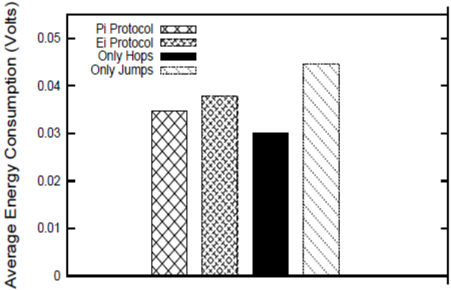
- The energy balance protocols indeed *achieve a good latency-cost trade-off*
- This is due to the hybrid nature of these two protocols transmissions

Experimental Results - Energy Efficiency



- When using direct transmissions to deliver data to the *Sink*, motes lying in distant sectors consume much more energy
- When using multi-hop propagation, motes closer to the *Sink* consume nearly double the energy compared to the rest of the network

Experimental Results - Energy Efficiency



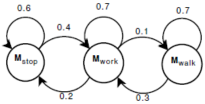
- The P_i protocol *actually manages* to balance the dissipated energy across the network at all sectors
- The E_i protocol consumes slightly more energy than the P_i protocol. This is due to the fact that E_i is more prone to direct transmissions than the P_i protocol

Critical Components - Reference Deployments Used

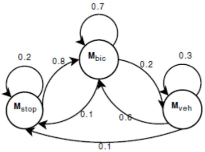
- **structured topologies** (star, grid, mesh) *vs* **randomized deployments** (random proximity graphs and nearest neighbour graphs)
- **homogeneous deployments** (all sensors of the same type) *vs* **heterogeneous deployments** (mix of high and low capabilities sensors)
- **flat deployment** (all sensors at the start) *vs* **incremental deployment** (sensors added during network evolution)
- **uniform node density** *vs* **high density diversity** (e.g. hot spots)
- **static deployments** *vs* **mobile deployments** (and hybrid combinations)

The Mobility Factor

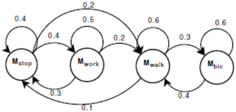
- Emphasis on **highly dynamic, diverse mobility** profiles



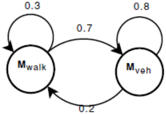
C1: Transitions between slow mobility roles



C3: Transitions for medium mobility with fast bursts



C2: Transitions for medium mobility level



C4: Transitions for fast mobility

- Novel patterns of **accelerated random Sink mobility** needed (inertia random walk, stretched walk, walk with limited memory)
- **Exploiting diverse dynamic node mobility** (new network parameters, like the mobility level) as a low cost replacement of connectivity and fault-tolerance

Critical Issues and Performance Metrics

- **scalability**: how does performance scale with size? Even correctness may be affected by size
  $\rightarrow$ need to evaluate **very large input sizes**
- **fault-tolerance**: can the network tolerate failures well?
  $\rightarrow$ **diverse fault models** needed (temporary/permanent, offline/online, etc.)
- **Inherent trade-offs** (e.g. energy vs time)
  - **Competing goals** / various aspects:
    - minimizing total energy spent in the network
    - maximizing the number of "alive" sensors over time
    - combining energy efficiency and fault-tolerance
    - balancing the energy dissipation
- **Application dependence**
- **Dynamic changes / heterogeneity**

Protocol Properties and Families

- variety of protocols needed/hybrid combinations
- adaptive protocols
- simplicity
- randomization
- distributedness
- locality

Garden Watering - System Description

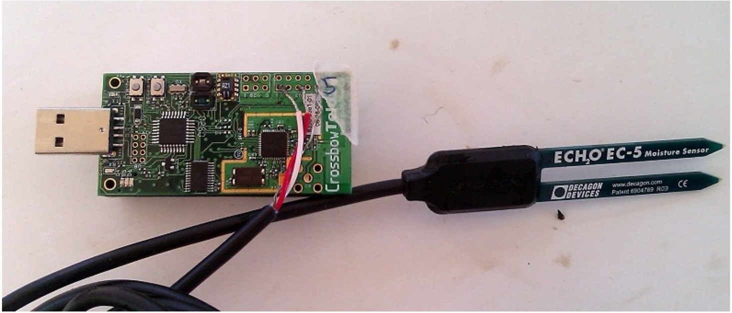
We have implemented a smart irrigation system that is able to adapt:

- to the specific watering needs of different plants/plantations
- to the specific watering needs of each area inside a garden/field
- instantly to diverse weather conditions

The system consists of:

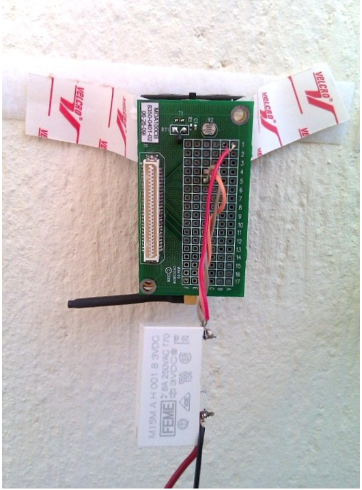
- sensor motes (TelosB & IRIS)
- soil humidity sensors (Decagon EC-5)
- watering electro-valves (Irritol)
- Java app & MySQL database running on a PC

System Description



TelosB motes equipped with EC-5 sensors monitor soil humidity. Measurements are sent to the Sink (PC connected) where they are stored to the database by the Java application. If the soil humidity in a given area falls below a predefined threshold then an alert message is sent to the corresponding IRIS mote to commence irrigation in that area.

System Description



Each IRIS mote, via a relay, drives an external power circuit that controls the irrigation electrovalve. When an alert signal is received the circuit is closed and irrigation starts.

When soil humidity levels reach a predefined upper threshold then a close message is sent from the TelosB to the IRIS motes to stop irrigation.

System Description

- For performance evaluation we use three pots with diverse water needs (geranium, lavender, mint).
- Summertime in Greece: $36^o C$ at daytime, $30^o C$ at night.
- We compared the WSN system to common irrigation programmer

Performance Evaluation

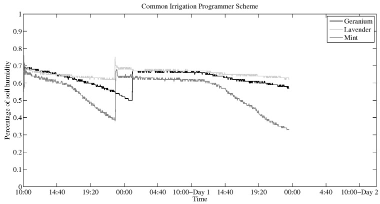


Figure: Common irrigation programmer

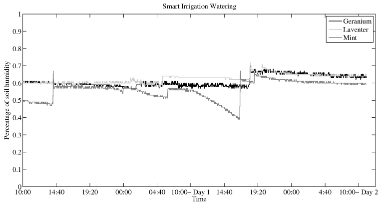


Figure: WSN-based smart irrigation system

- With common programmer there exist great variations in soil humidity.
- Soil dries out and then is flooded with water.
- Dried-out soil, when flooded, withholds much less water.

- Smart irrigation system maintains soil humidity levels
- Less water is dissipated.
- By constantly monitoring soil humidity the system adapts to the watering needs of each plant.
- System is also able to adapt to current weather conditions.

REST Architecture

- REST is an architectural style of building network-based systems
- It is a certain approach to creating Web Services
- Resources: Every distinguishable entity is a resource: anything from a physical device, like a sensor/actuator, to a Web site, XML file, etc..
- URIs Identify Resources: Every resource is uniquely identified by a URI

6LoWPAN

- Low-power RF + IPv6 = The Wireless Embedded Internet
- 6LoWPAN = IPv6 over Low-Power Wireless Area Networks
  - Stateless header compression
  - Enables a standard socket API
  - Minimal use of code and memory
  - Direct end-to-end Internet integration

CoAP

CoAP is:

- A RESTful application protocol
- Both synchronous and asynchronous
- For constrained devices and networks
- Specialized for M2M applications
- Easy to proxy to/from HTTP

HOBNET BWSP (Building Web Server Proxy)

Two-fold contribution:

- Via an Object Building Interface API, it assists the application developer to gather sensor data and control appliances
- Eases the deployment of heterogeneous nodes, management and maintenance of the network by providing the Embedded Building Interface; an abstraction of all embedded services

More information: HOBNET Deliverable D1.3

Conclusions

- Measurements based experimental research nicely complements rigorous performance analysis and simulation based evaluation
- The results are more realistic and can contribute to validating and fine tuning the abstract algorithms
- A large variety of realistic topologies, mobility profiles, traffic patterns is needed
- Novel network parameters as well as performance measures (and their trade-offs) arise
- Ad hoc approaches are useful but there is a need to converge to widely accepted, common integrated approaches, systems and tools