Wireless Sensor Networks for Extreme Environments

Jan Beutel, ETH Zurich
Schedule

- Wireless Sensor Networks – Applications in Environmental Monitoring – PermaSense
- System Design – Low-Power Architectures – Wireless Protocols
- Test + Debugging Methodology – Examples
- End-to-end Data Management – Data Quality Analysis/Control
- PermaSense Demo – Q/A
Wireless Sensor Networks
Applications in Environmental Monitoring – PermaSense
PermaSense – Aims and Vision

Geo-science and engineering collaboration aiming to:

- provide **long-term high-quality** sensing in **harsh environments**
- facilitate near-complete **data recovery** and **near real-time delivery**
- obtain **better quality** data, more effectively
- obtain measurements that have **previously been impossible**
- provide **relevant information** for research or decision making, **natural hazard early-warning systems**
Understanding Root Causes of Catastrophes

Eiger east-face rockfall, July 2006, images courtesy of Arte Television
Rockfall release mechanisms and their connection to time and climate (change) are not understood.
Our patient does not fit into a laboratory
So the laboratory has to go on the mountain.
PermaSense

- Consortium of several projects, start in 2006
- Multiple disciplines (geo-science, engineering)
- Fundamental as well as applied research
- More than 20 people, 9 PhD students
- Nano-tera.ch X-Sense is a PermaSense project

http://www.permasense.ch
PermaSense – Competence in Outdoor Sensing

- Wireless systems, low-latency data transmission
- Customized sensors
- Ruggedized equipment
- Data management
- Planning, installing, operating (years) large deployments
PermaSense Deployment Sites 3500 m a.s.l.

A scientific instrument for precision sensing and data recovery in environmental extremes.
PermaSense – System Architecture

- Sensor network
- Base station
- Backend

Diagram showing the system architecture with components such as Sensor Node, Access Node, Embedded PC, GSM Modem, PV Controller, 5V DC-DC Converter, 12V Battery, PV Cells, Backup GSM, System Monitoring, and Server.

Logos and labels include GSN, Tikk, and ETH Zürich.

Keywords: Computer Engineering and Networks, Technische Informatik und Kommunikationsnetze.
Simple Low-Power Wireless Sensors

- Static, low-rate sensing (120 sec)
- Simple scalar values: temperature, resistivity
- 3 years operation (~200 μA avg. power)
- < 0.1 Mbyte/node/day
- 3+ years experience, ~130’000’000 data points

In relation to other WSN projects
- Comparable to many environmental monitoring apps
  - GDI [Szewczyk], Glacsweb [Martinez], Volcanoes [Welsh], SensorScope [Vetterli], Redwoods [Culler]
- Lower data rate
- Harsher environment, longer lifetime
- Higher yield requirement
- Focus on data quality/integrity
Challenge: The Physical Environment

- Lightning, avalanches, rime, prolonged snow/ice cover, rockfall
- Strong daily variation of temperature
  - $-30$ to $+40^\circ C$
  - $\Delta T \leq 20^\circ C$/hour
Field Site Support

- Base station
  - On-site data aggregation
  - Embedded Linux
  - Solar power system
  - Redundant connectivity
  - Local data buffer
  - Database synchronization

- Cameras
  - PTZ webcam
  - High resolution imaging (D-SLR)

- Weather station

- Remote monitoring and control
WLAN Long-haul Communication

- Data access from weather radar on Klein Matterhorn (P. Burlando, ETHZ)
- Leased fiber/DSL from Zermatt Bergbahnen AG
- Commercial components (Mikrotik)
- Weatherproofed
PermaSense – Sensor Node Hardware

- **Shockfish TinyNode584**
  - MSP430, 16-bit, 8MHz, 10k SRAM, 48k Flash
  - LP Radio: XE1205 @ 868 MHz

- **Waterproof housing and connectors**

- **Protective shoe, easy install**

- **Sensor interface board**
  - Interfaces, power control
  - Temp/humidity monitor
  - 1 GB memory

- **3-year life-time**
  - Single Li-SOCl\textsubscript{2} battery, 13 Ah
  - ~300 \(\mu\)A power budget

Measured avg. current consumption ~148 \(\mu\)A
PermaSense – Sensor Types

- Sensor rods (profiles of temperature and electric conductivity)
- Thermistor chains
- Crack meters
- Water pressure
- Ice stress
- Self potential

Data: Simple sensors, constant rate sampling, scalar values
Established: Rock/ice Temperature

Aim: Understand temperatures in heterogeneous rock and ice

- Measurements at several depths
- Two-minute interval, autonomous for several years
- Survive, buffer and flush periods without connectivity

[Hasler 2011]
Results: Rock/ice Temperatures

Established: Crack Dilatation

Aim: To understand temperature/ice-conditioned rock kinematics

- Temperature-compensated, commercial instrument
- Auxiliary measurements (temperature, additional axes, …)
- Two-minute interval, autonomous for several years
- Protection against snow-load and rock fall
Results: Rock Kinematics

Key PermaSense Challenges

- System Integration
- Correct Test and Validation
- Actual Data
- Interdisciplinary Team
System Design – Low-Power Architectures – Wireless Protocols
Bell’s Law: New Class of Computers Every 10Y

- Number Crunching
- Data Storage
- Productivity
- Interactive
- Streaming information to/from the physical world

- Mainframe
- Minicomputer
- Workstation
- PC
- Laptop
- PDA
- Sensors

[figure abridged from D. Culler]
Wireless Networked Embedded Systems

Highly Resource Constrained

Distributed State

Unreliable Communication

Interaction and Tight Embedding in Environment
Basic Concepts of WSN Platforms

- “Mote class” devices
  - Microcontroller + low-power radio
  - Battery powered
  - Many custom applications
  - Large design space, many variants
  - Most prominent examples: Mica2, Mica2Dot, Tmote Sky

- Hardware is packaged with
  - System software and apps
  - Base stations, network access
  - Server-side solutions (backends)
  - Tools (e.g. simulators, virtual machines ...)

- First anticipation: Small = cheap = low complexity
A Popular Software System – TinyOS

- Event driven “Operating System”
  - Written in nesC, a C dialect
  - Geared towards simple applications (max. 10k RAM, 40 k ROM)
  - Basically just a collection of drivers and an event queuing system
  - Event handlers must not block…

- Compositional nature
  - Hard- and software components
  - Interfaces
  - Modularity

- De-facto standard in WSN applications
  - Popular in academia and industry

- Many (comparable) systems exist: Contiki, SOS, Mantis, BTnut…
WSN Platform Variants
WSN Design and Development Peculiarities

- Interaction with the environment
  - Cannot stop time and use breakpoints

- Visibility of what is going on
  - Getting information in and out of the network

- Low-power modes

-Disconnected medium
  - Wireless is unreliable
  - Split phase operation, interaction across the ensemble

- Lack of comprehensive models/analysis
  - Often there is lot’s of data but no clue what to do with it

**Low-power wireless: Where is the bar?**

110 uA is baseline, 20 uA is research goal

99.5-9% reliability over multiple 70% links

[D. Culler, SenSys 2007]
PermaSense Wireless technology for a laboratory on the mountain
System Design – Architectural Considerations

- **Requirements**
  - Capabilities for multiple sensors
  - Extreme low-power
  - High quality data acquisition (ADC resolution on MSP430 not sufficient)
  - Reliability in extreme environment

- One platform vs. family of devices? Make vs. buy? Existing components?

- **Decisions**
  - Modular, accommodating different sensors on one platform
  - Single, switchable serial bus architecture
  - Strict separation of operating phases (TDMA)
  - Enough storage for disconnected operation over months
  - Temp/Humidity/System voltage supervision in every box
Sensor Node Hardware

- **Shockfish TinyNode584**
  - MSP430, 16-bit, 8MHz, 10k SRAM, 48k Flash
  - LP Radio: XE1205 @ 868 MHz

- **Waterproof housing and connectors**

- **Sensor interface board**
  - Interfaces, power control
  - Temp/humidity monitor
  - 1 GB Flash memory

- **3-year life-time**
  - Single Li-SOCl₂ battery, 13 Ah
  - ~300 μA power budget
Ruggedized for Alpine Extremes
TinyNode + Sensor Interface Board

Extension: One Serial Bus

- (Power) control using GPIO
- Optimized for low-power duty cycling
External Storage Extension

- External SD card memory
- Data buffering
- End-to-end validation

<table>
<thead>
<tr>
<th>DAQ Interval</th>
<th>1min</th>
<th>2min</th>
<th>30min</th>
</tr>
</thead>
<tbody>
<tr>
<td>Byte/day/node</td>
<td>233280</td>
<td>116640</td>
<td>7776</td>
</tr>
<tr>
<td>TinyOS packets /min</td>
<td>7.04</td>
<td>3.52</td>
<td>0.234</td>
</tr>
<tr>
<td>Mbyte/year/node</td>
<td>80.0</td>
<td>40.1</td>
<td>2.64</td>
</tr>
<tr>
<td>Mbyte/3years/20nodes</td>
<td>4805</td>
<td>2403</td>
<td>160</td>
</tr>
</tbody>
</table>

*a 23 byte per TOS packet*
Power Optimization – Squeeze with Implications

- Regulator uses 17uA quiescent current
- Bypass used to shutdown regulator -> ~1uA in standby
- No Bypass increases ADC accuracy: stddev 0.8844 -> 0.0706
Power Quality Increases Data Accuracy

Before

- Crack extension
- Temperature

- std dev = 24.0 μm

After

- Crack extension
- Temperature

- std dev = 1.0 μm
Ultra Low-Power Multi-hop Networking

- Dozer ultra low-power data gathering system
  - Beacon based, 1-hop synchronized TDMA
  - Optimized for ultra-low duty cycles
  - 0.167% duty-cycle, 0.032mA (@ 30sec beacons)

- But in reality: Connectivity can not be guaranteed…
  - Situation dependent transient links (scans/re-connects use energy)
  - Account for long-term loss of connectivity (snow!)

[Burri, IPSN2007]
TDMA System Integration

- System-level, round-robin scheduling
  - “Application processing window” between data transfers and beacons

- Local temperature clock drift compensation

- Separation of communication and application processing
  - Custom DAQ/storage routine per application/sensor
  - Less interference between software components
  - Strict periodicity
  - Timing guarantees

![Diagram showing time slots and application processing window]
1-hop Validation – Simultaneous Power Traces

(1) Receive parent beacon  (2) Send beacon to children  (3) Send data to parent  (4) Sampling sensor (DAQ)
Multihop Concurrency Causes Timing Violations

- In reality due to phase shift timing violations and interference occur
  - Communication arbitration over multiple hops
  - Priority is given to communication
  - Slight variations in periodicity
Global Time Stamping

- No network-wide time synchronization available
  - Implications on data usage

- Elapsed time on arrival
  - Sensor nodes measure/accumulate packet sojourn time
  - Base station annotates packets with UTC timestamps
  - Generation time is calculated as difference $t_g = t_b - t_s$

```
2011/04/14 10:03:31 – 7 sec
= 2011/04/14 10:03:24
```
Sensors Contribute to Power Consumption

- ADC calibration
- Sys
- Volt
- Temp
- Humid
- External ADC channels
- Digital protocol

- Earthpressure cell: 4.58 mA (avg)
- Sensor rod: 3.90 mA (avg)
- Digital sensor: 1.92 mA (avg)
Total Power Performance Analysis

<table>
<thead>
<tr>
<th>Operating Mode Characterization</th>
<th>[mA]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sleep</td>
<td>0.026</td>
</tr>
<tr>
<td>DAQ active(^a)</td>
<td>2.086</td>
</tr>
<tr>
<td>Dozer RX idle</td>
<td>13.64</td>
</tr>
<tr>
<td>Dozer RX</td>
<td>14.2</td>
</tr>
<tr>
<td>Dozer TX</td>
<td>54.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Measured Average Values</th>
<th>[mA]</th>
</tr>
</thead>
<tbody>
<tr>
<td>DAQ only (2min)</td>
<td>0.110</td>
</tr>
<tr>
<td>Dozer only (30sec/2min)(^b)</td>
<td>0.072</td>
</tr>
<tr>
<td>PermaDozer total (30sec/2min)</td>
<td>0.148</td>
</tr>
</tbody>
</table>

\(^a\) Averages power consumption measured over a complete DAQ routine execution without attached sensor  
\(^b\) Dozer only includes communication, not including network initialization and access to flash memory

![Power Consumption Chart]

148 uA average power
Test + Debugging Methodology – Examples
WSN Design and Development

Simulation
- TOSSIM [Levis2003]
- PowerTOSSIM [Shnayder2004]
- Avrora [Titzer2005]

Virtualization and Emulation
- EmStar [Ganesan2004]
- BEE [Chang2003, Kuusilinna2003]

Test Grids
- moteLab [Werner-Allen2005]
- Emstar arrays [Cerpa03/04]
- Kansei [Dutta2005]

Specialized simulation tools for WSN applications

Fast-prototyping in a controlled environment

Closing in on the “real” experience

Figure abridged from D. Estrin/J. Elson
Code Distribution Using A Fixed Testbed

- Traditional test grid
  - Wired backchannel
  - Programming and logging from a remote station
  - Simple centralized control and data collection

MoteLab @ Harvard
- Ethernet-based
- Multi-user arbitration
- ~200 nodes
- Indoor environment

[Werner-Allen - IPSN 2005]
WSN Design and Development

Simulation
- TOSSIM [Levis2003]
- PowerTOSSIM [Shnayder2004]
- Avrora [Titzer2005]

Virtualization and Emulation
- EmStar [Ganesan2004]
- BEE [Chang2003, Kuusilinna2003]

Test Grids
- moteLab [Werner-Allen2005]
- Emstar arrays [Cerpa03/04]
- Kansei [Dutta2005]

Figure abridged from D. Estrin/J. Elson
WSN Testbed – Deployment-Support Network

**DSN Testbed Key Differentiators**
- Distributed observers, local intelligence
- Mobility: Wireless, battery powered

**DSN Testbed Functionality**
- Remote reprogramming
- Extraction of log data
- Stimuli, e.g. fault injection
- Synchronization of traces and actions

**Target Sensor Network**

- Centralized logging
- Detailed behavioral analysis

DSN – A Distributed Testbed with Local State

- Distributed, stateful observer co-located with the DUT
- Capable of capturing multiple context with different granularity
- Problem of “correct” instrumentation is left to the user
Other Approaches – Packet Level Overhearing

- Symptoms are detectable by passive inspection
  - Dead nodes, Node reboots, Network partitions
  - Approximate neighborhood
- Failure root causes remain unclear

[Ringwald DCOSS 2007, Roemer IPSN 2009]
Evaluation With Passive Distributed Assertions

- Assertions: express belief that a condition holds
  
  ```
  int i;
  ...
  assert(i > 50);
  ```

- Assertions over distributed program variables
  - Value of variable i should equal value of variable k at node 100

  ```
  int i;
  ...
  PDA(i = 100:k);
  ```

- Checked by passive inspection
  - Nodes only broadcast assertions (PDA msg), changes of relevant variables (SNAP msg)
  - Overheard by sniffer network
  - Minimize interference

```
k = ...;
SNAP(k);
```
Assertion Evaluation Example

Node 1

Latest snapshot of k on node 2 before T=10

Latest snapshot of k on node 2 before T=10

T=10 i=5; PDA(i=2:k)

Earliest snapshot of k on node 2 after T=10

Message Trace

N=2 S=1 T=6 SNAP(k) k=4

T=10-Δ

N=2 S=2 T=9 SNAP(k) k=5

N=1 S=1 T=10 PDA(i=2:k) i=5

N=2 S=3 T=11 SNAP(k) k=4

T=10+Δ

Earliest snapshot of k on node 2 after T=10

Timesync accuracy ± Δ

Node 2

T=6 k=4; SNAP(k)

T=9 K=5; SNAP(k)

T=11 K=4; SNAP(k)
Validation of Traces using Formal Bounds

- Assertions based on reference traces/specification
- Integrated with each build (regression testing)

$\forall t \in [-\Delta t, \Delta t], \forall i \in \mathbb{N} :$

\[
f^-(t + \bar{t}_k) = \begin{cases} 
  f^-(t_i^-) & \text{if } -f(t_i^-) + f(t_{i+}) \leq 0 \\
  f^-(t_i^+) & \text{if } -f(t_i^-) + f(t_{i+}) > 0 
\end{cases}
\]

The upper bound $f^+$ follows accordingly with a bound value $\Delta y^+$. [WEWSN2008, SUTC2008]
Pitfalls – Dangerous Voltage Drops

\[ V_{cc\_min} = 2.8 \, V \]
Regression Tests Using Continuous Integration

On code change applications are built from scratch and analyzed

- Standard practice in enterprise level software development
- Deeper understanding of long term development trends
- Service to the TinyOS community, increasing software quality

+4500 TinyOS-2.x regression builds over the last 2 years at ETHZ
Quantification of Change with Visualizations
Multilevel Regression Testing Framework

- Multiple levels of abstraction
  - Simulation (e.g. TOSSIM, AVRORA)
  - Emulation (e.g. EmStar)
  - Testbeds (e.g. DSN, moteLab)

- “Test Case” wraps around application code
  - Environment
  - Monitors
  - Stimuli

- Global pass/fail conditions
  - Unit testing
  - Parameter extraction, reporting

[EmNets 2007]
Conformance Testing using Timed Automata
Conformance Testing using Timed Automata

Power trace → Model of observed behavior → UPAAL → Model of expected behavior

\[ \exists \pi \in \prod_{Sys} || PT : s \xrightarrow{\pi} t \Leftrightarrow PT \models Sys \]

System in operation → Expected behavior
WSN Design and Development Tools

Can we Emulate Reality in the Lab?

Simulation
- TOSSIM [Levis2003]
- PowerTOSSIM [Shnayder2004]
- Avrora [Titzer2005]

Virtualization & Emulation
- EmStar arrays [Ganesan2004, Cerpa03/04]
- BEE [Chang2003, Kuusilinna2003]

Test Grids
- moteLab [Werner-Allen2005]
- Twist [Handziski2006]
- Kansei [Dutta2005]

DSN Wireless Testbed

Figure abridged from D. Estrin/J. Elson
Physical Emulation Architecture

- Influence of power sources/quality
- Detailed physical characterization
  - Emulation of environment and resources
    - Temperature Cycle Testing (TCT)
    - Controlled RF attenuation
    - Sensor stimuli and references

[EmNets2007]
Impact of Environmental Extremes

- Software testing in a climate chamber
  - Clock drift compensation yields ± 5ppm
- Validation of correct function
- Tighter guard times increase energy efficiency
Experiences – Behavioral Data from the Field

- Battery Voltage & Temperature
- # Lost Packets
- System Outage
- Loss of Connectivity
3 Months Later – Increasing Runtime Errors

- Node Reboot
- System Outages
- Loss of Connectivity
- # Lost Packets

Graphs showing the occurrence of system outages and lost packets over time.
Understanding the Exact Causes is Hard...

- Missing EMP protector – broken radio chip?
- If so is it snow/wind or lightning induced?
- Long-term software bug – buffer overflow; timing?
- Misunderstood “feature” of the implementation?
- Hardware breakdown?
- Energy-supply dependent?
- Spring-time behavior based on the environment?
- Cosmic rays at 3500 m a.s.l.?
- Can external effects trigger the reed contact on the reset?

... Probably not feasible to test/ reproduce this in the lab!
End-to-end Data Management – Data Quality Analysis/Control
Global Sensor Network (GSN)

- Data streaming framework from EPFL (K. Aberer)
- Organized in “virtual sensors”, i.e. data types/semantics
- Hierarchies and concatenation of virtual sensors enable on-line processing
- Dual architecture translates data from machine representation to SI values, adds metadata

Data Management – Online Semantic Data

Metadata
- Position
- Sensor type
- Validity period

Import from field → GSN → GSN → Web export

Private

Public

GSN

GSN

Web export
Multi-Site, Multi-Station, Multi-Revision Data…
Central Web-based Data Access

PermaSense :: GSN

Welcome to the PermaSense Data Frontend. PermaSense observes physical parameters related to permafrost in steep high-alpine terrain over a period of multiple years. Live sensor network data is transmitted from the Matternhorn and Jungfraujoch field sites at 3500 m a.s.1, every 2 minutes. See the data live on the real-time tabs below or as plots in the data browser.

Sensorrod temperatures (Position 10, 7d)

- Group: jungfraujoch
- Group: matternhorn
  - matternhorn_backlogstatus_chart1
  - matternhorn_backlogstatus_mapped_1355
  - matternhorn_baestationstatus_chart1
  - matternhorn_baestationstatus_chart2
  - matternhorn_baestationstatus_chart3
  - matternhorn_baestationstatus_mapped_1355
  - matternhorn_crackmeter_ntc_1390
  - matternhorn_crackmeter_ntt_1396
  - matternhorn_crackmeter_pos1_chart1
  - matternhorn_crackmeter_pos1_chart2
  - matternhorn_crackmeter_pos2_chart1
  - matternhorn_crackmeter_pos2_chart2
  - matternhorn_crackmeter_pos3_chart1
  - matternhorn_crackmeter_pos3_chart2
  - matternhorn_crackmeter_pos4_chart1
  - matternhorn_crackmeter_pos4_chart2
Value of Online Data

- **Data quality/integrity**
  - Inconsistencies
  - Duplicates
  - Gaps
  - Sporadic/systematic
  - Long-term stability

- **Equipment health**

- **Future: Early warning…**
Challenge: Data Integrity/Validation

- August 2008 – June 2010
- Single sink
- Up to 19 sensor nodes
- TinyOS/Dozer [Burri, IPSN2007]
- Constant rate sampling
- < 0.1 MByte/node/day
Sensor Data Outlier Filtering

- A. Hasler: Threshold-based removal of bogus data, down sampling from 2 to 10 minutes sampling interval
- Tolle et al.: Temperature measurements, outlier rejection based on battery voltage level
- E. Elnahrawy et al.: Bayesian approach for cleaning noisy sensor data
- H. Jeung et al.: Data cleaning with model-based anomaly detector

- Necessary step to mitigate artifacts of faulty sensors
- Usually done by scientific data user/domain expert
- Must assume a certain input data quality
Currently Untouched Artifacts

- We can observe
  - Packet duplicates
  - Node restarts
  - Order inconsistencies
    - Temporal vs. logical
Goals of the Data Analysis

- Validate packets based on a model of the real system
  - For valid packets
    1. Add extra packet ordering information
    2. Provide guarantees on time information
  - Mark other packets as non-conforming
Model of Multi-hop Data Collection

- **Periodic sampling**
  - Sampling period $T$

- **Sequencing**
  - Increasing sequence number
  - Resets on arithmetic overflow

- **Elapsed time on arrival**
  - Sensor nodes measure packet sojourn time
  - Base station annotates packets with UTC timestamps
    - $\tilde{t}_g = t_b - \tilde{t}_s$

$s(i) = i \mod s_{max}$

2011/04/14 10:03:31 – 7 sec

$= 2011/04/14 10:03:24$

$4$ sec

$6$ sec

$2$ sec

$1$ sec

$7$ sec

$t_s(N)$

$\sum \tilde{t}_s$
Error Model

- **Clock drift** $\rho \in [-\hat{\rho}; \hat{\rho}]$
  - Affects measurement of
    - Sampling period $T$
    - Packet sojourn time $t_s$
  - Indirectly leading to ordering inconsistencies
    - Temporal vs. logical

- **Node restarts**
  - Cold restart: Power cycle
  - Soft restart: Watchdog reset

- **Packet duplicates**
  - Lost 1-hop ACK
  - Retransmission

- **Packet loss**
  - Node restart
  - Queue reset
  - Empty queue
Formal System Model (1/2)

Considering a single sensor node with source address $o$:

- Abstract sequence counter: $i$
- At last cold restart: $i_{\text{offset}}$
- Packet sequence number: $s(i) = (i - i_{\text{offset}}) \mod s_{\text{max}}$
- Sampling period: $T$
- Clock drift and resolution: $\rho \in [-\hat{\rho}, \hat{\rho}]$, $\hat{t}_u$
- Packet generation time:
  $$t_g(i) = t_g(i - 1) + \frac{T}{1 + \rho(i)}$$
Formal System Model (2/2)

- Estimated sojourn time on node $N$: $\tilde{t}_s(N, i)$
- Estimated total sojourn time: $\tilde{t}_s(i) = \sum_{N \in \mathcal{N}_{(o, i)}} \tilde{t}_s(N, i)$
- Arrival time at base station: $t_b(i)$
- Estimated generation time: $\tilde{t}_g(i) = t_b(i) - \tilde{t}_s(i)$
- Maximum network diameter: $\hat{h}$

Error bounds on generation time calculation:

$$t_g(i) - \tilde{t}_g(i) \in \left[ -\frac{\tilde{t}_s(i) \cdot \hat{\rho} + \hat{h} \cdot \hat{t}_u}{1 - \hat{\rho}}, \frac{\tilde{t}_s(i) \cdot \hat{\rho}}{1 + \hat{\rho}} \right]$$
Data Processing

Input format: \((o, s, p, \tilde{t}_s, t_b)\)
- Origin \(o\), Sequence number \(s\), total sojourn time \(\tilde{t}_s\), payload \(p\), arrival time \(t_b\)

Output format: \((o, s, p, \tilde{t}_s, t_b, id, [t^l_g, t^u_g])\)
- Unique packet identifier \(id\) reflects temporal order of generation
- Bounds on packet generation time \([t^l_g, t^u_g]\)
Analysis Concepts

- Remove uncertainty caused by sequence number
  \[ s(i) = (i - i_{\text{offset}}) \mod s_{\text{max}} \]
  - Assign packets to epochs
  - Determine unique packet id

- Determine upper and lower bounds on generation time
  - Use forward and backward reasoning

- Remove non-compliant packets
  - Duplicated packets
  - Incorrect time information
  - Behavior not covered by formal model

  problems:
  - arithmetic overflow
  - node restarts

  problems:
  - clock drift
  - node restarts
Bounds on Packet Generation Time

- Worst-case bounds for a single packet
  - $t_g(i) \in [t^l_g(i), t^u_g(i)]$
  - $t^l_g(i) := t_b(i) - \frac{\tilde{t}_s(i) + \hat{h} \cdot \hat{t}_u}{1 - \hat{\rho}}$
  - $t^u_g(i) := t_b(i) - \frac{\tilde{t}_s(i)}{1 + \hat{\rho}}$

- Forward and backward reasoning is applied to tighten these bounds
  - Requirement: Exact ordering information
Duplicate Filtering

- We consider packets with
  - the same source address \( o \)
  - the same sequence number \( s \)
  - an equal payload \( p \)

- We construct a graph \( G = (V, E) \)
  - \( (v, w) \in E \iff (t^u_g(v) \geq t^l_g(w)) \land (t^u_g(w) \geq t^l_g(v)) \)

- Duplicate-free data set is achieved by only considering packets that are within the maximum independent set of \( G \)
Separate Data into Epochs

- Observation: Sequence number $s(i)$ resets to zero
  - Every $s_{max}$ packets due to arithmetical overflow
  - After a cold restart due to loss of state

- After epoch assignment: $id(i) = e(i) \cdot s_{max} + s(i)$
  - $k$ “generated before” $l$ $\Leftrightarrow$ $id(k) < id(l)$
Epoch Assignment (1/3)

- For each packet, calculate a reference point
  \[ T_C(i) = \tilde{t}_g(i) - s(i) \cdot T \]

- Ideal case: Perfect clocks, absence of node restarts
  - Packets belonging to the same epoch have an equal reference point \( T_C \)

- Real case: Imperfect clocks, node restarts
  - Epoch assignment based on bound \( \Delta T_C \)
Epoch Assignment (2/3)

Theorem 1
All packets $k, l$ that belong to the same epoch, i.e., $e(k) = e(l)$, satisfy

$$|T_c(k) - T_c(l)| \leq \Delta T_c$$

where

$$\Delta T_c = (s_{\text{max}} - 1)(\hat{\rho}T + T - T') + T' + 2\hat{\rho} \cdot t_s^{\text{max}}$$

where $t_s^{\text{max}}$ is an upper bound on the network sojourn time, i.e., $t_s(k) \leq t_s^{\text{max}}$ and

$$T' = \frac{1}{\frac{1+\hat{\rho}}{T} + \frac{1}{t_{\text{reset}}}}$$
Epoch Assignment (3/3)

Theorem 2
Suppose that the generation period $T$ satisfies

$$T > 2(1 + \hat{\rho}) \frac{\Delta T_c}{S_{\text{max}}}$$

Then all packets $k$, $l$ that belong to different epochs, e.g., $e(k) < e(l)$, satisfy

$$T_c(l) - T_c(k) > \Delta T_c$$

Where $\Delta T_c$ is defined in Theorem 1.
Forward and Backward Reasoning

- Initially set worst-case bounds $t_g(i) \in [t^l_g(i), t^u_g(i)]$ are often too pessimistic.

- Given the correct order of packet generation, initially set bounds can be improved by using information from temporarily adjacent packets.

- Example: A packet cannot be generated earlier than its predecessor.

$$t^l_g(i) := \max \left( t^l_g(i), t^l_g(i - 1) \right)$$
Matterhorn Deployment Data

Three phases of system operation

A. Initial difficulties with hardware and software
   - Non-conforming system operation

B. Sensor nodes subject to a high number of restarts

C. Daily shut down of base station due to insufficient energy
Model Validation (1/2)

1) Unfiltered Data
   - Duplicate filtering
   - Epoch assignment
   - Violating packets

2) Model-based approach
   - Model
   - # of sequence violations

3) Verified Data
   - Model
   - # of sequence violations
# Model Validation (2/2)

<table>
<thead>
<tr>
<th>Counter</th>
<th>A) Jul 08-Nov 08</th>
<th>B) Nov 08-Aug 09</th>
<th>C) Sep 09-May 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total packets</td>
<td>1,064,884 (100.0%)</td>
<td>2,180,684 (100.0%)</td>
<td>2,703,998 (100.0%)</td>
</tr>
<tr>
<td><strong>I) Unfiltered data set</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sequence violations (Signal model)</td>
<td>189,645 (17.8%)</td>
<td>65,839 (3.0%)</td>
<td>46,987 (1.7%)</td>
</tr>
<tr>
<td>Sequence violations (Ext. storage)</td>
<td>n/a</td>
<td>69,004 (3.2%)</td>
<td>47,253 (1.7%)</td>
</tr>
<tr>
<td><strong>II) Model-based approach</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discarded packets</td>
<td>432,826 (40.6%)</td>
<td>69,829 (3.2%)</td>
<td>124,554 (4.6%)</td>
</tr>
<tr>
<td>Packet duplicates</td>
<td>4,020 (0.4%)</td>
<td>69,422 (3.2%)</td>
<td>44,601 (1.7%)</td>
</tr>
<tr>
<td>( t_s(i) &gt; t_s^{\text{max}} )</td>
<td>0 (0.0%)</td>
<td>0 (0.0%)</td>
<td>0 (0.0%)</td>
</tr>
<tr>
<td>Failed epoch assignment</td>
<td>235,927 (22.2%)</td>
<td>277 (0.0%)</td>
<td>2,466 (0.1%)</td>
</tr>
<tr>
<td>Invalid interval ( t_g^{\mu,\nu}(i) )</td>
<td>192,879 (18.1%)</td>
<td>130 (0.0%)</td>
<td>77,487 (2.9%)</td>
</tr>
<tr>
<td>Accepted packets</td>
<td>632,058 (59.4%)</td>
<td>2,110,855 (96.8%)</td>
<td>2,579,444 (95.4%)</td>
</tr>
<tr>
<td>Sequence violations (Signal model)</td>
<td>7 (0.0%)</td>
<td>11 (0.0%)</td>
<td>0 (0.0%)</td>
</tr>
<tr>
<td>Sequence violations (Ext. storage)</td>
<td>n/a</td>
<td>1 (0.0%)</td>
<td>0 (0.0%)</td>
</tr>
</tbody>
</table>

- Previously “dirty” data set has been restored for use
- Appropriate method for continuous system validation
Conclusions

- Data integrity testing and order reconstruction based on a system model of a real system

- Give guarantees on data quality
  - Duplicate-free data
  - Correct temporal order of generation
  - Correct logical ordering

- Proposed intermediate packet filtering step facilitates the usage of wireless sensor networks for applications that require highest data quality

Case Study Deployments

- **Matterhorn, 3,500m**
  - Since July 2008
  - 19-24 sensor nodes

- **Jungfraujoch, 3,550m**
  - Since February 2009
  - 13-16 sensor nodes

- **Thur, 370m**
  - Since March 2010
  - 5-6 sensor nodes
## Long-term Data Quality

<table>
<thead>
<tr>
<th></th>
<th>Matterhorn</th>
<th>Jungfraujoch</th>
<th>Thur</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Total packets</strong></td>
<td>29,275,461 (100.0%)</td>
<td>27,723,571 (100.0%)</td>
<td>7,189,877 (100.0%)</td>
</tr>
<tr>
<td><strong>Accepted packets</strong></td>
<td>27,786,712 (94.9%)</td>
<td>26,026,120 (93.9%)</td>
<td>7,105,879 (98.8%)</td>
</tr>
<tr>
<td><strong>Packet duplicates</strong></td>
<td>1,462,213 (5.0%)</td>
<td>1,472,664 (5.3%)</td>
<td>83,992 (1.2%)</td>
</tr>
<tr>
<td><strong>Violating packets</strong></td>
<td>26,536 (0.1%)</td>
<td>224,787 (0.8%)</td>
<td>6 (0.0%)</td>
</tr>
</tbody>
</table>
PermaDozer Performance Analysis

<table>
<thead>
<tr>
<th>Deployment</th>
<th>Matterhorn</th>
<th>Jungfraujoch</th>
<th>Thur</th>
</tr>
</thead>
<tbody>
<tr>
<td># of samples</td>
<td>10,347,562</td>
<td>4,805,958</td>
<td>2,785,221</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Link class</th>
<th># of links</th>
<th>$\sigma_R$</th>
<th>$\max \sigma_R$</th>
<th># of links</th>
<th>$\sigma_R$</th>
<th>$\max \sigma_R$</th>
<th># of links</th>
<th>$\sigma_R$</th>
<th>$\max \sigma_R$</th>
<th>$\max \sigma_R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bidirectional communication -- downlink</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Good links (RSSI &gt; -65 dBm)</td>
<td>143 (33.7%)</td>
<td>2.1</td>
<td>7.6</td>
<td>14 (20.6%)</td>
<td>6.3</td>
<td>9.4</td>
<td>20 (31.8%)</td>
<td>3.4</td>
<td>5.0</td>
<td></td>
</tr>
<tr>
<td>Med. links (-65 &gt; RSSI &gt; -85 dBm)</td>
<td>208 (49.1%)</td>
<td>2.9</td>
<td>16.6</td>
<td>40 (58.8%)</td>
<td>3.0</td>
<td>7.0</td>
<td>41 (65.1%)</td>
<td>3.2</td>
<td>9.9</td>
<td></td>
</tr>
<tr>
<td>Bad links (RSSI ≤ -85 dBm)</td>
<td>73 (17.2%)</td>
<td>1.4</td>
<td>4.3</td>
<td>14 (20.6%)</td>
<td>1.5</td>
<td>2.4</td>
<td>2 (3.2%)</td>
<td>2.2</td>
<td>2.8</td>
<td></td>
</tr>
</tbody>
</table>

| Bidirectional communication -- uplink |
| Good links (RSSI > -65 dBm) | 149 (35.1%) | 2.1 | 7.3 | 14 (20.6%) | 5.8 | 8.6 | 17 (27.0%) | 3.5 | 5.6 |
| Med. links (-65 > RSSI > -85 dBm) | 201 (47.4%) | 3.1 | 15.5 | 38 (55.9%) | 3.0 | 6.7 | 44 (69.8%) | 3.3 | 12.6 |
| Bad links (RSSI ≤ -85 dBm) | 74 (17.5%) | 1.7 | 5.7 | 16 (23.5%) | 1.9 | 2.5 | 2 (3.2%) | 1.7 | 2.2 |

- The received signal strength indicator (RSSI) is measured for every successful reception of a packet
- Ratio between signal strength and noise floor
- Higher ratio of duplicates at more challenging environments at Matterhorn and Jungfraujoch

[Matthias Keller, Matthias Woehrle, Roman Lim, Jan Beutel, Lothar Thiele: Comparative Performance Analysis of the PermaDozer Protocol in Diverse Deployments, SenseApp 2011, October 2011, accepted for publication]
PermaSense Demo, Q/A
• ETH Zurich
  – Computer Engineering and Networks Lab
  – Geodesy and Geodynamics Lab
• University of Zurich
  – Department of Geography
• EPFL
  – Distributed Information Systems Laboratory
• University of Basel
  – Department Computer Science

Interested in more?
http://www.permasense.ch