WalT: testing sensor networks and distributed environments

E. Dublé, F. Rousseau, J. Baranguán, J. Zeni, I. Juc,
P.-H. Frohring, B. Faure

April 7, 2014

LIG / DRAKKAR
The problem

1 The problem

2 The WalT project

3 WalT: how it works

4 Demos
The problem

How to efficiently debug a distributed system?

Sensor networks: even harder (constrained devices)
WSNs: existing solutions

- **Senslab - IoT-Lab**, other public platforms:
  - Pros: many nodes, readily available
  - Cons: remote access, not very flexible (e.g. you cannot replace a sensor)

- "Hand-made" one-time experiment:
  - Pros: on your own desk, very flexible, local setup
  - Cons: time-consuming setup, not conceivable for many nodes
1 The problem

2 The WalT project

3 WalT: how it works

4 Demos

WalT: testing sensor networks and distributed environments
The WalT project

The WalT approach

- **WalT**: Wireless Testbed

- Intermediary approach:
  - *Local* in your lab or company
  - *Open source* testbed management software
  - *Open documentation* and hardware recommendations

- You buy the (cheap) infrastructure, the project provides the rest of the stack
Walt nodes are built on a cheap (around 35€) Single-Board-Computer called *Raspberry Pi*\(^1\)

Walt nodes boot an OS stored on the LAN. Depending on the use case, it may be:

- Your choice of *Linux* distribution
- *Android* (if you simulate a network of smartphones)

Walt nodes are linked together using a Power-Over-Ethernet network

\(^1\)We may evolve to another platform in the future if needed / valuable.
The WalT project

WalT scenarios

1 **Sensor networks:**
   connect 1 or more sensors to each RPi node
   - The RPi is used as a gateway to communicate with the sensor and control it (flash, reboot)
   - Useful to test WSN protocols

2 **Distributed systems:**
   use the RPi nodes themselves
   - Useful to test data management systems, TCP congestion avoidance optimizations, etc.
The WalT project

Founding

- **AGIR (UJF/INP) 2013/2014**: Hardware, trainees
- **ST**: Sensors, industrial contract (up to end 2013)
Contents

1. The problem

2. The WalT project

3. WalT: how it works

4. Demos
WalT: architecture (scenario 1)
Collecting sensor traces...
Collecting sensor traces...

trace from sensor X: **sending** <packet>
WalT: how it works

Collecting sensor traces...
Collecting sensor traces...

trace from sensor Y: received <packet>
trace from sensor Z: received <packet>
WalT: how it works

...and then:

- The traces are analysed by the server (e.g. "which sensors received this packet?")
- The traces are then directed to the VizWalT GUI
WalT: how it works

WalT GUI: VizWalT

A modified simulator: **cooja**

- VizWalT reuses cooja's interface

But:

- In an unmodified cooja: each node is *smart*
- In VizWalT: each node *reflects* the behaviour of a real sensor
Technical challenges

- Sensors send traces through their **serial link**, which is slow (slower than the radio!)
  - It may disturb the behavior of the sensor.
  - Solutions being implemented:
    - Send only one small part of the packet (enough to identify it).
    - Improve the serial link driver (use DMA).

- **Synchronization** issues (being explored)

- **Cooja-related** issues (real-nodes versus emulated-nodes paradigm)
Contents

1 The problem
2 The WalT project
3 WalT: how it works
4 Demos
Demos

1. Preliminary version of a **WalT tesbed management** tool
2. Sample **WalT session with DSME nodes**
The end...

Questions?