Demo Abstract: A Testbed Management Architecture for Wireless Sensor Network Testbeds (TARWIS)

Philipp Hurni, Gerald Wagenknecht, Markus Anwander, Torsten Braun

Institute of Computer Science and Applied Mathematics
University of Bern, Switzerland
hurni, wagenknecht, anwander, braun@iam.unibe.ch

Abstract. TARWIS is a flexible and generic Testbed Management System for Wireless Sensor Network Testbeds - a re-usable management solution for research and/or educational oriented research testbeds of wireless sensor networks. TARWIS offers a comprehensive web-based graphical user interface, over which it provides its testbed management services. These include multi-user access to testbed resources, online experiment configuration and scheduling, data acquisition and logging as well as real-time experiment observation.

1 Introduction

All over the world, researchers have set up small wireless sensor network testbeds for research purposes, in order to test and evaluate the real-world behaviour of developed protocol mechanisms. A large number of testbeds have been put into operation, each with different equipment and testbed architecture design (e.g. MoteLab, Kansei, PowerBench, JAWS-DSN, DES-Testbed). The popularity of wireless sensor networks is increasing, and many researchers are setting up and deploying their own new testbeds. Although each testbed may differ with respect to hardware and software, all wireless sensor network testbeds require common functionalities. As every shared resource, a testbed needs a notion of users, it requires support for reprogramming and reconfiguration of the nodes, provisions to debug and remotely reset sensor nodes in case of node failures as well as a solution for collecting and storing experimental data.
TARWIS targets at providing these functionalities independent from the node type and node operating system. The system has been developed within the WISEBED Project (EU FP7, 2008-2011) and has been designed to let researchers access testbed resources remotely over the Internet, in order to share testbed resources with European research partners in a federation of testbeds. TARWIS hence relieves researchers setting up a sensor network testbed from the burden to implement their own scheduling and testbed management solutions.

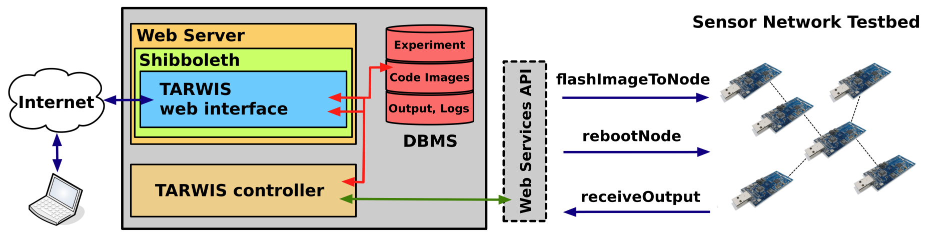


Fig. 1: TARWIS System Architecture

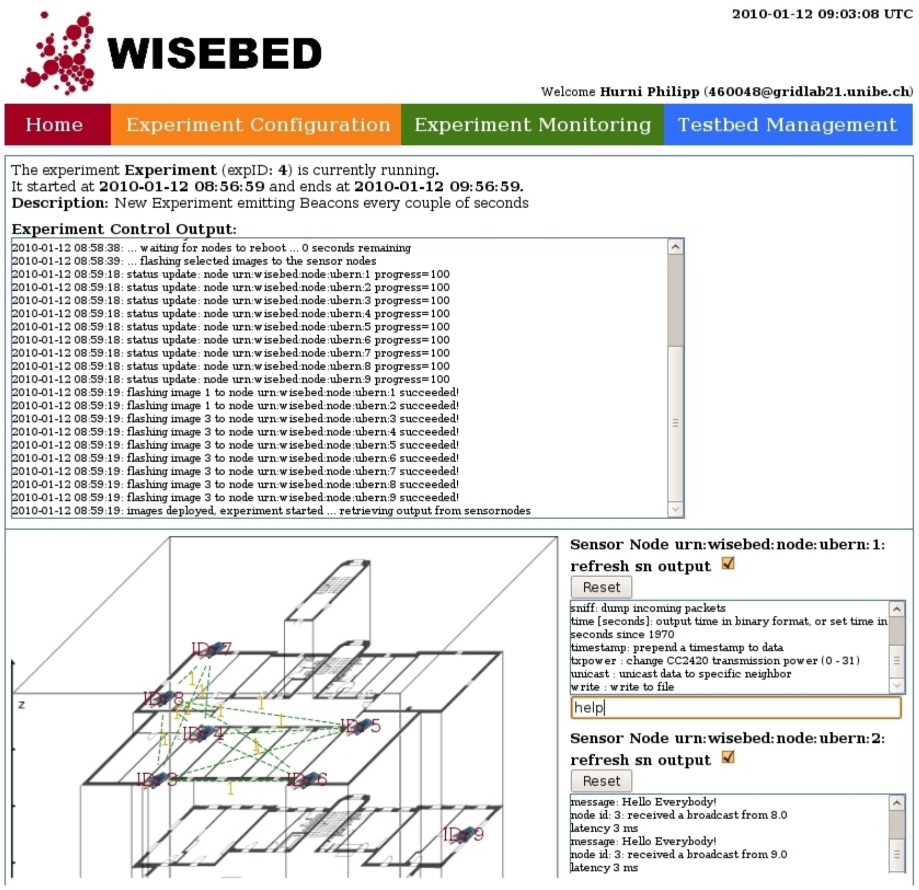
2 TARWIS Architecture

The architecture of TARWIS is depicted in Figure 1. The Figure displays a portal server on which the essential parts of the testbed management system is running. Any user can access TARWIS over the Internet. The portal server hosts the TARWIS web interface, which is protected by the Authentication and Authorization System Shibboleth (www.shibboleth.internet2.edu), a de-facto standard for user and account management in distributed systems, which is widely used in European education and research networks.

The TARWIS controller daemon controls the experiment execution and connects to the individual sensor nodes. It remains independent of the type and the operating system of the sensor nodes, however relies on the presence of an API of WebService primitives in order to access the testbed resources, manipulate the sensor nodes and to retrieve run-time experiment data. This set of functions needs to be implemented as WebServices, hence providing a language- and operating-system-independent interface to the resources of the sensor network testbed. Figure 1 displays three examples of such primitives, which are invoked by TARWIS: reprogramming a node using *flashImageToNode*, rebooting a node using *rebootNode* and retrieving output from a node *retrieveOutput*. A database management system is further used for saving experiment definition data (e.g. which nodes participate in the experiment, the code image they need to be reprogrammed with, etc.) and for storing experiment data and log traces during the experiment.

3 TARWIS Graphical User Interface

The TARWIS user interface is implemented as a dynamic webpage, offering the user access to the testbed via web browser. Users can upload and administrate locally-compiled binary sensor node code, inspect and configure nodes to set up an experiment and schedule the experiment. The TARWIS user interface conveniently guides the user through the workflow to define experiment resources and the preferred time for experiment scheduling.
As soon as the TARWIS controller executes the experiment and invokes the WebService-based primitives, the user can monitor the ongoing experiment by



Welcome **Hurni Philipp (460048@gridlab21.unibe.ch)**

The experiment **Experiment** (expID: **4**) is currently running.
It started at **2010-01-12 08:56:59** and ends at **2010-01-12 09:56:59.**
Description: New Experiment emitting Beacons every couple of seconds

Experiment Control Output:

```
2010-01-12 08:58:38: ... waiting for nodes to reboot ... 0 seconds remaining
2010-01-12 08:58:39: ... flashing selected images to the sensor nodes
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:1 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:2 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:3 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:4 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:5 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:6 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:7 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:8 progress=100
2010-01-12 08:59:18: status update: node urn:wisebed:node:ubern:9 progress=100
2010-01-12 08:59:19: flashing image 1 to node urn:wisebed:node:ubern:1 succeeded!
2010-01-12 08:59:19: flashing image 1 to node urn:wisebed:node:ubern:2 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:3 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:4 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:5 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:6 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:7 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:8 succeeded!
2010-01-12 08:59:19: flashing image 3 to node urn:wisebed:node:ubern:9 succeeded!
2010-01-12 08:59:19: images deployed, experiment started ... retrieving output from sensornodes
```

Sensor Node urn:wisebed:node:ubern:1:
refresh sn output ✓

```
sniff: dump incoming packets
time [seconds]: output time in binary format, or set time in
seconds since 1970
timestamp: prepend a timestamp to data
txpower : change CC2420 transmission power (0 - 31)
unicast : unicast data to specific neighbor
write : write to file
```

help

Sensor Node urn:wisebed:node:ubern:2:
refresh sn output ✓

```
message: Hello Everybody!
node id: 3: received a broadcast from 8.0
latency 3 ms
message: Hello Everybody!
node id: 3: received a broadcast from 9.0
latency 3 ms
```

Fig. 2: TARWIS Graphical User Interface

observing the output of the selected sensor nodes. Furthermore, the user can interact with the sensor nodes remotely by sending commands to them via TARWIS web interface, as displayed in Figure 2. In addition, the TARWIS GUI displays the nodes' connectivity on the node map (cf. Figure 2, bottom left), as soon as nodes transmit packets and discover each other.

4 TARWIS Data Management

Throughout the entire experiment time, all experiment data is retrieved and stored for offline analysis. At the end of an experiment, the experiment results (consisting in arbitrary string-based and binary output) is stored in an XML file and saved to the TARWIS database. The user who scheduled the experiment receives a notification via email as soon as the experiment time expires. He/she is henceforth given the opportunity to download the experiment results and logged data via the web-based user interface.