Using SensLAB* as a First Class Scientific Tool for Large Scale Wireless Sensor Network Experiments

C. Burin des Roziers[2], G. Chelius[2,1,5], T. Ducrocq[2], E. Fleury[1,2,5], A. Fraboulet[3,2,5], A. Gallais[4], N. Mitton[2], T. Noel[4], and J. Vandaele[2]

[1] ENS de Lyon. 15 parvis René Descartes - BP 7000 69342 Lyon Cedex 07 - FRANCE
[2] INRIA – firstName.lastName@inria.fr
[3] INSA de Lyon – firstName.lastName@insa-lyon.fr
[4] Université de Strasbourg – firstName.lastName@unistra.fr
[5] Université de Lyon

Abstract. This paper presents a description of SensLAB(Very Large Scale Open Wireless Sensor Network Testbed) that has been developed and deployed in order to allow the evaluation through experimentations of scalable wireless sensor network protocols and applications. SensLAB's main and most important goal is to offer an accurate open access multi-users scientific tool to support the design, the development tuning, and the experimentation of real large-scale sensor network applications. The SensLAB testbed is composed of 1024 nodes over 4 sites. Each site hosts 256 sensor nodes with specific characteristics in order to offer a wide spectrum of possibilities and heterogeneity. Within a given site, each one of the 256 nodes is able both to communicate via its radio interface to its neighbors and to be configured as a sink node to exchange data with any other "*sink node*". The hardware and software architectures that allow to reserve, configure, deploy firmwares and gather experimental data and monitoring information are described. We also present demonstration examples to illustrate the use of the SensLAB testbed and encourage researchers to test and benchmark their applications/protocols on a large scale WSN testbed.

Keywords: Wireless Sensor Network, Testbed, Radio, Monitoring, Experiments

1 Introduction

Wireless sensor networks (WSN) have emerged as a premier research topic. In the industrial domain, wireless sensor networks are opening up machine-to-machine (M2M) communications paradigms. However, due to their massively distributed nature, the design, the implementation, and the evaluation of sensor network applications, middleware and communication protocols are tedious and really time-consuming tasks. It appears strategic and crucial to offer to researchers and

* Supported by the French ANR/VERSO program. http://www.senslab.info

developers accurate software tools, physical large scale testbeds to benchmark and optimize their applications and services. Simulations remain an important phase during the design and the provisioning step. However, they suffer from several imperfections as simulation makes artificial assumptions on radio propagation, traffic, failure patterns, and topologies [6, 8]. As proposed by initiatives in Europe and worldwide, enabling "*open wireless multi-users experiment facility testbeds*", will foster the emergence of the Future Internet and would be increasingly important for the research community. There is an increasing demand among researchers, industrials and production system architects to access testbed resources they need to conduct their experiments.

In this paper, we describe SensLAB, an open access multi-user WSN testbed (see Figure 1), which has been designed and deployed to answer all these needs (Section 2). SensLAB provides appropriate tools, methods, experimental facilities for testing and managing large scale wireless sensor network applications. As such, it lowers the entry cost to experimentation, often considered as a complex and heavyweight activity, with no extra management burden, accelerating proof-of-concept evaluation and competitiveness. We first describe the hardware and software architectures of the platform (Section 3), then we show how easy and efficient it is to use it through a couple of application examples (Section 4).

2 Context and Design Requirements

One barrier to the widespread use of wireless sensor networks is the difficulty of coaxing reliable information from nodes whose batteries are small, whose wireless medium is sporadic, etc. It is thus very important to conduct *in situ* experiments and researches to better understand the characteristics and compensate for some of these flaws and reach the state of maturity to make them practical. Unfortunately, the development and testing of real experiments quickly become a nightmare if the number of nodes exceeds a few dozens: *(i)* sensors are **small** devices with very **limited capacities** in terms of debugging and friendly programming; *(ii)* software deployment and debugging require the connection of the device yielding to **individual manipulations** of each single node; *(iii)* sensors are generally powered by a **battery** which has limited lifetime, etc.

When the SensLAB project was initiated in 2005/2006, the number of WSN testbeds was limited or did not met the requirements we did have in mind (independence from any specific OS/language, accurate consumption monitoring, reproducible experiments). Nowadays, other WSN testbeds like moteLab [10][6], Kansei [1][7], WASAL[8] or TWIST [4] exist. Our goal is not to compare here all the features of all platforms. Note that it will however really valuable to do so in order to gather all functionalities available in various systems and ease future large scale federations. The Kansei testbed runs 210 Extreme Scale Motes (XSM) hooked individually onto 210 Extreme Scale Stargates (XSS). Kansei provides

[6] `http://motelab.eecs.harvard.edu/`
[7] `http://ceti.cse.ohio-state.edu/kansei/`
[8] `http://wasal.epfl.ch/`

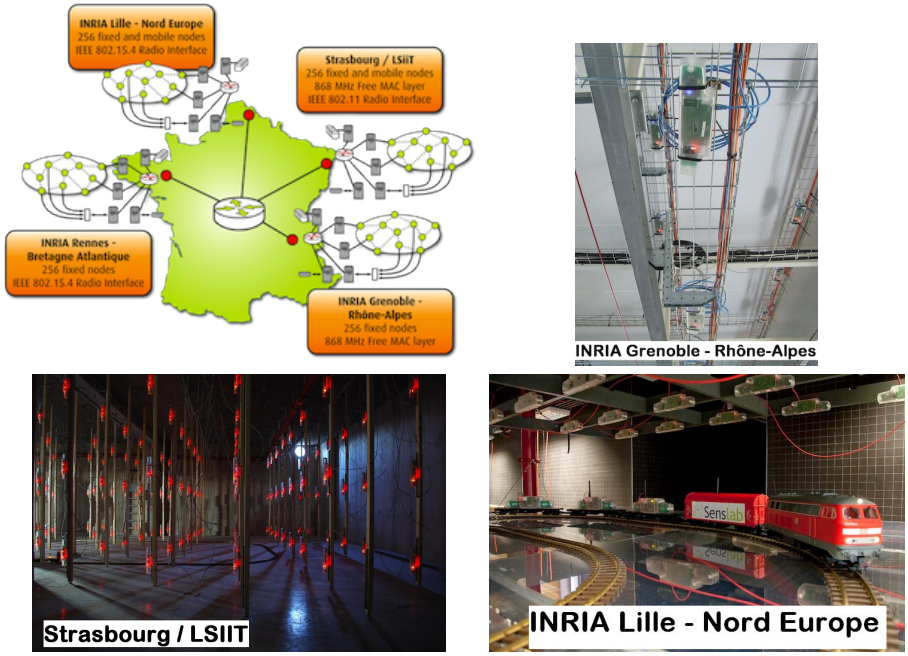


Fig. 1. SensLAB testbed is composed of 4 distributed WSN interconnected by Internet (1024 nodes in total). Mobile nodes are available on Strasbourg and Lille thanks to several toy electric trains remotely controllable and to supply power to the nodes. There is no fundamental limitation on the number of sites or number of nodes per site.

a testbed infrastructure to conduct experiments with 802.11b networking and XSMs. For some testtbeds specific operating system (*e.g.* TinyOS), MAC layer (*e.g.* IEEE 802.15.4) or/and program language (*e.g.* NesC) are imposed, which drastically limit optimization possibilities. Some platforms do not provide radio or power instrumentation (*e.g.* WASAL) nor noise injection. Recently, the WISEBED[9] project shows the ambition to federate several WSN testbeds in order to establish a European wireless sensor network. It seems that application should be developed by using a specific API dedicated to the WISEBED platform. A great benefit of the WISEBED project outcomes is the release of Wiselib, an algorithm library for sensor networks (localization, routing) since it pursues a common goal of lowering the accessibility investment in terms of time for WSN application development .

We propose to address the problems listed above by operating SensLAB as an *open* research facility for academic and industrial groups. SensLAB provides a research infrastructure for the exploration of sensor network issues in reproducible experimental conditions. The platform is **generic**, **open** and **flexible**:

[9] `http://www.wisebed.eu/`

it means that a user is able to remotely access (web access) and deploy his/her applications *without any kind of restrictions on the programming language, on the programming model or on the OS that he/she desires to use*. SensLAB provides an easy way to set an experimentation, to let the user choose the number of nodes, sensor & radio characteristics, topology considerations, experimentation time, etc. SensLAB also integrates an efficient reservation tool to schedule different experimentations. During a experiments running, users have an online access to his/her nodes (either by the web or by a command line shell) at anytime.

The SensLAB architecture must satisfy several strong requirements in terms of software and hardware: *(i)* **reliable access** to the nodes in order to perform operations such as a reset or firmware flashing whatever is the state of the sensor node or the software it is running. *(ii)* non intrusive and application transparent **real time monitoring** of each sensor node. The external monitoring (*i.e.,* totally independent from the deployed user application code) will include precise and real-time access to fundamental parameters such as energy consumption and radio activity; *(iii)* **real time control** of the experiments by providing a set of commands that may influence an application environment (*e.g.*, turn on/off nodes to mimic crashes, emit radio noise by sending fake data in order to tamper with transmissions, modify the monitoring frequency parameters).

3 Main Elements of SensLAB

Figure 2 gives an overview of the testbed services. The user sets his/her experiment through a webportal. A virtual machine is setup with all the development tools and chains preconfigured (cross compilation chains, OS, drivers, communication libraries). The user can also access and use higher level development and prototyping tools (like cycle accurate hardware platform simulator / radio wireless network simulator [3]). When an experiment is launched, specific SQL tables related to the experiment are created. All monitoring data collected during an experiment are stored in tables to support subsequent analysis. The user is thus able to perform postmortem analysis but the system also provides online data analysis services (OLAP). Each service is replicated on each site in order to be fault tolerant (DNS for users virtual machines, LDAP for authentication...). Fig. 3 details all the software components deployed on each site.

3.1 SensLAB Hardware Components and Infrastructures

The SensLAB hardware infrastructure consists of three main components: *(i)* The **open wireless sensor node** dedicated to the user, *(ii)*; The full **SensLAB node** that encompasses the open node also includes a gateway and a closed wireless node; *(iii)* The global networking backbone that provides power and connectivity to all SensLAB nodes and guaranties the out of band signal network needed for command purposes and monitoring feedback.

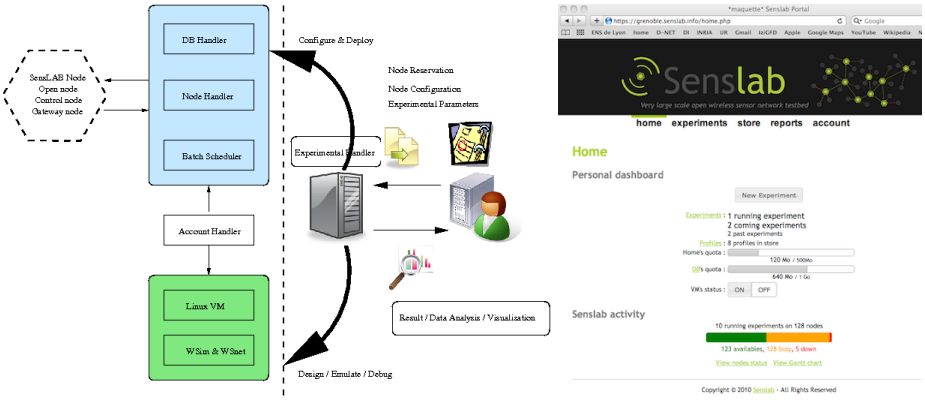


Fig. 2. Simplified view of the platform usage/services (left). SensLAB portal (right).

The **open wireless sensor node** is made available to the user during his/her experimentation. This node is totally open and the user is granted a full access to the memory. The current trend for wireless sensors nodes is geared toward a common architecture based on off the shelf 16-bit micro-controllers. We thus clearly target low power wireless sensor nodes constrained in memory and energy like existing products already on the market[10]. In order to meet with the requirements in terms of energy monitoring, reproducibility, we need to master the architecture and a solution has been to design our own board, named WSN430, in order to include all control signals and thus guarantee a reliable control and feedback[11]. The nodes are based on a low power MSP430-based platform, with a fully functional ISM radio interface[12].

To control the *open* WSN430 node that the user will request and use, we choose to mirror it with another WSN430 whose specific role is to control the open one. In order to link the two WSN430 nodes and also to meet with all mandatory requirements listed previously, we design the SensLAB gateway board in order to insure the control and management of the platform: Automated firmware deployment on open node; Accurate power monitoring of open nodes, both on battery and DC power supply (measure precision is 10μA, and power sampling around 1kHz); RSSI measures and noise injection; Configurable sensor polling on the control node (temperature, light, acoustic activity); Fixed (Ethernet) as well as mobile (Wifi) communication with Node Handler via Connect EM module for the Ethernet version (or a Digi Connect Wi-EM for the Wifi version[13]). These modules embed an ARM7-based module plus Ethernet

[10] WiEye, Micaz, Tmote-Sky, TinyNodes
[11] All designs are released under a Creative Commons License.
[12] Two versions are currently available: an open 868MHz radio interface and IEEE 802.15.4 radio interface at 2.4GHz.
[13] These 2 modules are pin-to-pin compatibles, allowing only 1 unique board design, with either an Ethernet or a Wifi module on it.

or Wifi specialized chips; Power over Ethernet support for a standardized and easy power management; Sink capability for each open node (in/out characters stream redirection). The trade off made was to design our own SensLAB gateway instead of using linux box.

3.2 SensLAB Software Architecture

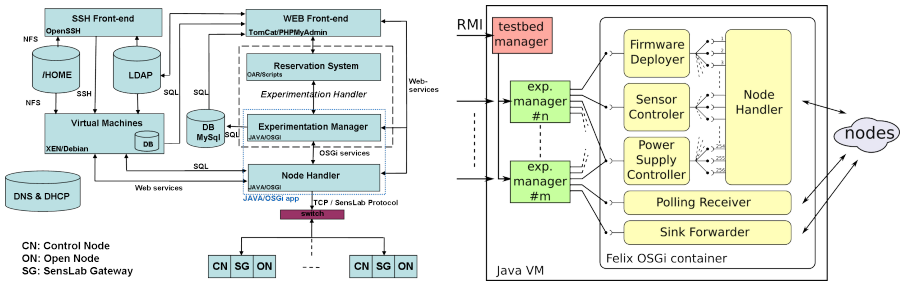


Fig. 3. Software SensLAB architecture and technological choices and the Experiment Handler software structure.

The SensLAB software architecture is replicated over the four testbed sites, and it is divided in several parts which interact together as shown in Figure 3. We detail below their functionalities:

Control node software: *i.e.* the firmware running on the control node. It is in charge of powering up/down, resetting and monitoring the open node activity (power consumption, radio activity/RSSI). All these actions can be executed either asynchronously on the user's request, or automatically and periodically. The control node can also be used to set the ADC pins of the open nodes, allowing to send specific stimuli to the open nodes and thus guarantying a totally reproducible environment on the sensing part. By reproducible environment, we mean that it is possible to record a trace of events/values on a ADC sensor device and during another experiment to "*replay*" such values instead of reading real ones. It allows to change application parameters without changing the environment sensed, and thus allows to compare things that are comparable.

Gateway node software: *i.e.* the firmware running on the gateway. It manages the interface between the open and control nodes, and the SensLAB site server over IP communications. It forwards the command frames addressed to the control node, updates the open node's firmware (BSL protocol), and forwards the open node's serial link to the server (sink application).

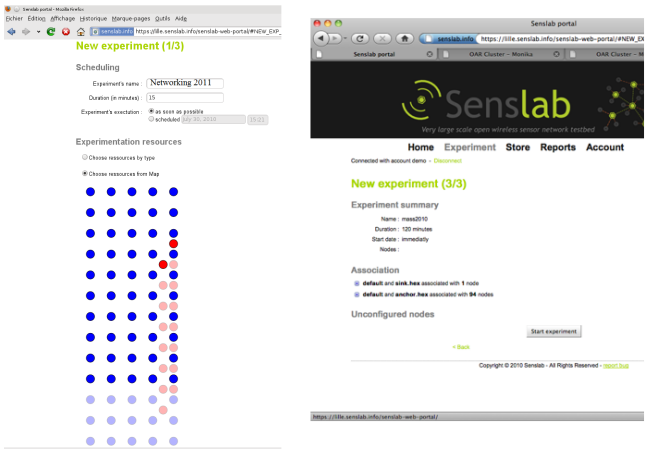
Experiment handler software: The experiment handler software (Fig. 3) is the server side interaction point with all the 256 nodes of a site. It can execute all methods described above such as firmware update, energy consumption monitoring, polling, etc. It also receives the data coming from the serial links of open nodes relayed by the gateway nodes. It instantiates a 'testbed manager' object

and an OSGi container when started. The OSGi container embeds several bundles, responsible for all the interactions with the nodes: the Node Handler bundle sends command frames to the gateway and the control node; the Firmware Deployer bundle provides one service allowing parallel deployment of a firmware on several open nodes; the Sensor Controller bundle allows parallel sensor measurement; the Sink Forwarder bundle provides efficient data redirection between nodes and users' VMs.

Batch scheduler software: Through a web form (or by uploading an xml file), the user configures his/her experiment and specifies requested nodes (either mobile or fixed nodes, with a CC1100 or CC2420 radio chip, outdoor or indoor, location and the number of nodes), experiment's duration and eventually a start date. Those information are transmitted to the batch scheduler software, which is the server-side module allowing optimal experiments scheduling and resources allocation. This module is based on the use of OAR[14], which is a versatile resource manager for large clusters.

User virtual machines: A complete Linux environment is made available to each registered user allowing him/her to build sensor firmwares thanks to the complete set of tools installed. from his/her VM, user interacts with the nodes of his/her running experiment. He/She can also run any dedicated application to handle the nodes' serial link outputs.

4 Conducting experimentations with SensLAB



(a) Step 1 (b) Step 3

Fig. 4. The SensLAB web portal for Lille's platform.

[14] http://oar.imag.fr

When a user wishes to conduct some experiments on the SensLAB platform, he/she first needs to configure his/her experiment through the webportal. As displayed by Fig. 4(a), SensLab offers an easy way to select nodes needed. Through this web portal page: the user is able to choose the experiment name (*Networking 2011* on Fig.), the duration of the experiment (15min) and if the experiment has to start as soon as possible or at a chosen date (*as soon as possible* in our case). Then, nodes on which the experiment will be run have to be chosen, either through their Id or on the map. These nodes are then associated with a specific firmware provided by the user. A summary page is then displayed (Fig. 4(b)). In our case, one node is configured as a sink and 94 as anchors.

Once configuration is done, the experiment is submitted. SensLAB automatically reserves nodes. At starting time, *i.e.* in our case, as soon as chosen nodes are all available, SensLAB configures *control nodes*, deploys firmwares, and resets the experiment's nodes. Every action is performed automatically and in a transparent way for the user.

Once the experimentation is launched, the user is able to interact with the running experiment as we will see in following sections illustrating an experiment. Results chosen to be retrieved (by polling or reading on serial port) are available in the user's virtual machine.

In the remaining of this section, we give some examples of utilization of the platform through several applications examples.

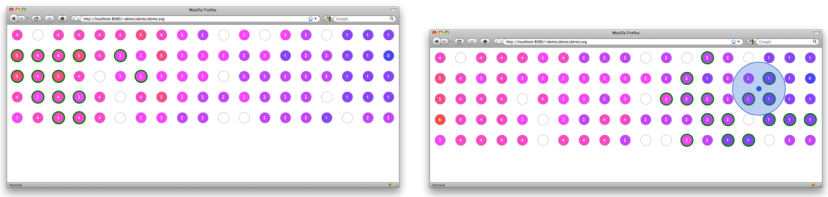
4.1 SensLAB for testing geolocalization protocols

To illustrate the benefits and the use of the SensLAB testbed in designing new algorithms, we focus on an animal tracking application [7]. To geolocalize them, some fixed nodes called *anchors* are spread in the park and receive signals from mobile nodes as soon as they are in range. *Anchor* nodes register the mobile node identifier, the RSSI (Received Signal Strength Indicator) of the signal and the date. Then, these data needs to be routed to a *sink* node. This latter is connected to a computer gathering data and computing *mobile* node location based on these data. Note that the geolocalization application has been simplified as possible since the main purpose here is to highlight SensLAB benefits.

In the setup, the first step is to cover the bounded area by deploying *anchors*. The next step is to set up the routing infrastructure to allow every anchor to send data to the sink. Once again, the routing process used here is simplified. When *anchors* are deployed and powered on, the *sink* is initialized. It then starts to send `BEACON` and every *anchor* receiving this `BEACON` attaches itself to the sink. The *sink* becomes its parent. Then every attached *anchor* u forwards the `BEACON`. Every unattached *anchor* receiving a beacon from u, chooses u as its parent. When every *anchor* has chosen a parent, the whole area is covered and *mobile* messages can be forwarded to the *sink* as follows. When an anchor receives a data message from another anchor or needs to send its own data, it forwards it to its parent. Step by step, the message eventually reaches the *sink*. The *sink* sends data through its serial link and the computer connected to it gathers the different messages and estimates mobile node positions.

Demonstration overview: Demonstration was conducted from the Lille's SensLAB site. Nodes of this site are featured with CC2420 radio chip and 32 among the 256 nodes are mobile, mounted on several toy electric trains. For the need of the application, we will reserve a grid of 5×10 nodes and 2 mobile nodes located on different train paths. Mobile nodes will represent the animal while fixed nodes will stand for anchor nodes. The reservation is be performed through the SensLAB web portal (see Section 3). Three firmwares have to be compiled for the experimentation, respectively for *mobile*, *anchor* and *sink* nodes. In the experiment only one *profile* will be used tuned with nodes on DC power. Consumption polling is sampled every 50 ms. All fixed nodes except one will be associated with *anchor* firmware and the last node is associated with the *sink* firmware.

Experimentation is launched. During the experiment, monitoring data (power consumption, RSSI, ...) are logged like described in the experiment profile which is the same for all nodes. Once the experimentation is launched, the user is able to interact with the running experiment via his/her VM as shown by Fig. 6.From the VM, and using the UNIX *netcat* tool to create and open a TCP socket on a specific port of the node handler host, which is the serial link redirection of the SensLAB node specially created for the ongoing experiment, the *sink* is ready to be activated and to start sending `BEACONS`. Results are gathered in real time. A route tree is created by the *anchors* to forward reports to the *sink* (In Fig. 5(a), numbers on nodes give the distance each node estimates itself to the sink at a given time.). *Mobile* nodes are activated through their serial links and start sending `Hello Messages` received by anchor nodes (Green nodes in Fig. 5(a)). Based on the RSSI of these Hello messages, anchors compute the location of mobile nodes (Fig. 5(b)).



(a) Gradient and Hello messages (b) Localization computation

Fig. 5. Real time interaction with the running experiment.

For the experimentation purpose, the VM is hosting an application which collects data from serial links, analyzes them to compute *mobile* node locations and provides a web server to visualize application status in real time (Fig. 6). *Anchors*, routes, messages and estimated *mobile* node positions can then be displayed in a web browser.

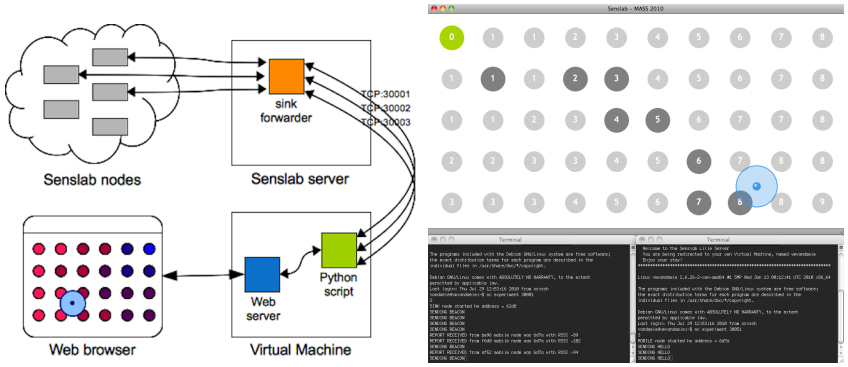


Fig. 6. Demo visualization interface showing routes, messages transmission and mobile nodes location in the upper window, and messages printing on serial links for sink and mobile nodes on lower windows.

4.2 SensLAB for testing medium access control protocols

Among the large set of communication protocols that have been studied especially for wireless sensor networks, an important amount of literature focusing on medium access control (MAC) mechanisms has been produced. The MAC layer dictates whenever a node may listen to the channel, transmit and receive so that some crucial properties regarding the usage of the shared radio medium (e.g. fairness, reliability) remain guaranteed. In WSN, the MAC layer should also be carefully designed in order to ensure a low energy consumption. The two widespread ideas consist in either allocating time slots to every device (that is allowed to sleep outside of these) or in having regular wake-ups during which any upcoming transmissions should be detectable (i.e. through the use of a preamble sent by the data source). These two main schemes are referred to as synchronized and preamble-sampling protocols, respectively.

Recently, versatile layers, such as B-MAC [9] and X-MAC [2], have gained much attention, especially due to their common tunable low-power listening mechanism that is expected to be suited to any traffic pattern. Still, an overview of today's WSN deployments shows no implementation of any of these protocols [5]. The fact that their performances have never been evaluated under realistic conditions is for sure one of the main reasons why. Indeed, the limited simulation models and the small scale testbeds that were used to evaluate the performances of these solutions have led us to use the SensLAB platform in order to conduct a thorough study of them. We here report some feedbacks concerning such an experiment.

Description of the experiment The empirical analysis of X-MAC was led over the SensLAB testbed portion installed in Strasbourg. Figure 7 shows the feedback interface that was used for the management and the demonstration of the results. The 3-dimensional grid is composed of 240 static sensor nodes, with

an homogeneous step of 1 meter between every pair of devices. Each one of them is equipped with Texas Instruments CC1101 radio interfaces[15].

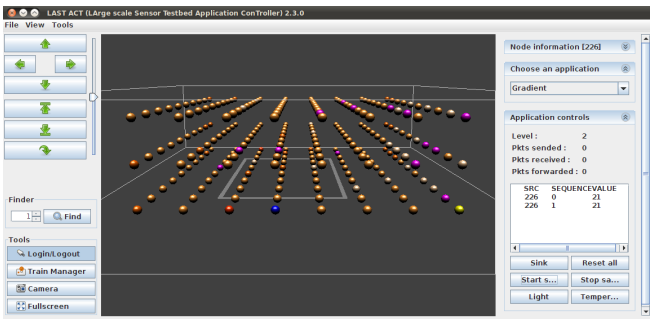


Fig. 7. 3D feedback application.

At the beginning of an experiment, up to 30 sensor nodes were selected as data sources. These nodes implement a time-driven application that transmits a 7 byte sample of data, every 10 seconds toward the sink (located a the center of the 3D grid). Multi-hop transmissions were achieved with a basic gradient-based routing protocol. Concerning the X-MAC deployment over the devices, its low-power listening feature was managed by the Wake On Radio (WOR) mechanism embedded in the CC1101 chipset. The sleep period duration was fixed to 101.5ms while the listening time was set to 15ms.

Experimentation results and analysis For each number of senders, a dozen of experiments were conducted. Each set of experimentations uses the same topology and the same senders to evaluate the MAC layers in a similar environment. The results presented here are an average of the overall collected data.

First, we studied the average overhead induced by MAC layer retransmissions that necessarily occur upon non-reception of an acknowledgment message (e.g. collisions, busy channel during the clear channel assessment). Figure 8 illustrates the average number of retransmissions induced by X-MAC. As expected, we can observe the stress of the medium as more and more data sources are added. While the number of application transmissions linearly increases, both the MAC and the physical layer require much more sent messages.

Energy consumption being one of the crucial points in WSNs, the SensLAB platform allows users to evaluate the instantaneous energy consumption of any node in the network during the experimentations. The INA209 chipset embedded in the nodes was used to retrieve the average value of 128 samples each 68ms. Results obtained during a 30 minute monitoring are exposed on Figure 8.

[15] http://focus.ti.com/docs/prod/folders/print/cc1101.html

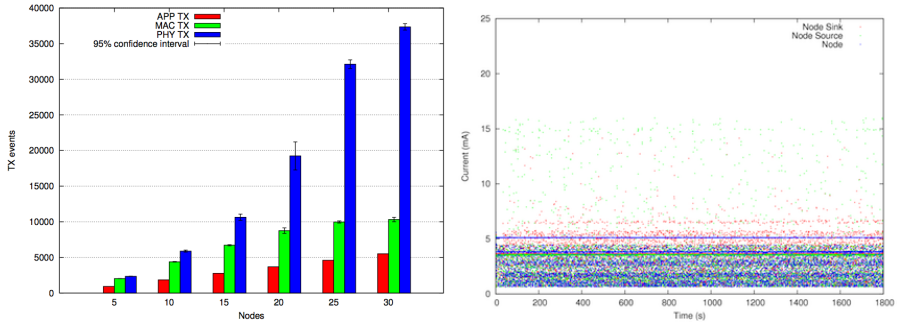


Fig. 8. (left) X-MAC retransmissions. (right) Instantaneous energy consumption among nodes of the SensLAB testbed.

5 Conclusion

The architecture concepts, the hardware design, the software implementation of SensLAB, a large scale distributed open access sensor network testbed were presented. We have also provided examples showing how to use SensLAB to launch and evaluate experiments, how to configure any nodes as a sink with a serial feed back channel connectable through TCP/IP to any application. The testbed is deployed and operational and researchers are invited to register. Obtaining an account is free and can be done from the main web page: `http://www.senslab.info`. We are working on the deployment of the OAR-grid version in order to allow fully flexible distributed node reservation. Several research works remain. One extension concerns the use of hybrid simulation within SensLAB by allowing to connect node and or network emulator to the testbed. Another extension is the development of actuator nodes, plugged directly on SensLAB nodes.

Another main direction concerns the study of the federation of research platforms, and more precisely with OneLab. A federation will offer a higher dimension in the spectrum of applications that the research community will design, test, deploy, and tune. But even more important, SensLAB will strongly benefit from the monitoring tools and supervising infrastructure developed and used in OneLab. Thanks to French government's stimulus funds, the FIT (*Future Internet of Things*) project is being awarded 5.8 millions Euros and will enable large scale and diverse experiments with Future Internet technologies, from components to complete systems, and to validate and compare them with existing or evolving solutions. The goal is to develop the tools and the management system required to run a global open federated testing facility.

The work program of the FIT project will consist in establishing the different sites with a set of competitive testbeds including wireless and wired technologies, with a strong emphasis on sensors, networked and embedded objects, multihop and cognitive radio, mobility and overlays. FIT is by nature a distributed facility, involving heterogeneous devices in a set of complementary components situated

in adequate and relevant locations. These testbeds are original not only in the cutting edge technologies that they offer, but also in their federated management system, which will provide experimenters a set of common methods to access and control numerous facilities.

References

1. A. Arora, E. Ertin, R. Ramnath, M. Nesterenko, and W. Leal. Kansei: A high-fidelity sensing testbed. *IEEE Internet Computing*, 10:35–47, 2006.
2. M. Buettner, G. V. Yee, E. Anderson, and R. Han. X-MAC: a short preamble MAC protocol for duty-cycled wireless sensor networks. In *SenSys '06: 4th International Conference on Embedded Networked Sensor Systems*, pages 307–320. ACM, October 2006.
3. G. Chelius, A. Fraboulet, and E. Fleury. Worldsens: development and prototyping tools for application specific wireless sensors networks. In ACM, editor, *International Conference on Information Processing in Sensor Networks (IPSN)*, 2007.
4. V. Handziski, A. Köpke, A. Willig, and A. Wolisz. Twist: A scalable and reconfigurable testbed for wireless indoor experiments with sensor networks. In *RealMAN 2006*, 2006.
5. R. Kuntz, A. Gallais, and T. Noël. Medium access control facing the reality of WSN deployments. *ACM SIGCOMM Computer Communication Review*, 39(3):22–27, July 2009.
6. S. Kurkowski, T. Camp, and M. Colagrosso. Manet simulation studies: the incredibles. *SIGMOBILE Mob. Comput. Commun. Rev.*, 9(4):50–61, 2005.
7. N. Mitton, T. Razafindralambo, and D. Simplot-Ryl. *Theoretical Aspects of Distributed Computing in Sensor Networks*, chapter Position-Based Routing in Wireless Ad Hoc and Sensor Networks. Springer, 2010. To appear.
8. K. Pawlikowski and J. L. R. Jeong. On credibility of simulation studies of telecommunication networks. *IEEE Communications Magazine*, 2001.
9. J. Polastre, J. Hill, and D. Culler. Versatile low power media access for wireless sensor networks. In *SenSys '04: 2nd International Conference on Embedded Networked Sensor Systems*, pages 95–107. ACM, November 2004.
10. G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: A wireless sensor network testbed. In *In Special Track on Platform Tools and Design Methods for Network Embedded Sensors (SPOTS)*, 2005.