Design and Implementation of the Vehicular Network Testbed Using Wireless Sensors

Jovan Radak(✉), Bertrand Ducourthial, Véronique Cherfaoui, and Stéphane Bonnet

UMR CNRS 7253 Heudiasyc, Université de Technologie de Compiègne, BP 20529, 60205 Compiègne Cedex, France
{jovan.radak,bertrand.ducourthial,veronique.cherfaoui, stephane.bonnet}@hds.utc.fr

Abstract. Testbeds are indispensable tools in research and development process in wireless networks technologies. They show us how our solution is going to work in a real environment. In the recent years there is a growing trend in the development of testbeds aimed to be used as tools for both research and verification of the results obtained theoretically and using simulators. We are presenting an experimental vehicular network testbed based on cheap, off-the-shelf wireless sensors that are gathering environmental data, temperature, humidity and luminosity. These sensors are connected to road-side units (RSUs) running the Linux operating system and dedicated software distribution, Airplug. This complete system (wireless sensors, RSUs and Airplug software distribution) can be used for simulation, emulation and experiments in vehicular networks but also for any other type of wireless network. We use this system to gather environmental data and then reuse collected data in different emulation and experimental scenarios. We are showing the usefulness of our wireless sensors testbed and possible scenarios of its usage in emulation and real experiments.

Keywords: Wireless sensors · Roadside units · Testbed · Emulation · Experiments · Airplug · Airbox

1 Introduction

Practical implementation is one of the final steps in research and development for the novel solutions in wireless ad-hoc networks (and in general any research and development process). This step is performed on a real hardware platform and it is usually the most critical one. The problem lies in the complexity of the hardware platforms that cannot be completely taken into account in the process of modeling and simulating the wireless ad-hoc networks. Thus, a dedicated hardware platform should be used for implementation and testing of the solution previously developed for some specific problem.

Different kinds of high-quality network simulators (like ns2, ns3, Omnet++ – just to name a few) are currently in use as primary tools for the development and testing of solutions for wireless ad-hoc networks. Network simulators have a large user base who develop different kinds of libraries suiting specific problems they are addressing. It is relatively easy to find a library for the specific problem that we want to tackle (for example tens of different energy efficient MAC layers for wireless sensor networks) and to run the simulation for our targeted instance (for example 50 fixed sensor nodes with 20 mobile nodes having moving patterns that correspond to the movement of the buyer in the supermarket). But, the main problem remains in the limitations of the models used in different simulators (their accuracy and completeness) and the lack of compatibility between imposed models, hardware platforms and real scenarios that are planned to be used in implementation.

Emulation using network simulators [12] or using a dedicated platform [13] takes the middle ground between simulation and experiments. It usually includes a mix of the real hardware or data retrieved from the real hardware and simulation. In this way emulation can be viewed either as an enhanced simulation or as some kind of experiment with a limited number of hardware elements. Indeed this is an approximation of the real experiment that guarantees the flexibility of the simulation with more realistic data but at the cost of limited hardware usage. This approach is advocated because of its practical usefulness and flexibility in rapid application development and testing. However, the main problem, previously mentioned for the network simulators, remains – limitations of the used models and gap between it and the dedicated platform planned for implementation.

We are presenting our testbed for vehicular ad-hoc networks. This testbed is developed to be used in conjunction with the previously developed tools for simulation and testing of dynamic wireless networks. We are using cheap, off-the-shelf, wireless sensors that are gathering environmental data – temperature, humidity and light. Our solution is based on the wireless Xbee sensors[1] (produced by Digi international), Airbox units, dedicated hardware based on the IGEP[2] developments boards and the Airplug software distribution, developed in our laboratory. Airplug is a modular and flexible software platform developed for the simulation, emulation and testing of dynamic wireless networks (and more generally distributed systems) that can be used on any Linux-compatible platform. Thus, our testbed is not limited to this specific embedded architecture (like Airbox), it can be used on any Linux-compatible embedded platform or desktop computer that fulfills a small subset of requirements. Airplug also allows us to reuse previously gathered data in such a way that we can either emulate an entire experiment on a single computer or recreate experiments using dedicated hardware.

The solution that we are proposing in this article is specific in four aspects:

– **Fully developed solution** – it consists of hardware and software elements that can be used as-is without any modifications. Currently we are using environmental wireless sensors that measure temperature, humidity and luminosity.

[1] http://www.digi.com/products/wireless-modems-peripherals/wireless-range-extenders-peripherals/xbee-sensors.

[2] https://www.isee.biz/products/igep-processor-boards.

- **Mobile platform** – our current testbed is fixed and developed close to our laboratory but using our guidelines, hardware and software distribution it can be developed on any site – indoors or outdoors, using different hardware platforms.
- **Modular platform** – our platform is modular in both physical implementation and program support. Additional hardware elements can be easily added and they do not depend on the manufacturer as long as they comply with general communication standards used in our platform. The Airplug platform is compatible to any Linux system, so any hardware platform (Linux compatible) can be used to extend current physical implementation. Also, additional program support can be developed using development guidelines for our software platform.
- **Flexible platform** – this platform is developed to be used in experiments on vehicular networks, but it is flexible and can be used in other types of experiments with devices like UAVs (unmanned air vehicles) or robots as long as the communication modules of these devices are compatible to those used in our platform.

In this article we are going to present our solution for vehicular network testbed emphasizing the usage of wireless sensors in it. We will also present program support for these wireless sensors and possible applications of the testbed equipped with environmental sensors. The rest of this article is organized as follows: Sect. 2 presents the work related to our solution, overview of testbeds used for wireless ad-hoc and sensor networks and the approaches in network emulation, Sect. 3 presents the general hardware architecture of our testbed, while Sect. 4 presents the Airplug software distribution and specific application support for the use of wireless Xbee sensors. Section 5 provides example of the data gathered using our testbed and possible scenarios of the use of our solution. Finally, Sect. 6 presents our conclusion of this article and ideas for the future work.

2 Related Work

In recent years we have seen great effort to develop experimental infrastructures for wireless ad-hoc networks and the Internet of things paradigm that will allow researchers to test their solutions in a real world environment. Some of these platforms are developed for specific applications while the others are more generic and can be used for a wider range of problems.

SmartSantander is a platform developed for research and experimentation on wireless sensor network and Internet of things in urban environment. It is a city-scale test site consisting of 20000 sensor nodes developed in 4 cities across the Europe: Belgrade, Guilford, Lübeck and Santander [15]. It is mainly viewed as a platform for experiments with the Internet of things services and infrastructures in the smart city. So far it is used for various experiments including the streaming of acoustic data [14]. GreenOrbs in China is the test site deployed using wireless sensor nodes and mainly aimed at forestry applications [1]. It has evolved into a dedicated environment monitoring system for real-time CO_2 management in the city [11].

Wisebed [10] proposes a different approach to the testbed infrastructure. It is a joint project of 9 different institutions in Europe with the testbed consisting of 750 sensor nodes (with different types of sensors, both static and mobile) organized in federation architecture. This platform also supports virtualization and co-simulation with part of the testbed. OneLab [8] is also federation of platforms but it aims to help both developers and users of the platform. It is open to third-party platforms allowing developers to promote their testbed but also users to chose among a variety of testbeds that best fits their testing platform.

Senslab is a large experimentation testbed deployed in 4 cities in France – Grenoble, Lille, Rennes and Strasbourg [7]. It consists of 1024 sensor nodes (WSN430 nodes developed specially for this project) with 2 types of radio chips and several environmental sensors. This platform's main purpose is experiments on wireless sensor networks both at high and low level of abstraction [4]. Additional tools are also developed: WSim (simulating sensor nodes) and WSNet (wireless sensor network simulator) simulators and ports to different kinds of real time operating systems (FreeRTOS, TinyOS, Contiki). Currently, there is an ongoing upgrade of the Senslab platform, called IoT Lab[3]. This platform promotes a new approach in the architecture of the testbed which is basically a service allowing users hands-on experience with real platforms. Wisebed, Smart-Santander and Senslab are parts of this platform allowing users to perform multidisciplinary research in the area of wireless networks and Internet of Things.

Emulation is widely used as a term that explains uses of both hardware and software in the process of evaluation and experimentation. The term emulation covers different approaches in testing and evaluation and it goes from pure simulation with the enhanced models [9] to the usage of smaller hardware platforms to replicate results of large scale networks [3,13].

In our laboratory we have previously developed the Airplug software distribution. This platform is the complete system that can be run several different modes including simulation, emulation and experiment mode. The user defines the parameters of Airplug usage on the start of application giving details about the scenario (type of mobility or fixed position, range, type of communication..., number of nodes as well as the mode of usage).

The solution that we are proposing in this article is enhanced comparing to the current testbed solutions in three aspects:

- Hardware and software co-development – both physical architecture and program support are developed simultaneously i.e. if we decide to use new type of wireless sensors that are measuring different set of values then we are also developing appropriate software support for that piece of hardware.
- Mobility – while the other testbed solutions also propose certain kind of mobility to the nodes (or subset of nodes) used in testbed our solution is unique in the sense that each part of the hardware platform can be easily relocated

[3] http://www.iotlab.eu/.

according to our needs in the experiment, also the same type of dedicated hardware (Airbox) can be used as the part of in-vehicle hardware platform making it fully mobile. Although we envisaged this testbed for vehicular network testing and development, it's concept can be used for different types of mobile agents – robots or unmanned air vehicles – and it can be easily transformed (using same software platform) in the testbed for other specific purposes (obstacle avoidance, object tracking, ...).

– Modularity – additional software elements can be easily added and they do not depend on the type of the elements used in physical implementation as long as they comply with general communication standards used in our platform. This means that if we need to test for example multi-hop routing algorithm we can develop application that implements that specific algorithm and to deploy additional number of Airbox units that will allow us to have appropriate test results.

3 Wireless Sensors Testbed

Architecture of our testbed is simple, it consists of wireless Xbee sensors and Airbox devices (dedicated embedded hardware). In the terminology of the vehicular networks Airboxes are road-side units – RSUs (we will use these two terms interchangeably throughout this paper). In the creation of the wireless sensor testbed our leading ideas were *simplicity* of usage, *modularity* and *mobility* of the given solution. Following our main concepts we have decided to use simple wireless sensors that do not communicate between themselves but only the device that is hierarchically above them in this case with the road-side units. To avoid possible data collisions we have decided to use different wireless standards for communication with sensors and for communication between RSUs. Figure 1 presents conceptual scheme of our testbed.

Wireless sensors are communicating only to the RSUs that are in their neighborhood using dedicated communication link (802.15.4 standard). All road-side units are running Airplug software distribution (more details about it in the Sect. 4) that has application dedicated to communication with Xbee sensors – given as XBE block in Fig. 1. Each RSU can serve one or more Xbee sensors. RSUs communicate between themselves using wifi communication link (802.11).

Configuration, that we are showing in Fig. 1 is fixed. This means that each Airbox used in this configuration is actually a road-side unit. Airboxes that we are using inside of the vehicle cannot communicate with wireless sensors (they are not equipped with Xbee modem), they are meant to gather data from vehicles using other interfaces, namely using CAN bus.

3.1 Wireless Sensors

We are using cheap off-the-shelf wireless Xbee sensors equipped with temperature, humidity and luminosity sensors. They are using Xbee modem based on the 802.15.4 protocol to communicate with other devices. These sensors are not

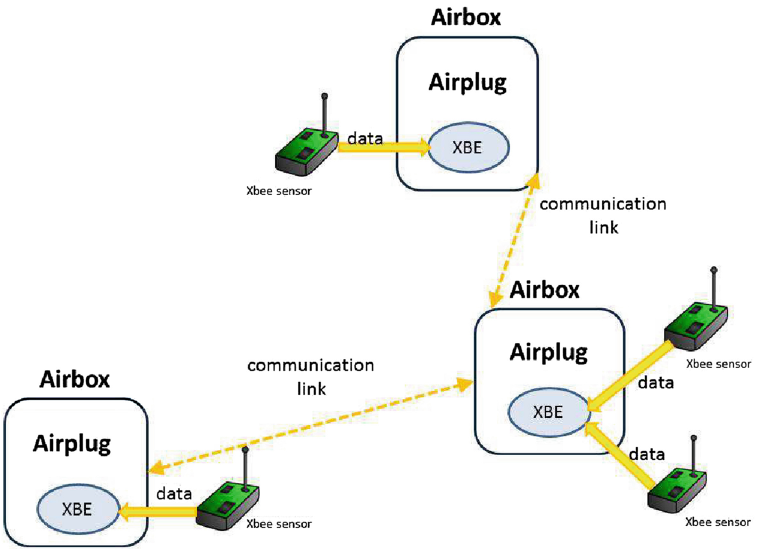


Fig. 1. Testbed with Xbee sensors, road-side units (Airboxes) and software support from Airplug software distribution

able to exchange messages between themselves, only to the device that is hierarchically above them (Fig. 2).

Usage of 802.15.4 protocol modems, so called series 1 Xbee modems, guarantee simple setup and communication between the sensors and other devices. Xbee modem is physically attached to the Airbox unit's dedicated slot. It communicates with the Airbox device using its (and device's) serial port. There are two possibilities for data gathering from Xbee sensors:

– Polling – Xbee modules are configured in such a way that they initiate periodical reading of the sensors and transfer of the retrieved data.

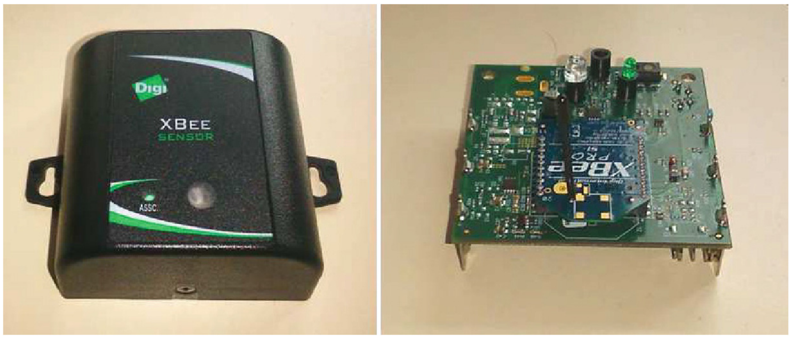


Fig. 2. Xbee sensor (with Xbee modem visible and enclosed)

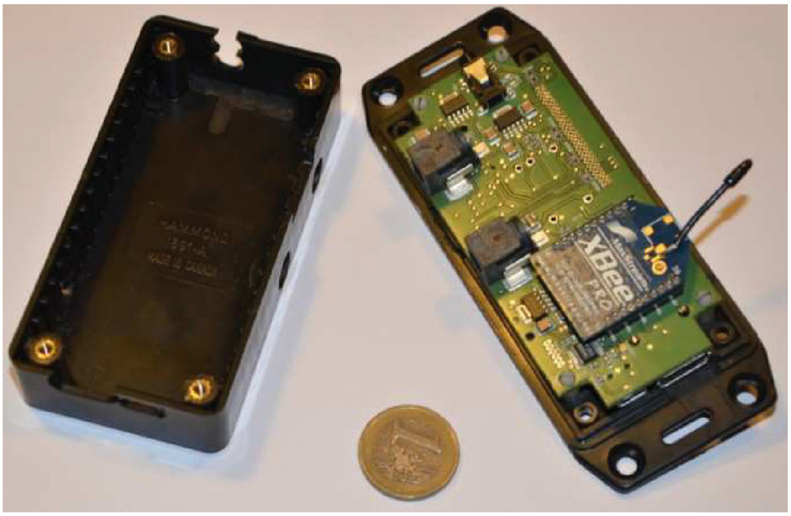


Fig. 3. Airbox – platform based on IGEP development boards, Xbee modem with external antenna is shown attached to the dedicated slot (1 euro coin is used for the comparison of sizes)

– Upon request – RSUs are initiating reading of the sensors sending the specific read request packet to Xbee sensors; for this to be available all Xbee modems must have API mode enabled.

3.2 Road-Side Units

Road-side units are dedicated embedded hardware units, called Airbox. They are based on the IGEP platform. These units are running Linux operating system and Airplug software distribution. RSUs are equipped with Xbee modems that allow them to communicate with wireless Xbee sensors. They are also having wifi communication module that allows them to communicate between themselves and to form ad-hoc network (Fig. 3).

RSUs and Xbee modems are configured to communicate using API mode of Xbee modems. This means that they exchange dedicated packets that follow the rules imposed by their manufacturer (Digi international).[4]

3.3 Placement of the Testbed

We have chosen to deploy our testbed near to the Research Center of the Technological University in Compiégne. Three road-side units are deployed. Configuration of the RSUs is shown on the Fig. 4. RSU1 is deployed close to the garage and has continuous power supply making it available all the time. RSU2 and RSU3 are

[4] http://www.digi.com/support/kbase/kbaseresultdetl?id=3215.

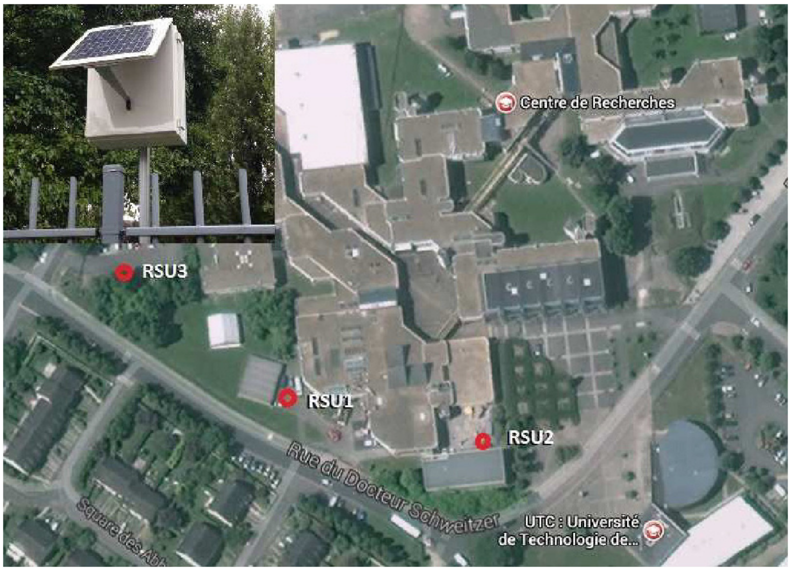

Centre de Recherches
RSU3
RSU1
RSU2
Rue du Docteur Schweitzer
Square des Abe
UTC : Université de Technologie de...

Fig. 4. Geographical position of the road-side units, RSU3 is shown in the upper left corner encased with an energy harvesting module

deployed in the parking space, closer to the street. In this way they can exchange messages with the vehicles equipped with Airbox units. These two RSUs are using batteries as a power supply, their autonomy is roughly around 20 h.

Xbee wireless sensors (not shown on the Fig. 4) are deployed in the proximity of each road-side unit. RSU2 and RSU3 have one Xbee sensor in their proximity while RSU1 has 2 Xbee sensor, one placed inside of the garage and the other one positioned outside of the garage. In this way it is possible to obtain different readings and to use this data in other applications.

4 Program Support for Wireless Sensors Testbed

Important part of our testbed is its program support, it is based on the Airplug software platform[5]. To incorporate wireless Xbee in our testbed and in the Airplug software distribution sensors we have implemented dedicated, Airplug compatible, application that implements all the functionalities of the wireless sensors. In this section we will give brief overview of Airplug software distribution and its functionalities and the support for the Xbee sensors, more details about design philosophy and Airplug's architecture can be found out in previous publications [5].

[5] http://www.hds.utc.fr/airplug/.

4.1 Airplug Software Distribution

Airplug software distribution actually presents practical implementation of the Airplug framework. This framework is developed to support dynamic wireless network and it is based on the few simple guidelines:

- Independence of the programming language implementation – to follow this guideline framework is using standard input/output channels (each programming language can read/write from input/output channels).
- Portable message format – we are using ASCII text messages, binary messages, if needed, are encoded; the only limitation is that the field delimiter (for the different parts of the message) cannot be used as the part of the field.
- Simple addressing scheme – Airplug compatible applications are uniquely addressed with a pair (application_name, host_name); communication between nodes rely mainly on the broadcast in the neighborhood thus three keywords are reserved for the host: **AIR** when we broadcast to all neighboring nodes, **LCH** when we broadcast to all local applications (run by the localhost) and **ALL** includes both local and remote broadcast.

Airplug software distribution presents several implementations that are also called *modes*. These *modes* are standalone implementations and they are complementing each other, i.e. while the one is dedicated for the laboratory studies, the other is used in real experiments.

Terminal Mode is also called airplug-term, it is based on the implementation of the Airplug framework in UNIX compatible terminal. This mode is dedicated for the rapid application development and prototyping and it gives many functionalities thanks to the wide range of libraries. These libraries are dedicated to Tcl/Tk programming language but they can be also programmed in any other programming language as long as they comply with message format and addressing scheme.

Emulation Mode is also called airplug-emu, it is a *network emulator* with the upper layers (of the communication stack) same as the ones in the real experiments while lower layers (physical communication) are reproduced – simulated [2]. Emulation scenarios are described using XML files with the possibility to detail each node's mobility and the applications that are running on each of the node. This mode can be extended using *remote mode* (airplug-rmt) that allows some applications to use remote execution connecting them via sockets to a dedicated application called RMT (an application that relays messages between computers).

Live Mode also called airplug-live, is the implementation dedicated for the real experiments. It is efficient implementation written in C programming language and it manages both local and inter-node communication while running on the top of POSIX compatible operating system [6]. This implementation actually represents middleware between Airplug compatible applications and network interfaces.

4.2 Xbee Sensors Connectivity

Program support for the wireless Xbee sensors is Airplug compatible application – called XBE – written, like the most Aiplug libraries, in Tcl/Tk. This application (XBE) is compatible with all Airplug software distribution modes, meaning that it can be run both on the PC running Linux terminal, but also on the embedded devices used in experiments, running airplug-live.

XBE establishes connection between Xbee modem and serial port of the device running the XBE application, using either predefined configuration settings for the serial port or the settings that user gives upon the start of XBE application. It communicates with Xbee sensors sending the request for read packages either periodically or upon the users request (on the graphic user interface – GUI). Communication with other applications is periodic and the messages containing sensors readings, unique ID and time-stamp of the reading of each sensor are sent with the period that user defines on the start of application. Sending of message is also possible on the request but only in the terminal mode when the GUI is present. Read data can be logged into the file using Airplug saving facilities.

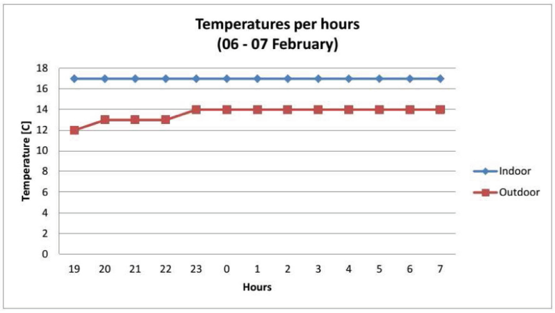
This application has ability to read the log files previously created with this application. We have implemented this with an intention to replicate the experiments in emulation mode using the real data previously gathered from Xbee sensors.

5 Preliminary Results and Experimentation Scenarios

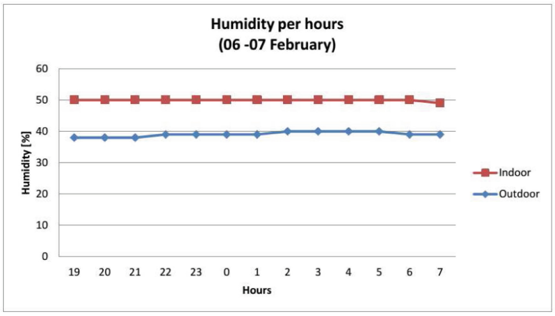
In this section we will give brief overview of the usage of our testbed and explain possible scenarios of usage for the data collected using it. Testbed is used for environmental data gathering that should be used in the emulation of new protocols that we are developing. Four different scenarios of testbed and data usage are given along with an explanation on the specificity of each scenario and its possible application.

5.1 Environmental Data Gathering

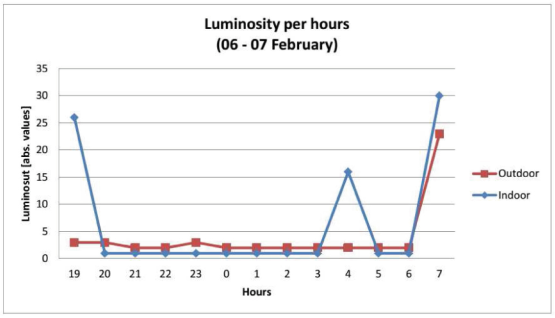
As a part of our work on the distributed data fusion we were evaluating algorithms using generated environmental data. Our idea was to develop and use our own testbed equipped with sensors gathering environmental data. In the Fig. 5 we are showing the example of the data gathered by our testbed. More precisely, two wireless Xbee sensor were attached to the one road-side unit (RSU1 in the Fig. 4) one being inside of the building and the other one outside. On these graphs (Fig. 5) we are showing mean values of the data gathered (RSU has requested data from Xbee sensors each minute) in the night between 6 and 7 February.



(a) Temperature



(b) Humidity



(c) Luminosity

Fig. 5. Environmental data gathered during 12 h period (from 19 h - 07 h) in the night between 6th and 7th February using Airplug software RSU unit and Xbee sensor

5.2 Different Scenarios for Testbed and Gathered Data Usage

Gathered data along with the possibilities of our testbed and different Airplug modes give us four possibilities for the experiments with environmental data. We must stress in here that only two of four possibilities are real experiments (in the sense that they are using all deployed hardware) but the idea is that they all use real data, either previously gathered or collected on the fly from the sensors running at the time of experiment.

Real Time Experiment with Collected Data. The idea is to use our testbed hardware with data gathered in the real time. Each RSU is running Airplug-live mode with XBE application while being connected to Xbee sensor(s). Data is gathered in the real time, depending on the parameters set for each XBE application.

This kind of experiment is good for the verification of communication algorithms that depend on the environmental data but in the case when algorithm itself does not depend on the varying of environmental parameters. This is due to the change of the environmental data that is rather slow and cannot be easily observed in an experiment that lasts for a short period of time.

Experiment with Emulated Environmental Data. Real hardware is used with RSUs running Airplug-live mode and XBE application. The only difference to the previous solution is that XBE application is not gathering data in real time, it is reading the log files of the previously collected data with XBE application.

In this kind of experiment we can choose the speed of reading from the file. Knowing that we have delays between the readings in the log file we can produce the same kind of an experiment like the one using real data. In this mode we can also choose to speed up readings of the data from log file thus effectively speeding up the experiment for a parameter defined at the start of experiment.

We can use this kind of experiment when we want to track how the algorithms respond to the rapid changes of the environmental parameters. All other parameters (V2I communication, GPS readings, delays between transmissions, etc.) remain the same but the change of environmental parameters is rapid so we can better observe its influence on the algorithm. Moreover, using this scenario we can easily setup different environmental data than the ones that we are able to retrieve in real time. For example, we can run the experiment during the warm days using data gathered during winter, thus effectively we will have the experiment run in the winter (according to the environmental data).

Emulation Using Real Data. Idea behind this scenario is to use the data gathered in the real time by one or several RSUs, each of them having Xbee sensors and XBE applications running. The data is then used either in one computer running the appropriate emulation scenario with airplug-emu mode or it can be run on multiple computers using airplug-rmt mode.

For example, we can setup emulation scenario using several vehicles moving using map traces and fix several RSUs in the scenario. Some of these RSUs can be the ones that are gathering data in real time and some of them running XBE application with data gathered previously, or we can avoid using XBE applications at all on some of the RSUs (if the testing does not have use of them) and run other Airplug compatible applications that are implementing tested algorithms.

In this kind of scenario we are mixing different kind of applications and sources of data while communication parameters are emulated in the Airplug-emu mode. These kinds of scenarios are good for longer experiments in which we have repetitive change of some parameters (movements of vehicles, communication between RSUs or V2I communication) but we want to see how different environmental data may influence execution in this specific case.

Emulation Using Gathered Data. This scenario can also be called pure emulation. In this kind of experiment we do not depend on the hardware that we are using, everything can be executed using only one PC, running the airplug-emu and appropriate emulation scenario or using multiple PCs with aiplug-rmt.

This is multipurpose experiment, since we can easily change source of the data, at which rate it is being read and communication parameters. It is best suited for the rapid development of the applications and as the first step after the simulations ran on some of generic simulators (ns2, omnet++, etc.).

6 Conclusion

In this paper we have explained our solution for the usage of wireless sensors and dedicated hardware to build testbed for vehicular ad-hoc networks. While it is the truth that our current solution has only three fix nodes it can be viewed as the proof of concept of our idea of wireless sensors testbed specially suited for the urban areas and vehicular networks. Flexibility of our hardware and software platform permits us to easily deploy this kind of testbed that can contain significantly higher number of nodes (measured in tens or hundreds). This architecture can also be used with different wireless sensors, with the only limitation that they have to use same type of communication module as our dedicated Airbox units. Development and testing of different kinds of algorithms, if not already a part of the Airplug platform, is based on the development of the dedicated applications that handle data provided from other Airplug applications.

We see great potential in the future usage of our testbed architecture. We are already using it for the gathering of environmental data that we plan to use for thorough testing of previously developed algorithms. We are also planning to use this platform to solve different problems in vehicular networks (distributed data fusion using information gathered from sensors, data propagation, correlation of gathered spatial data, etc.). While primarily developed as a tool for vehicular network testing this same platform can be used for the experiments with UAVs (unmanned air vehicles) and for robot networks.

Acknowledgments. This work was carried out and funded in the framework of the Labex MS2T. It was supported by the French Government, through the program "Investments for the future?" managed by the National Agency for Research (Reference ANR-11-IDEX-0004-02).

References

1. Bo, C., Ren, D., Tang, S., Li, X.-Y., Mao, X.F., Huang, Q., Mo, L., Jiang, Z., Sun, Y., Liu, Y.: Locating sensors in the forest: a case study in greenorbs. In: INFOCOM, 2012 Proceedings IEEE, pp. 1026–1034 (2012)
2. Buisset, A., Ducourthial, B., El Ali, F., Khalfallah, S.: Vehicular networks emulation. In: ICCCN, pp. 1–7 (2010)
3. Coulson, G., Porter, B., Chatzigiannakis, I., Koninis, C., Fischer, S., Pfisterer, D., Bimschas, D., Braun, T., Hurni, P., Anwander, M., Wagenknecht, G., Fekete, S.P., Kröller, A., Baumgartner, T.: Flexible experimentation in wireless sensor networks. Commun. ACM **55**(1), 82–90 (2012)
4. des Roziers, C.B., Chelius, G., Ducrocq, T., Fleury, E., Fraboulet, A., Gallais, A., Mitton, N., Noel, T., Valentin, E., Vandaele, J.: Two demos using SensLAB: very large scale open WSN testbed. In: Distributed Computing in Sensor Systems and Workshops (DCOSS), 2011 International Conference on, pp. 1–2 (2011)
5. Ducourthial, B.: Designing applications in dynamic networks: the Airplug software distribution. In: ASCoMS@SAFECOMP (2013)
6. Ducourthial, B., Khalfallah, S.: A platform for road experiments. In: VTC Spring (2009)
7. Ducrocq, T., Vandaele, J., Mitton, N., Simplot-Ryl, D.: Large scale geolocalization and routing experimentation with the SensLAB testbed. In: Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on, pp. 751–753 (2010)
8. Fdida, S., Friedman, T., MacKeith, S.: OneLab: developing future Internet testbeds. In: Di Nitto, E., Yahyapour, R. (eds.) ServiceWave 2010. LNCS, vol. 6481, pp. 199–200. Springer, Heidelberg (2010)
9. Girod, L., Stathopoulos, T., Ramanathan, N., Elson, J., Estrin, D., Osterweil, E., Schoellhammer, T.: A system for simulation, emulation, and deployment of heterogeneous sensor networks. In: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems, SenSys 2004, pp. 201–213. ACM, New York (2004)
10. Hellbruck, H., Pagel, M., Kroller, A., Bimschas, D., Pfisterer, D., Fischer, S.: Using and operating wireless sensor network testbeds with wisebed. In: Ad Hoc Networking Workshop (Med-Hoc-Net), 2011 The 10th IFIP Annual Mediterranean, pp. 171–178 (2011)
11. Liu, Y., Mao, X., He, Y., Liu, K., Gong, W., Wang, J.: Citysee: not only a wireless sensor network. IEEE Netw. **27**(5), 42–47 (2013)
12. Mahrenholz, D., Ivanov, S.: Real-time network emulation with NS-2. In: Distributed Simulation and Real-Time Applications, 2004, DS-RT 2004, Eighth IEEE International Symposium on, pp. 29–36, Oct 2004
13. Pavkovic, B., Radak, J., Mitton, N., Rousseau, F., Stojmenovic, I.: From real neighbors to imaginary destination: emulation of large scale wireless sensor networks. In: ADHOC-NOW, pp. 459–471 (2012)
14. Pham, C., Cousin, P.: Streaming the sound of smart cities: experimentations on the SmartSantander test-bed. In: Green Computing and Communications (GreenCom), 2013 IEEE and Internet of Things (iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing, pp. 611–618 (2013)
15. Sanchez, L., Galache, J.A., Gutierrez, V., Hernandez, J.M., Bernat, J., Gluhak, A., Garcia, T.: SmartSantander: the meeting point between future internet research and experimentation and the smart cities. In: Future Network Mobile Summit (FutureNetw), 2011, pp. 1–8 (2011)