OpenWiNo: An Open Hardware and Software Framework for Fast-Prototyping in the IoT

Adrien van den Bossche, Réjane Dalcé, Thierry Val

OpenWiNo: An Open Hardware and Software Framework for Fast-Prototyping in the IoT

Adrien van den Bossche, Réjane Dalcé, Thierry Val
Institut de Recherche en Informatique de Toulouse
IRIT, Université Fédérale de Toulouse, INP, UPS, UT1, UT2J
Toulouse, France
{vandenbo, dalce, val}@irit.fr

***Abstract*— The Internet of Things promises an always-connected future where the objects surrounding us will communicate in order to make our lives easier, more secure, etc. This evolution is a research opportunity as new solutions must be found to problems ranging from network interconnection to data mining. In the networking community, innovative solutions are being developed for the Device Layer of the Internet of Things, which includes the IoT wireless protocols. In order to study their performance, researchers turn more often to real world platforms, commonly designated by the term "testbeds", on which they may implement and test the protocols and algorithms. This is even more important in the Industrial IoT field, where environments are perturbed by industrial systems like automated production systems. In this paper, after a brief presentation of the context of testbeds, we introduce WiNo and OpenWiNo, an open hardware and software framework for fast-prototyping in the field of the Internet of Things. Compared to existing platforms, the solution WiNo+OpenWiNo offers a wide array of Physical layers and easy integration of various sensors as it is developed as part of the Arduino ecosystem. It also allows research teams to easily and quickly deploy their own testbed into real environments.**

Keywords— Internet of Things; Wireless Sensor Networks; Fast prototyping; Testbed; Open Hardware; Arduino

I. INTRODUCTION

The Internet of Things (IoT) is currently revolutionising the field of networks and telecommunications. Many specialists expect an exponential growth, in the years to come, of the number of connected devices [1] in industrialised countries, especially in the comfort, leisure activities, and quality of life and health areas. Among these connected devices are fixed elements for environment monitoring, but also mobile elements, for instance carried by a human being, or attached to livestock or to smart vehicles. Today, these different classes of devices form unconnected networks, since they use communication protocols and technologies which often are not interoperable. The goal of the IoT is to get them to collaborate by giving them all the ability to communicate via the Internet. This revolution is at the crossroads between several areas of expertise, and opens up major opportunities for scientific contributions in the computing, electronics and telecommunication domains, and far beyond if we consider how they are applied by users.

In the IoT context, the literature defines the Industrial IoT (IIoT) [2] as a subset of the IoT, focusing on communication between objects and tools without human interaction. The IIoT is generally associated with the concept of Machine-to-Machine communications (M2M) [3]. The hardware-specific aspects and the low-level protocols, which constitute the IoT Device Layer (IoT-DL) [4] – the first (or last) links in its overall structure – also find a new lease of life in this cloud of things and areas of research for the specific features of Wireless Sensor Networks (WSN).

The network and protocol development tools and methods must be suited to the IoT. At present, we observe the emergence of a great amount of work [5] based on simulation tools, such as NS3 [6] or Cooja for Contiki, designed to test and analyse the new low-level protocols (point-to-point or mesh network, time and/or energy constrained devices, etc.) used by the nodes in the IoT-DL. However, the research community is also getting more involved in analysis and performance investigation through hardware test platforms known as testbeds. This trend often comes as a complement to traditional simulation-based studies. The low cost and high availability of efficient and fully reprogrammable components favour the development of these assessment techniques. Several new platforms are coming into being, which sometimes originate from hardware and software environments used a few years ago for WSN. In order to support our studies and investigations on the IoT-DL, we developed an open framework named OpenWiNo, which mainly focuses on simple replacement of the physical layer (PHY) and simple/low cost deployments of testbeds. This paper's aim is to introduce this platform, its strengths and some first results obtained with this environment.

The following sections of this article will introduce the context and the related works. The OpenWiNo framework will then be presented, before conclude and provide perspectives for this research area.

II. CONTEXT AND OBJECTIVES

At the start of the 2010 decade, many innovative transmission modes made their debut in the fields of Wireless Personal Area Networks (WPANs) and Wireless Local Area Networks (WLANs). Ground-breaking solutions like light-based LiFi, Ultra-Wide Band (UWB) or LoRa as well as

additions to standards (802.11ac, 802.15.1 BLE, 802.15.6) are all striving to become the reference in the field. These physical layers (PHY) and the associated Medium Access Control (MAC) layer, whether proprietary or from open standards, are now commonly found and implemented in devices of the IoT. Although a similar competition has been observed in the 1990's, the outcome will not be a unique standard, meant to rule all short-range communications. When taking into account the development of Cognitive Radio or Software-Defined Radio (SDR) and Software-Defined Network (SDN), one can perceive a future based on inter-network cooperation: on one hand, the diversity of transmission modes now seems to be accepted both by the designers and the users; on the other hand, the high availability of the Internet Protocol, including on hardware targets with very limited resources, guarantees a convergence above heterogeneous physical layers. This new vision can be summarised through the following: "as long as a connection is provided, the communication technology doesn't matter". The coming years will show whether the market as a whole accepts this diversity for good.

Regarding the hardware, the IoT community is very prolific, thanks to the numerous FabLabs and Makers: a great many hardware platforms based on the Open Hardware concept are now available. These platforms have two well-known advantages: on the one hand, they make full use of the open nature, both in terms of hardware and software. While the Open Source has demonstrated its efficiency in software, the hardware now adopts its codes (design of the boards under BSD licence, Creative Commons, even GPL, access to platforms for fast prototyping and fast production of printed circuit boards at a very low cost, etc.). The Arduino ecosystem is an example of widely used Open Hardware. The community can then take these systems over and quickly and efficiently make headway with innovation in the area of the IoT. On the other hand, beyond the networking aspects, the high accessibility of these platforms means that their use extends far beyond the networking community and brings pluridisciplinarity in the research projects: for example, it is now easier to study the performance of a real-deployed sensor network, by taking into account the human experience, with the help of the Human-Machine Interface (HMI) and psychology teams, thanks to real nodes, implementing real sensors. This pluridisciplinarity allows researchers to come up with innovative products and services, extending far beyond the sphere of networking.

In this context, studying the performance of a network protocol can be done by using a testbed [7], as a complement of classical network simulators. As there is, on the one hand, an explosion in the diversity of forms of transmission, and on the other hand, a very simple access to hardware, the prototyping of innovative communicating devices is facilitated. Deploying a set of prototyped connected objects and studying the performance of the system with both classical networking performance metrics and human feeling gives a very interesting approach for the researchers.

III. Related Work and Motivations

The typical IoT research topics can be approached from either a data angle or an infrastructure angle. In the first case, the problem is how to transform a massive amount of data into relevant information, giving rise at the same time to innovative applications. This approach puts the middleware in the spotlight, as it hides the physical differences between networks. The Vital IoT [8] is a case in point.

Approaching the IoT with the infrastructure in mind involves developing protocols allowing for secure remote interaction with equipments. This generates the need to set up structures for the study of these solutions. Due to our own specialisation in wireless networks, we will focus on the communication aspects of the issues related to the IoT. Thus, we will be interested in the means available for implementing and testing communication solutions on testbeds.

The term testbed is commonly used to refer to platforms often developed for research purpose, which include a great number of nodes (large scale), and are open and accessible on request. These platforms enable users to test their protocols, from functional validation to performance analysis, on a limited number of nodes or on a larger scale. The most prominent testbeds nowadays are FIT/IoT-Lab [9] and the SmartSantander project. The SmartSantander platform for smart cities vies with FIT/IoT-Lab with respect to the scale of the deployment: each of these testbeds makes several thousands of nodes available. Both testbeds support various communication media (IEEE 802.15.4-2006 standard, at 868MHz and 2.4GHz for FIT and a considerably more heterogeneous set for SmartSantander as it includes NFC tags, smartphones, nodes embedded on public transportation buses…) and a wide array of sensors (magnetometers, accelerometers, gyroscopes…). FIT/IoT-Lab supports robot-based mobility while SmartSantander includes mobile nodes such as smartphones as well as mobility constrained units (buses). From a software point of view, both platforms offer a framework allowing reprogramming of the nodes, results visualisation etc.

Although the traits we enumerated make these testbeds very interesting from a research point of view, they may be impractical for certain studies. When considering testbeds for the IIoT, it is important to remember that the deployment environment may impact the performance evaluation parameters such as throughput, latency, loss of messages: Warehouses, factories and assembly lines are environments which are highly perturbed by industrial systems such as automated production systems, motors, etc. Driving a performance study on a classical testbed may not be representative; deploying a testbed directly on the final environment, as soon as possible in the development process, may be interesting.

Last but not least, as seen in the previous section, there are today many transmission technologies (PHY layers) providing new functionalities, ranging from evolution of standardised technologies to ground breaking transmission modes. All these technologies are commonly available thanks to low-cost transceivers. In the IIoT context, studying the most suitable transmission PHY, to ensure the most reliable networking experience, is an important deal. Nowadays, a wide variety of nodes is available, ranging from memory-constrained devices to others which are capable of running an operating system

(FreeRTOS, LiteOS or even Linux). Unfortunately, these standard devices hardly ever match the required set of functionalities and are not meant to be customized from a hardware point of view. There is an opportunity for Open Hardware solutions, to enable fast replacement of transceivers on testbed motes.

If we sum this up, on one hand, we have powerful platforms, ready to be remotely used, and on the other hand, we have devices that can be installed rather easily in the desired space but are difficult to modify. The aim of our work is to fill the gap by providing a set of software and hardware tools which enable the creation of a cost-effective, easy to manage, customizable testbed. Although we have deployed one such solution in the context of IIoT, the objective is not the size of the network but the suitability of our solution to a wide array of issues, especially when a local testbed must be deployed in a particular environment.

As will be described in the following section, our open architecture allows a variety of sensors to be seamlessly integrated in the testbed. In addition, we also support many physical layers and offer the related Hardware Adaptation Layer (HAL): this will enable researchers to evaluate the impact of a change in the PHY layer on their protocols with minimal development.

IV. OpenWiNo: an open hardware and software framework for Fast-Prototyping in the IoT

WiNo (Wireless Node) is an Open Hardware platform for fast prototyping and pragmatic assessment of the performance of wireless protocols at the MAC (L2) and Network (NWK, L3) levels. Used in the context of the IoT-DL or WSNs, it enables fast-prototyping and easy-deployments of nodes in a real environment. In combination with the OpenWiNo software, the WiNos offer a low-level access for a demanding developer who wishes to precisely control the medium access delays, the standby and wake-up modes of the nodes as well as the CPU load with a small memory footprint. Whether the objective is to apply drastic power-saving policies or to maintain a high Quality of Service, such a control of the node's components is necessary; WiNo is a hardware platform suitable for protocols with stringent time constraints and an uptime objective of several months of operation using two AAA batteries [10][11]. Table 1 gives the precise consumption state of a typical WiNo.

The architecture of WiNo (hardware) and OpenWiNo (software) is represented on Fig. 1. OpenWiNo comes with a kernel which proposes the classical tools to compensate the absence of an Operating System: a software interrupt engine, FiFos' management, etc. OpenWiNo's kernel also provides specific networking tools such as pushing/poping bytes into messages to help to the protocol implementation, and a serial console to interact with the node with a set of commands via its USB port [12].

OpenWiNo aims three main objectives: Simple replacement of the physical layer, very simple deployment of testbeds and test in real-life conditions, including usage feedback.

TABLE I. WiNo Power Consumption

Hardware element	State	Power consumption (Vcc=3.7V)
CPU Freescale MK20DX256 VLH7	Working, 96MHz	129mW
	Working, 72MHz	103mW
	Working, 48MHz	88.8mW
	Working, 24MHz	55.5mW
	Working, 16MHz	32.9mW
	Working, 8MHz	22,2mW
	Working, 4MHz	14.8mW
	Working, 2MHz	5.18mW
	Sleep, any freq, LPTMR wake	2mW
	Deepsleep, any freq, LPTMR wake	650µW
	Hibernate, any freq, LPTMR wake	<30µW
Transceiver HopeRF RFM22b	Transmit (10dBm)	76mW
	Receive	57mW
	Idle	26mW
	Sleep	<5µW

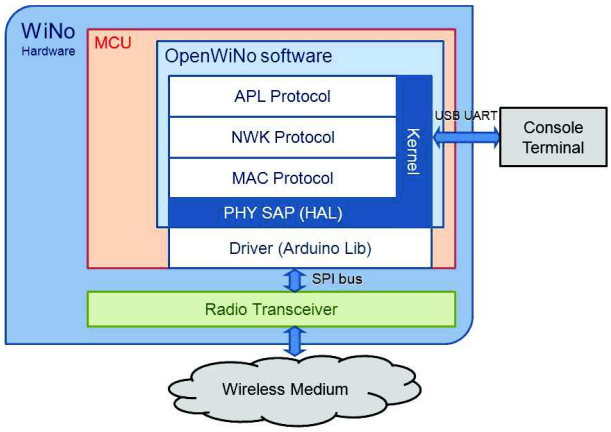

Fig. 1. OpenWiNo architecture

1) Simple replacement of the Physical Layer. Nowadays, there are many ways to transmit information (conventional IEEE communication technologies, unusual transmission modes such as LiFi or LoRa…) and it may be very useful to compare these various ways. WiNo supports both standardised PHY layers and ground-breaking transmission modes like the UWB or LoRa. In fact, thanks to the Open Hardware nature of the WiNo, any transceiver chip supported by the Arduino ecosystem can be easy integrated in OpenWiNo, by using the PHY Service Access Point (SAP), designed as a Hardware Abstraction Layer (HAL) (Fig. 1). The transceiver of a WiNo can be simply changed from a hardware point of view by modifying the electronic board and from a software point of view by changing the library (driver). While in a traditional

industrial approach, this operation would be time-consuming in terms of development and integration, the Open Hardware and the Arduino environment allow this operation to be carried out in a relatively simple manner: A first WiNo prototype can be assembled with a couple of breakout boards, including microcontroller and radio transceiver. Then, by using the services of a local FabLab, the new WiNo prototype can be produced with specific form factor, box, etc. and deployed in the test environment. The ability to change only one system component is important to enable precise performance comparison in real-life use. Table 2 and Fig. 2 illustrate some WiNo examples with various radios: WiNoRF22 (a) and TeensyWiNo (b), both based on the HopeRF RFM22b proprietary radio; DecaWiNo (c) [13], based on the DecaWave DM1000 IEEE 802.15.4-2011-compliant UWB radio [14].

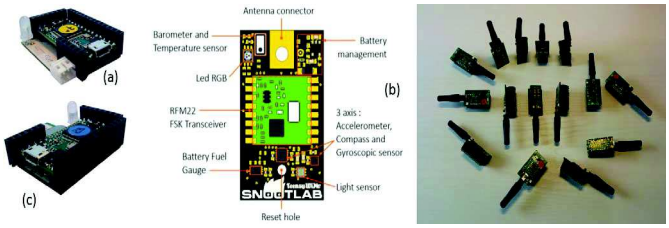

Fig. 2. Various WiNos: WiNoRF22 (a), TeensyWiNo (b), DecaWiNo (c)

TABLE II. WiNo CHARACTERISTICS

	TeensyWiNo	**WiNoRF22**	**WiNoLoRa**	**DecaWiNo**
Usage	Classical IoT		Long range, ultra low rate	Short range, ranging, localisation
CPU RAM	PJRC Teensy 3.2 Arduino-compliant (Freescale MK20DX256VLH7) ARM Cortex M4 72MHz, 64kB RAM, 256kB Flash			
Transceiver	RFM22b		RFM95	DW1000
Library	Radiohead			DecaDuino
Sensors	Temperature, Luminosity, Barometer, Accelerometer, Magnetometer, Gyroscope	Temperature, Luminosity		
Others	RGB LED, GPIOs, PWM, ADC/DAC, SPI, I2C, CANbus			
Availability	snootlab.com	DIY		

2) Very simple deployment of testbeds: OpenWiNo allows easy deployment of testbeds, either in controlled or natural environment. Once deployed, the WiNos execute the protocol stack (Fig. 1) on their wireless interface, while being managed by a second network, called supervision network (Fig. 3) via their USB interface, with the kernel console. This additional infrastructure is used for debugging, firmware injection and performance parameters gathering. Thanks to the supervision network, the WiNos can then be used for pragmatic assessment of the performance of the developed protocols. The supervision network comprises the WiNos, connected via USB to a controller, which can handle several WiNos. The controllers are connected to a central server via wired Ethernet/IP links, which centralises console logs and forward console commands to the WiNos, via the controllers. In our lab, the controllers are built using ordinary Raspberry Pi. If the experiment does not permit the deployment of the supervision network, the WiNo consoles remain accessible by using the built-in OpenWiNo remote-shell via a WiNo used as a gateway; In this case, the user must deal with the limited resources of the wireless network to optimise the quantity of data used in the assessment.

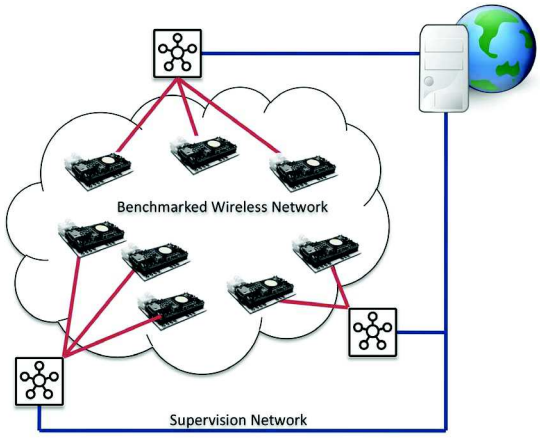

Fig. 3. Testbed architecture, with both wireless and supervision networks

3) Real-life usage: The WiNos small size and very low energy consumption facilitates their integration in prototypes of communicating objects, making them a component of the IoT. They can also easily be carried by a person or be attached to automatic motion systems. In addition, a great variety of sensors can be added to the platform: being compliant with the world of open hardware and software, the WiNo architecture permits addition of foreign libraries related to the desired sensor. Since the nodes are easy to carry and can be personalised using a wide array of sensors, designing a realistic implementation is possible. This setup may be used for a joint study of network performance and usage-based testing, but also for demonstrations and as a proof-of-concept.

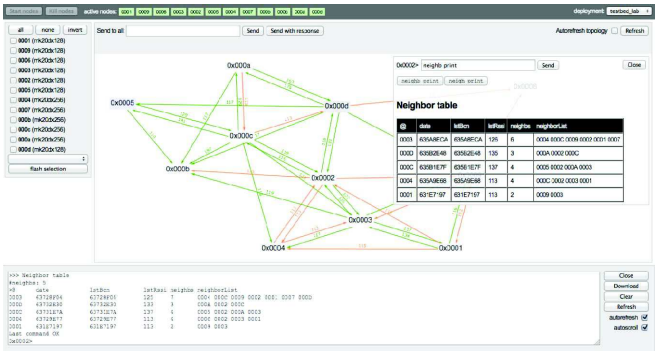

Fig. 4. Testbed management via the web interface

V. OpenWiNo Examples

OpenWiNo, associated to the WiNo hardware, has certain advantages, compared with the other platforms and testbeds described in the section III. As presented before, the main advantage of OpenWiNo is the simplicity of transceiver switching; To add the support of a given transceiver, the developer must get the primitive set of the transceiver, basically how to configure the transceiver (setting channel frequency, transmission power), send a frame, set receive mode, get a received frame and put transceiver in sleep mode. Considering specific protocols, some functionalities may be mandatory, such as sensing energy on medium for CSMA-based protocols, timestamping frame reception for synchronisation protocols, etc. Choosing the best transceiver is an important matter and may impact the MAC protocol under study. At this time, 4 different transceivers – enabling 4 different PHY layers – have been tested successfully with OpenWiNo:

- IEEE 802.15.4-2011 UWB (DecaWave DW1000),
- Proprietary 433MHz FSK/GFSK (HopeRF RFM22b),
- LoRa mode 868MHz (HopeRF RFM95),
- Classical IEEE 802.15.4 2.4GHz DSSS (Freescale).

At MAC level, the CSMA/CA from the IEEE 802.15.4-2006 standard is available in OpenWiNo. A non-hierarchical TDMA MAC, based on the SISP protocol [15] is also available. At NWK level, an implementation of a reactive routing protocol is available; Static routing is also possible. At APL level, an API to the Arduino sketch is available, implementing the standard send() and recv() primitives. Various mechanisms such as end-to-end acknowledgments and packet reordering/FiFos are also implemented.

Another advantage of OpenWiNo is the simplicity of local testbed deployment, with a lightweight infrastructure, on a real environment such as an industrial area like a factory. In the lab, a complete testbed of 16 WiNoRF22 has been deployed on a 200m² indoor zone, for a reasonable price, i.e. less than 2kEUR, including controllers. Fig. 5 illustrates the logical topology of this deployment, from the MAC layer point of view: We can see asymmetric links, which the MAC layer must deal with. The value on the links is the RSSI before dBm conversion.

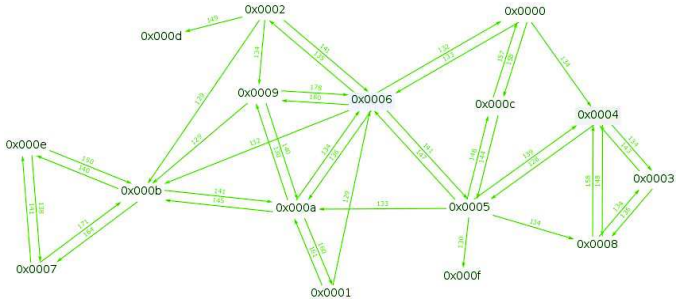

Fig. 5. Our 16-nodes testbed topology

In addition to be flexible with the transceiver replacement, the WiNo is also versatile on electronic usage: it is possible to connect external sensors or use I/O of the MCU; for example, by driving the General Purpose I/Os (GPIOs), it is simple to represent protocol states. For example, Fig. 6 illustrates the representation of a typical industrial time-constrained TDMA MAC slotting by asserting a single GPIO on each node of the testbed: Each node sets a dedicated GPIO at the beginning of its TDMA slot, than clears the GPIO at the end of its slot. The result is viewed with a logic analyzer and enables very precise measurements on timing, such as synchronisation and slotting at MAC-layer. The performance evaluation is complementary of results obtained by simulation.

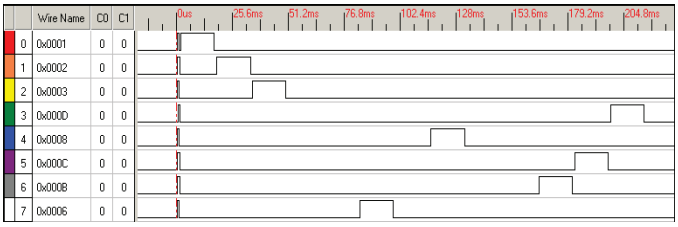

Fig. 6. Timing representation of a TDMA scheduling

Nevertheless, OpenWiNo also has few shortcomings, such as the lack of library implementing the most standardized protocols or the absence of Operating System; even if this last drawback is a choice at the foundation of OpenWiNo to get a highest level of portability on new hardware, the absence of OS can complicate the implementation of processes and tasks.

VI. Conclusion and Perspectives

In the coming years, the IoT Device Layer will no doubt be based on a variety of wireless technologies, according to the application and to the use of each of the connected devices: WiFi, Bluetooth BLE, Zigbee, NFC, Ant, 4G, IEEE 802.15.6, etc… will each have their role to play in this smart world. The prototyping platforms and the protocol and design tools in general, will therefore have to take this variety of technologies into account. Some platforms, such as OpenWiNo, FIT/IoT-LAB or SmartSantander, are already a part of this trend: they incorporate several PHY and MAC layers, with the eventual possibility of easily and quickly supporting new ones. In the case of WiNo, this is done very simply by swapping transceivers and associated libraries, or by programming new MAC layers, allowing for innovation in the IoT. The node architectures will move towards smaller but more powerful and more energy-efficient processors, since autonomy remains a major challenge for energy-constrained nodes: the new-generation platforms will have to incorporate this constraint for connected devices to be accepted by users. In addition to providing a platform for network protocol evaluation, OpenWiNo is designed to facilitate pluridisciplinary projects where the aim is not only to evaluate performance through traditional networking metrics but also to investigate the user experience. Although registered users can remotely access the existing OpenWiNo deployment, the main objective is to allow research teams to easily deploy their own testbed in a representative environment, while incorporating the most appropriate sensors and actuators for the target application. For

example, in the next months, WiNo+OpenWiNo solution will be tested in a real application to enable wireless communications between autonomous candelabras.

In the future, we plan on giving potential users a taste of OpenWiNo by allowing registration-based web access to our testbed. Another perspective is to develop a Hardware Adaptation Layer suitable for reference hardware platforms such as the ones found on FIT/IoT-LAB; this will allow the execution of protocols implemented on OpenWiNo on the FIT/IoT-LAB testbed. A final perspective is the development of a driver for a Software Defined Radio-based Physical layer; In addition to using the open software approach on the hardware, this will give an impulse to research on the opportunistic use of the radio interface.

ACKNOWLEDGMENT

The authors want to thank Snootlab, Toulouse Tech Transfer and Julien Aubé for their support on the industrialisation of OpenWiNo source code. Thanks to Snootlab for the finalisation and the marketing of the first WiNo. OpenWiNo source code will be available on http://openwino.cc as soon as WiNo hardware is available on http://snootlab.com.

REFERENCES

[1] Xu B, Xu LD, Cai H, Xie C, Hu J, Bu F, (2014) Ubiquitous Data Accessing Method in IoT-Based Information System for Emergency Medical Services, IEEE Transactions on Industrial Informatics, 05/2014, pp 1578-1586

[2] Lunardi WT, De Matos E, Tiburski RT, Amaral LA, Marczak S and Passuelo Hessel F, Context-based search engine for industrial IoT: Discovery, search, selection, and usage of devices. 20th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA'15, September 8-11, 2015, Luxembourg

[3] Jaewoo K, Jaiyong L, Jaeho K, Jaeseok Y, (2014) M2M Service Platforms: Survey, Issues, and Enabling Technologies, IEEE Communications Surveys & Tutorials, 01/2014, pp 61-76

[4] Khan R, Khan SU, Zaheer R, Khan S, (2012) Future Internet: The Internet of Things Architecture, Possible Applications and Key Challenges, in Frontiers of Information Technology (FIT), 2012 10th International Conference on , vol., no., pp.257-260, 17-19 Dec. 2012 doi: 10.1109/FIT.2012.53

[5] Ding Y, Jin Y, Ren L, Hao K, (2013) An Intelligent Self-Organization Scheme for the Internet of Things, IEEE Computational Intelligence Magazine, 01/2013 pp 41-53.

[6] Tang C, Song L, Balasubramani J, Wu S, Biaz S, Yang Q, Wang H, (2014) Comparative Investigation on CSMA/CA-Based Opportunistic Random Access for Internet of Things, Internet of Things Journal, IEEE 01/2014, pp171-179.

[7] Tonneau AS, Mitton N, Vandaele J, (2015) How to choose an experimentation platform for wireless sensor networks? Elsevier Adhoc networks, Elsevier, 2015, 30, pp.12.

[8] VITAL: The future of Smart Cities. http://vital-iot.eu/ last accessed on 20/02/2016

[9] Adjih C and al. (2015) FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed, IEEE World Forum on Internet of Things (IEEE WF-IoT), Milan, Italy, 14-16 December 2015.

[10] Fourty N, van den Bossche A, Val T (2012), An advanced study of energy consumption in an IEEE 802.15.4 based network: everything but the truth on 802.15.4 node lifetime. Computer Communications, Elsevier, June 2012.

[11] Chauvenet C, Tourancheau B, Genon Catalot D, (2013) Energy Evaluations for Wireless IPv6 Sensor Nodes. SENSORCOMM 2013, 7th International Conference on Sensor Technologies and Applications, 2013, pp.97-103 <hal-01073749>

[12] Van den Bossche A, Val T (2013) WiNo : une plateforme d'émulation et de prototypage rapide pour l'ingénierie des protocoles en réseaux de capteurs sans fil. 9th Journées Francophones Mobilite et Ubiquité (UbiMob 2013), June 2013, Nancy, France <hal-01240700> (in French)

[13] van den Bossche A, Dalcé R Fofana NI, Val T (2016), DecaDuino: An Open Framework for Wireless Time-of-Flight Ranging Systems, IEEE/IFIP Wireless Days, WD 2016, Toulouse, 23/03/2016-25/03/2016

[14] DecaWave Ltd. 2013, DWM1000 datasheet, last accessed on 20/02/2016

[15] Van Den Bossche A, Val T, Dalce R (2011), SISP: A lightweight synchronization protocol for Wireless Sensor Networks, 16th IEEE Conference on Emerging Technologies & Factory Automation (ETFA), 5-9 Sept. 2011 doi: 10.1109/ETFA.2011.6059182