FIT IoT-LAB: A Large Scale Open Experimental IoT Testbed

Cedric Adjih*, Emmanuel Baccelli*, Eric Fleury[†]*[§], Gaetan Harter*, Nathalie Mitton*, Thomas Noel[‡][¶],
Roger Pissard-Gibollet*, Frederic Saint-Marcel*, Guillaume Schreiner[‡],Julien Vandaele*, and Thomas Watteyne*
**Inria, France*
[†]*ENS de Lyon, France*
[‡]*University of Strasbourg, (UMR 7357 CNRS - ENGEES - INSA), France*
[§]*University of Lyon (UMR 5668 CNRS - ENS de Lyon - UCB Lyon 1 - Inria), France*

***Abstract*—This paper introduces the FIT IoT-LAB testbed, an open testbed composed of 2728 low-power wireless nodes and 117 mobile robots available for experimenting with large-scale wireless IoT technologies, ranging from low-level protocols to advanced Internet services. IoT-LAB is built to accelerate the development of tomorrow's IoT technology by offering an accurate open-access and open-source multi-user scientific tool. The IoT-LAB testbed is deployed in 6 sites across France. Each site features different node and hardware capabilities, but all sites are interconnected and available through the same web portal, common REST interfaces and consistent CLI tools. The result is a heterogeneous testing environment, which covers a large spectrum of IoT use cases and applications. IoT-LAB is a one-of-a-kind facility, allowing anyone to test their solution at scale, experiment and fine-tune new networking concept.**

***Keywords*-IoT; Testbed; Low Power Wireless Sensor Network; Robot; Mobility; Large Scale; Open Platform**

I. Introduction

IoT is not an emerging technology trend anymore, it is a reality today. IoT is a tremendous paradigm shift compared to the traditional Internet: it is an evolution from pure end-to-end communication between end-users and servers on the Internet, to Internet-enabled physical objects freely able to communicate with each other and with humans. However, all of this comes with important technical and scientific questions. *How does one ensure the reliability of wireless communications in dynamic environments? How does one allocate constrained resources? How does that translate in sustainable platform maintenance?* IoT solutions need to be thoroughly tested and fine-tuned before hitting the market.

To overcome this critical hurdle, FIT IoT-LAB[1] offers a large-scale federated experimental platform[2] allowing researchers, IoT designers, developers and engineers to construct, benchmark and optimize their protocols, applications and services. As a state-of-the-art testbed, IoT-LAB's goal is to answer the needs and requirements of today's and tomorrow's IoT technology. In particular, it offers: *(i)* a heterogeneous and rich environment (e.g. hardware, topologies, OS, up-to-date standardized protocol stacks and libraries) applicable to a large spectrum of IoT applications; *(ii)* the ability to manage, interact with and monitor running experiments; *(iii)* the ability to instrument an experiment, through visualization and reproducibility tools.

The IoT-LAB project is driven by a team that strongly believes in open-source. The software used in the back-end to manage the platform, all software tools and examples to help users get started quickly, and the design of the hardware used are published open-source. IoT-LAB is entirely open-access and free of charge, available to academia, for teaching purposes, and to industry alike to develop and test IoT applications.

IoT-LAB contributes to the opening of research on networking by solving the challenges that must be overcome to allow Open Science and Reproducible Research in this field, improving description and publication of experiments and their results, facilitating the analysis, comparison and reproducibility of results. We aim at conducting research in the spirit of free and open-source software/hardware. It favors the reproducibility of experiments and results, and the general increase of the quality of experiments.

The remainder of this paper is organized as follows. Section II presents the state of the art in large-scale testbeds. Section III presents an overview of the platform. Sections IV and V introduce the hardware and software tools available to a user, respectively. Section VI illustrates the potential of IoT-LAB through a number of real-world studies done using the IoT-LAB. Finally, Section VII presents concluding remarks and discussions.

II. State of the Art

In [1], the authors survey relevant experimental wireless sensor network testbeds available to the community today. The most advanced and active testbed in that survey is the "SmartSantander" project which offers an experimental research facility at the scale of a city, and which can be used to test smart city applications and services. The SmartSantander project targets a 20,000-node network. On the other hand, IoT-LAB is a federation of platforms, and is part of OneLab (Internet-overlaid, Broadband access, wireless & IoT). In the SmartSantander project, all nodes – called "service nodes" (battery-constrained nodes) – only produce data and can be configured only by the administrators of the

[1] https://www.iot-lab.info/

[2] Integrated in OneLab: https://onelab.eu

978-1-5090-0366-2/15/$31.00 ©2015 IEEE

testbed. They are, however, not open to be reprogrammed by users. Some nodes (with fewer battery constraints) can be reprogrammed so users can test their own protocols. Both approaches are complimentary: SmartSantander targets smart city applications and experimentation at the IoT application level, IoT-LAB give bare-metal access to all nodes in all sites.

III. Overview of the FIT IoT-LAB Platform

With over 2700 wireless sensor nodes, included 117 mobile robot nodes deployed across six sites in France, IoT-LAB is the largest open low-power wireless remote testbed in the world today. Table I summarizes the number of nodes deployed in each site, and highlights the 5 different hardware platforms available. Some nodes (200 in Rocquencourt, and 32 Grenoble) are equipped with GPS. An indoor GPS relay antenna is installed in both sites, and a GPS module is connected to each of these nodes. The GPS signal is not used for localization (these nodes are static) but for synchronization. The GPS modules feed each node with a pulse-per-second (PPS) line, allowing them to synchronize to within $\pm 10ns$ of one another.

Nodes are either static or mobile, and can be allocated in various topologies throughout all sites. Wireless networks are often mobile, and mobility can significantly affect the performance of network protocols (e.g. multi-path fading or routing). The ability to move the position of the robots during an experiment allows a user to quantify the effect of node mobility. Like their static counterparts, mobile nodes can be reserved, reprogrammed and monitored. The user can also control the movement of the robots.

A variety of wireless sensors are available, with different processor architectures and wireless chips (see Section IV-B). The nodes are fully programmable. The management interface allows a user to load arbitrary binaries onto the devices and have direct access to the gateways to which nodes are connected. Using web-based or command-line tools, a user can reserve an arbitrary number of nodes in one or several sites to run experiments. Once reserved, the user deploys his/her own firmware onto the nodes. Additional tools allow the user to configure some nodes to act as gateway nodes connected to the Internet, monitor each node's energy consumption, and measure other metrics such as end-to-end delay or throughput.

IV. FIT IoT-LAB Hardware

A. Architecture of an IoT-LAB Node

In order to enable a user to have full "bare-metal" access to IoT nodes, the IoT-LAB infrastructure must satisfy requirements such as reliable access to all nodes, non intrusive and application transparent real-time monitoring, real-time control of the experiments, security and data integrity. Based on these requirements, the IoT-LAB testbed consists of "IoT-LAB nodes" interconnected through a backbone. The backbone provides power to the nodes and connects them to the IoT-LAB back-end servers. The management software provides real-time access to the nodes, so a user can reset, reprogram, and monitor the state of each node. As depicted in Fig. 1, an IoT-LAB node consists of three components:

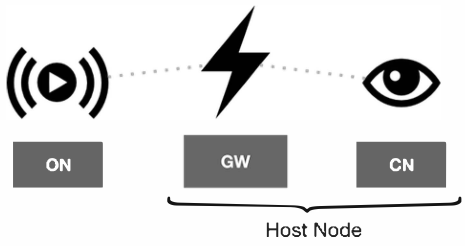
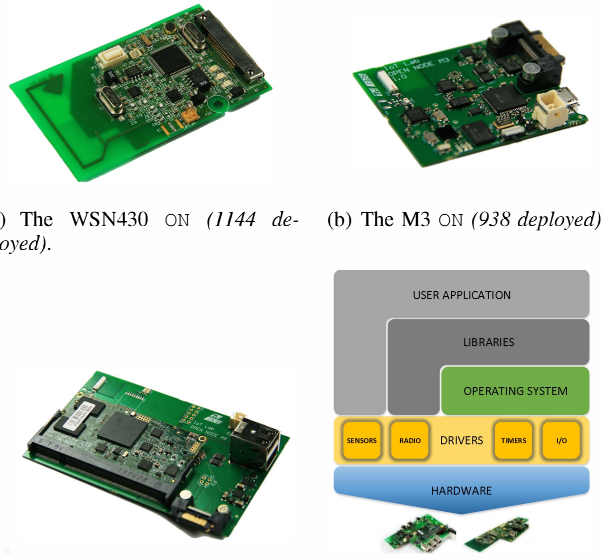


Figure 1. An IoT-LAB node is composed of an Open Node (ON), a Gateway (GW) and a Control Node (CN).



(a) The WSN430 ON *(1144 deployed).*

(b) The M3 ON *(938 deployed).*

(c) The A8 ON *(550 deployed).*

(d) User embedded software.

Figure 2. The different Open Nodes deployed in the IoT-LAB and software support.

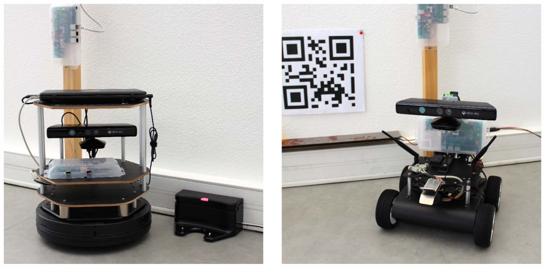
The **Open Node** (ON) is the low-power device the user reprograms, and to which he/she has "bare-metal" access. This node's serial port is connected to the Gateway.
The **Gateway** (GW) is a small Linux computer connected to the serial port of both the ON and the Control Node. It is itself connected to the backbone over Ethernet. On top of monitoring (and reprogramming) the ON, it forwards the ON's serial activity to the back-end servers.
The **Control Node** (CN) coordinates ON reprogramming, starts/stops/resets it and selects its power source (battery or mains). In addition, the CN monitors the ON's power consumption, drives its sensors, can serve as a wireless packet sniffer or inject arbitrary packets traffic.

B. Types of IoT-LAB Nodes

To offer a development environment which is representative of today's range of IoT hardware, three types of IoT-

	Grenoble	Lille	Rocquencourt	Strasbourg	Rennes	Paris	**Total**
WSN430 ON (862MHz)	256	-	-	256	-	-	512
WSN430 ON (2.4GHz)	-	256	120	-	256	-	632
M3 ON	384	320	24	120	-	90	938
A8 ON	256	-	200	24	-	70	550
Generic host node (no ON)	32	64	-	-	-	-	96
Total IoT-LAB nodes	928	640	344	400	256	160	2728
Turtlebot	8	32	-	45	-	-	85
Wifibot	-	32	-	-	-	-	32
Total mobile robots	8	64	-	45	-	-	117

Table I
THE NUMBER OF IOT-LAB NODES AND ROBOTS AVAILABLE TODAY IN THE FIT IOT-LAB.



(a) The "Turtlebot" mobile node.

(b) The "Wifibots" mobile node.

Figure 3. The robot models deployed in the IoT-LAB.

LAB Nodes are deployed, differing by the capabilities of their ON:

The WSN430 ON (Fig. 2(a)) features a 16-bit MSP430F1611 micro-controller, a CC2420 (2.4GHz version) or CC1101 (868MHz version) radio chip, and sensors (light/temperature).

The M3 ON (Fig. 2(b)) features a 32-bit ARM Cortex-M3 micro-controller (STM32F103REY), an AT86RF231 IEEE802.15.4 radio chip, and sensors[3]. The user can interact with the M3 ON over its serial port and also debug it using the JTAG protocol. The M3 ON is representative of today's state-of-the-art IoT devices.

The A8 ON (Fig. 2(c)) is the most powerful node in the IoT-LAB. It features a 32-bit ARM Cortex-A8 600 MHz mini-computer, a 32-bit ARM Cortex-M3 micro-controller, an AT86RF231 IEEE802.15.4 radio chip and sensors. The A8 ON is representative of more advanced devices such as set-top boxes, smartphones, or tablets and may run Linux or Android. Moreover, a GPS module can be connected to the Cortex-A8 to provide accurate synchronization.

In addition to static nodes, 117 mobile robots are deployed in the IoT-LAB. There are two types of robots:

The Turtlebot (Fig. 3(a)) is an open-source robot with a maximum speed of 0.7m/s and an odometry of 2578.33 ticks/wheel. An infrared beam allows the robot to find its docking station, where it recharges.

The Wifibot (Fig. 3(b)) is a high-mobility 4x4 platform with 4 brushless motors and 4 hall encoders. The robot uses its two infrared sensors and camera to align in front of the docking station, marked by a QR code, move forwards, and shut down its motors once it starts charging.

Operation of the robots is fully automated, and available for the users at all times. When not in an active experiment, a robot returns to its docking station, where it charges its batteries. Similarly to other nodes, a user can reserve one or more mobile robots in one or several sites. When the experiment starts, the robot leaves its docking station and starts following its circuit. During the experiment, the user can retrieve the position of the robot in real-time through a programmatic interface. At the end of an experiment, the robot returns to its docking station.

Each mobile robot is equipped with an M3 ON. Exactly like a static M3 ON, the user loads custom firmware on it.

V. FIT IoT-LAB SOFTWARE

Two classes of software run on the IoT-LAB. The first includes the source code, libraries and tools available for a user. The second is the "back-end" software which glues together all the pieces of the IoT-LAB (e.g. managing users, coordinating the reservation of nodes, reprogramming the nodes, and feeding back the experimental results to the user).

A. Software for the User

The firmware a user can develop on the IoT-LAB nodes may be composed of drivers, an operating system, libraries and applications, as depicted in Fig. 2(d).

Low-level Drivers & Libraries: A developer has full access to a node's hardware, including the schematics of the boards and datasheets of the components. We have developed and maintain code snippets, libraries, APIs and development environments. In particular, the OS-independent drivers – a thin layer of C-code – gives access through an API to all hardware modules on the nodes: radio chip, physical sensors, serial buses, digital and analog interfaces, and device timers. Wireless communication libraries offer simple and useful APIs that MAC protocol implementations can use.

Embedded Operating Systems: Support for five popular IoT operating systems is maintained (see Table II) including:

[3] Light, pressure, temperature, 3-axis gyroscope, 3-axis accelerometer, 3-axis magnetometer.

OS	WSN430 Node	M3 Node	A8 Node
RIOT	✓	✓	✗
OpenWSN	✓	✓	✗
FreeRTOS	✓	✓	✗
Contiki	✓	✓	✗
TinyOS	✓	✗	✗
Linux	✗	✗	✓

Table II
SUPPORTED OPERATING SYSTEMS, KERNELS AND NETWORK STACKS ON IOT-LAB DEVICES.

RIOT: an open-source OS that aims at providing a powerful, but very low memory footprint software platform. RIOT offers IPv6/6LoWPAN/RPL/UDP/CoAP network stack and also provides a CCN stack [2].

OpenWSN: an open-source implementation of a full protocol stack based on IoT standards (IPv6, 6TiSCH, 6LoWPAN, UDP, RPL, CoAP). OpenWSN is the de-facto open-source implementation of IEEE802.15.4e TSCH.

FreeRTOS: a popular micro-kernel providing multi-threading, mutexes, semaphores and software timers, with small memory footprint, low overhead, and fast execution. The OpenLAB project has developed MAC layer implementations (CSMA/TDMA) on top of FreeRTOS.

B. Back-End Software

An important part of the development is invisible to the user. Part of it is embedded in the hardware that supports the open nodes: the gateway and control nodes. Most of the back-end software runs on the servers in the backbone network. It is in charge of user management, resource allocation and experiment scheduling. The main objective of the back-end software is to allow a user to interact with all IoT-LAB nodes in a reliable and real-time fashion.

Firmware on the Control Node: The control node firmware is in charge of switching the open node on/off, and monitor its activity. This node can monitor the open node's energy consumption in real-time. It can monitor the radio activity, including RSSI power signal measurements. A user can configure the frequency to listen to and the number of measurement per unit of time. The user can also activate a radio sniffer mode. In this mode, the control node listens for incoming packets, and forwards those to the gateway, encapsulated in the ZigBee Encapsulation Protocol (ZEP). Tools such as Wireshark can be used to dissect the received packets in real time.

Firmware on the Gateway Node: The gateway node runs an application which offers a RESTful management interface to the outside world. This interface offers all the testbed's API commands. Measurements collected by the control node are collected using an open-source measurement library [3]. The gateway node is connected to the ON's JTAG port and runs an OpenOCD GDB server. This allows a user to remotely use JTAG debugging to place breakpoints and inspect variables directly on the (remote) open node.

Firmware on the Robots: The mobile robots run the ROS Linux distribution [4], which includes a navigation stack implementing AMCL, and the IoT-LAB RESTful management interface. Every robot uses dead reckoning (by counting the number of wheel rotations) to keep track of its location. The robot logs its position using the OML format, and synchronizes it with the back-end infrastructure over WiFi. Tools are available for drawing the path of the robot.

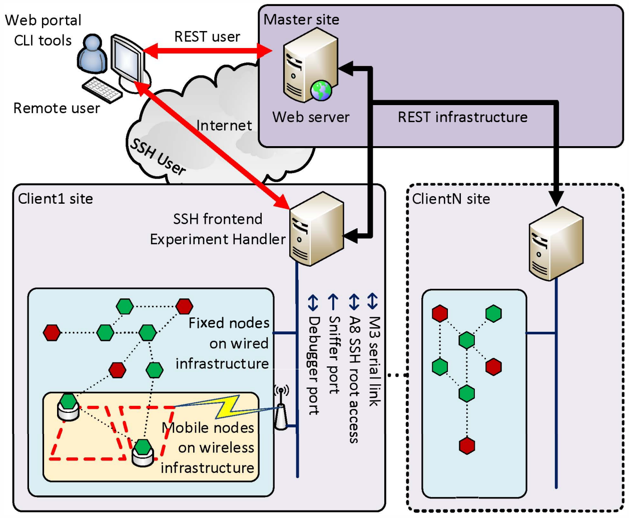


Figure 4. IoT-LAB Infrastructure.

Infrastructure Software: The IoT-LAB infrastructure consists of a master site (located in Grenoble) and 6 deployment sites, all sites are interconnected by a VPN. The master site hosts the servers responsible for services such as user authentication (LDAP directory tree) and a private domain name server (DNS) system. Its main application consists of a RESTful interface exposing the testbed interfaces to the user. A dispatcher is in charge of dispatching the requests to the different deployment sites. This application interacts with OAR Batch scheduler software [5], an open-source resource manager for large clusters, which allows optimal experiment scheduling and resource allocation. This is the application responsible for starting and stopping experiments.

C. The IoT-LAB User Experience

A user starts by creating a user account on the IoT-LAB's main website or using his/her OneLab login. He/she then communicates with IoT-LAB through its web portal or REST API that are available through CLI to offer a simpler user experience.

To launch an experiment, the user reserves one or more nodes, in one or more sites, for some duration. An interactive map allows the user to select individual nodes or he/she can ask for the first nodes available of a particular type. To each node are attached some features: mobile/fixed, location and radio chip. After reserving the nodes, the user can

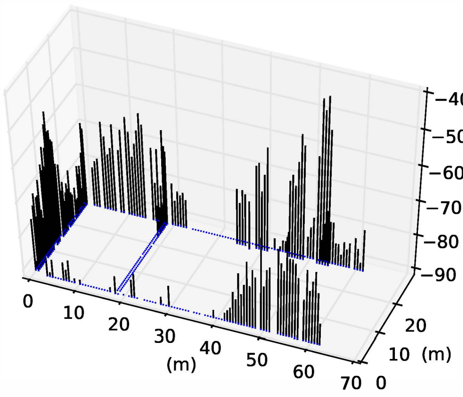


Figure 5. Max RSSI measurements on IEEE802.15.4 channel 17 ($2.435GHz$) reveals the presence of WiFi access points.

make additional configurations for the experiments, including whether to use any monitoring tools (power consumption, wireless sniffer), how the nodes should be powered (battery/mains), and specify the firmware image(s) to load on the nodes. The scheduler starts the experiment as soon as the nodes are available, or at a precise date, if specified. Once an experiment is running, the user has full control over the nodes reserved and dedicated to him/her. Nodes can be reseted, reconfigured, or reprogramed individually or globally. A user can control the experiment directly using CLI tools, quickly edit source code and build/deploy firmware, access a node's serial port, log into the A8 nodes over ssh, remotely debug the M3 nodes using OpenOCD and gdb, recover power consumption measurements in OML files, and analyze the sniffed radio traffic using Wireshark. A wealth of documentation is available IoT-LAB website, including tutorials[4] and a wiki[5].

VI. Example Use Cases

A. *Visualizing the Impact of WiFi Traffic*

Nodes in the Grenoble testbed are deployed in a typical office environment, alongside WiFi access points. As an example, we detail how we use the IoT-lab Grenoble to visualize the impact of WiFi traffic on an IEEE802.15.4 network. We developed and deployed firmware that scans the RSSI on all nodes to detect the presence of other technologies operating on the same frequency (IEEE802.15.4 channel 17, $2.435GHz$). Although no IoT-LAB nodes are transmitting, nodes are still registering significant background noise (Fig. 5).

To better understand the impact of these WiFi access points on the communication in the IEEE802.15.4 network, we have one node in the testbed transmit 100 50-bytes frames. One frame is transmitted every $2.5ms$. We configure all the other nodes to listen on the same frequency as before and count how many packets are received. In Fig. 6, the node identified by a red line is transmitting; the z-axis represents the percentage of packets received. A value of 1 indicates that all transmitted packets were received (this is the case for several nodes close to the transmitter).

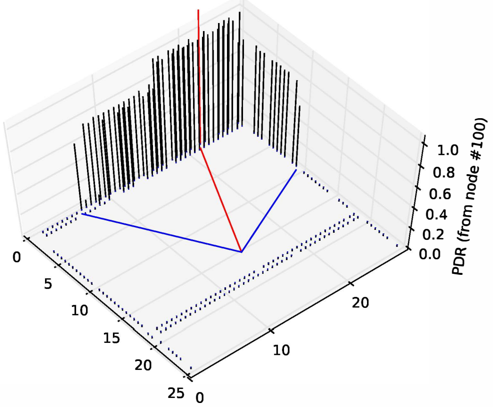


Figure 6. The number of frames received by each node when one node (identified by the red bar) transmits a burst of 100 frames.

A detailed analysis [6] shows that packet losses are strongly correlated for nearby nodes. Two WiFi access points are emitting IEEE802.11 beacons every $100ms$ to advertise the presence of the network to possibly roaming WiFi devices. Because the transmission power of a WiFi AP is roughly 100 times higher than that of an IEEE802.15.4 node, each time a WiFi beacon is transmitted, it prevents nearby IEEE802.15.4 nodes from receiving their frames.

B. *Smart Floor Demonstration*

The scenario of smart floor tile detection (see video[6]) is a good illustration of how to use the M3 node sensors located under the floor tiles in order to detect and track either people or mobile robots. In this example, the accelerometer detects the vibrations from a person walking on the floor tile located above it, the magnetometer detects the passage of a mobile robot through the magnetic perturbations its electrical motors generate. In addition to using these sensors, we implemented RF localization: the mobile node on the robot periodically sends packets which are received by nearby fixed IoT-LAB nodes. The IoT-LAB architecture has inspired the Intelligent Tiles Infrastructure deployed in the LORIA smartroom [7].

This study allows us to compare two techniques for detecting the presence of a robot: RF localization, and using the deflection of the magnetic field. During the experiment, the robot follows a circuit alternating straight lines and turns while traveling through a corridor. The localization algorithm built into the robots records its path with an accuracy better than $5cm$ (see Fig. 7), providing a ground truth for the experiment. Fig. 8 shows the real trajectory

[4] https://www.iot-lab.info/tutorials/

[5] https://github.com/iot-lab/iot-lab/wiki

[6] https://www.youtube.com/watch?v=IPxTfgNBjsI.

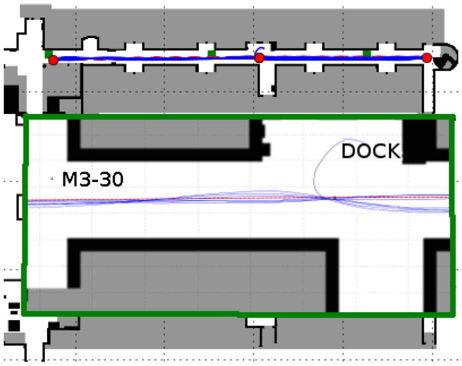


Figure 7. Robot trajectory recorded in a corridor.

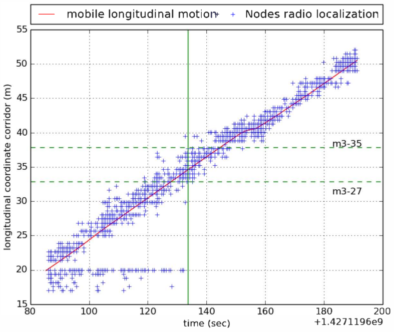


Figure 8. Robot localization using radio packet compared to its longitudinal motion on the corridor.

of the robot (line), and its measurement locations using the packet-based approach (crosses). Location accuracy varies greatly: when the robot is close to node `m3-30`, its presence is detected by 8 nodes, yielding a localization accuracy around $5m$. Accuracy is much worse at the end of the corridor, as less nodes hear the beacons broadcast by the robot. Fig. 9 shows the measurements gathered by node `m3-30` when using magnetometer-based localization. It depicts the distance between the robot and the node as the robots follows it trajectory. Circles represent when the magnetometer detects the presence of the robot. Magnetometer-based detection only happens when the robot is very close to the measuring node, yielding a localization accuracy around $10cm$. The study illustrates how IoT-LAB tools enable users to efficiently construct sophisticated algorithms and easily perform related experiments starting from core building blocks, in this case radio communication, motion sensors, indoor location, and mobile robots.

VII. Conclusion

This paper describes FIT IoT-LAB, a large scale open-access IoT testbed. IoT-LAB is operational today, and offers free-of-charge access to thousands of wireless IoT devices, which users can manipulate remotely through convenient tools, program with arbitrary firmware, launch repeatable experiments, and gather result. A team of full-time engineers maintains the testbed, and the community around IoT-LAB uses state-of-the-art online cooperation tools to file bug reports and suggestions. Cooperation is augmented though a wiki, and active mailing lists (which several hundred people subscribed today). These characteristics make IoT-LAB a extremely useful platform for benchmarking, validating and testing software deployed at scale on heterogeneous IoT hardware. IoT-LAB is used today as a tool for designing, evaluating and contributing to standardization in the IoT realm.

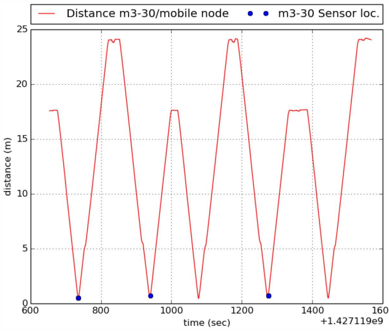


Figure 9. Node `m3-30` Magnetometer robot detection.

References

[1] A.-S. Tonneau, N. Mitton, and J. Vandaele, "How to Choose an Experimentation Platform for Wireless Sensor Networks? A Survey on Static and Mobile Wireless Sensor Network Experimentation Facilities," *Ad Hoc Networks (Elsevier)*, 2015.

[2] E. Baccelli, C. Mehlis, O. Hahm, T. C. Schmidt, and M. Wählisch, "Information Centric Networking in the IoT: Experiments with NDN in the Wild," in *ICN*. ACM, 2014.

[3] M. Singh, M. Ott, I. Seskar, and P. Kamat, "ORBIT Measurements Framework and Library: Motivations, Implementation and Features," in *TridentCOM*. IEEE, February 2005.

[4] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Ng, "ROS: an Open-source Robot Operating System," in *IEEE ICRA*, 2009.

[5] N. Capit, G. Da Costa, Y. Georgiou, G. Huard, C. Martin, G. Mounié, P. Neyron, and O. Richard, "A Batch Scheduler with High Level Components," in *CCGrid*, vol. 2, 2005.

[6] T. Watteyne, C. Adjih, and X. Vilajosana, "Lessons Learned from Large-scale Dense IEEE802.15.4 Connectivity Traces," in *IEEE International Conference on Automation Science and Engineering (CASE)*, Gothenburg, Sweden, 24-28 August 2015.

[7] N. Pepin, O. Simonin, and F. Charpillet, "Intelligent Tiles: Putting Situated Multi-Agents Models in Real World," in *ICAART*. ACM, January 2011.